MASTER

Die shape optimization for extrudate products

Kortenhoeven, Martijn J.

Award date:
2022

Die shape optimization for extrudate products

Master's thesis

Student:	M.J. Kortenhoeven
IDNR:	1383434
Master:	Systems and Control
Department:	Mechanical Engineering
Research Group:	Control Systems Technology
Report ID:	CST2022.012
Supervisors:	dr. ir. T.A.C. van Keulen prof. dr. ir. W.P.M.H. Heemels prof. dr. ir. P.D. Anderson

Breda, 8 April 2022

[This page intentionally left blank]

Declaration concerning the TU/e Code of Scientific Conduct for the Master's thesis

I have read the TU/e Code of Scientific Conduct[i].

I hereby declare that my Master's thesis has been carried out in accordance with the rules of the TU/e Code of Scientific Conduct

Date

06-04-2022

Name

M.J. Kortenhoeven

ID-number

1383434

Signature

Submit the signed declaration to the student administration of your department.

[i] See: https://www.tue.nl/en/our-university/about-the-university/organization/integrity/scientific-integrity/
The Netherlands Code of Conduct for Scientific Integrity, endorsed by 6 umbrella organizations, including the VSNU, can be found here also. More information about scientific integrity is published on the websites of TU/e and VSNU

[This page intentionally left blank]

Preface

With the completion of this research my formal education comes to an end. It has been a big part of my life for as long as I can remember. Some years ago, during my graduation of the bachelor Mechatronics at Avans Hogeschool Breda, I felt there was still more to learn for me. That although I learned to be pragmatic, I was lacking knowledge on a lot of mathematical concepts to help me thoroughly understand the topics we were being taught. To acquire this knowledge I started the master Systems and Control at the Eindhoven University of Technology. Little did I know, that this new knowledge not only gave me a better understanding of those topics, but it also made me a more confident engineer, and made me see relations I would have missed otherwise.

Although I had to make some sacrifices to pursue my master's degree, it was definitely worth it. I met some great new people, and new doors have opened up for me. One of which is a job offer for a great position at ASML, one of the most prominent companies in The Netherlands. A job offer I would not have received if not for my master's and the people I met along the way.

Outside of school, my graduation period has been eventful; I have had some high highs and some low lows, and have learned a lot about myself. One low I would like to mention is the recent passing of my grandfather, whom I owe my middle name to and whom I would have very much liked to have attended my defense.

First and foremost, I would like to thank Thijs van Keulen, my supervisor for over a year and a half now. He has been a pleasure to work with and has provided me with great insight, knowledge and support during our countless meetings. We even submitted a paper together; fingers crossed it will be accepted. I would like to thank Maurice Heemels for his believe in me and the opportunities he provided. The courses System Theory for Control and Hybrid Systems and Control were definitely my favorites, owing to his passion, humor and his ability to explain difficult subjects in a simple manner. Michelle Spanjaards and Patrick Anderson were the foundation for my graduation project. They crossed into our territory of control theory to successfully solve a complex optimization problem, and were able to help me in the process of applying other control techniques. Michelle and Patrick, thank you for your guidance and support during my graduation; it was always a pleasure to speak to you. To all, I sincerely hope that our paths will cross again.

Martijn Jacobus Kortenhoeven
Breda, 3 April 2022

[This page intentionally left blank]

Abstract

Extrusion is a common production technique in the polymer processing industry to obtain products with a desired cross-section. In this process a polymer is molten and pushed through a die with a certain cross-sectional shape, to obtain a product (extrudate) with a desired cross-sectional shape. A common requirement on the extrudate is dimensional precision. However, the dimensions of the extrudate are highly influenced by a phenomenon called extrudate swell, where the extrudate starts to expand due to internal stresses in the polymer once it leaves the die. Die shape optimization is the term for finding a die shape that corresponds to the desired extrudate shape once the swelling process is complete and the extrudate shape remains constant. Examples of products that are produced with polymer extrusion are pipes, tubes, sheets, films, structural parts and tires.

Recently published research focuses on die shape optimization using feedback control, where the die shape is adjusted online to obtain the desired extrudate shape. The employed controller is in the form of an integrator, where the difference between the desired and measured extrudate shape is added at discrete time intervals. The control scheme is implemented in a digital environment and uses the Finite Element Method to calculate the extrudate shape.

The shape of the extrudate is measured at a different location than the die exit, as to allow the extrudate to settle to a constant shape. This results in a significant transportation delay between input and output. In addition, the nonlinear underlying dynamics of the extrudate shape are treated as black box, such that the assumption is imposed that the process is stable. These challenges resulted in the need for a controller that is conservatively tuned.

As such, there is a need for improvements of the existing control structure, to either guarantee stability and/or improve the rate of convergence to the die shape that results in the desired extrudate shape. Three optimization methods are applied in this research: (1) extremum seeking, a method that perturbs the input such that local gradient estimation of a cost function can be obtained, guaranteeing stability under certain conditions, (2) surrogate modeling identifies input-output relations using a limited number of constant inputs and subsequently estimates the optimal die shape, and (3) the Smith Predictor, a dead-time compensation control structure that seeks to diminish the effect of the transportation delay, and as such is able to converge to the optimum die shape more quickly.

Experimental simulation results show a significant increase in rate of convergence for surrogate modeling and the Smith Predictor. Surrogate modeling reduces the convergence time by approximately 60% and the Smith Predictor by 82%. The Smith Predictor does however require a model of the system and needs to be conservatively tuned for the best convergence rate, robustness and to allow the same model to be used on different polymers. Extremum seeking however, is very slow compared to surrogate modeling, the Smith Predictor and even the integrative controller from the state of the art. On the other hand, the extremum seeking scheme guarantees stability and convergence.

Contents

[This page intentionally left blank]

Chapter 1

Introduction

Extrusion is a common production technique in the polymer processing industry to obtain products with a desired cross-section. In this process a polymer is molten and pushed through a die with a certain cross-sectional shape, to obtain a product (extrudate) with this same cross-sectional shape. A common requirement on the extrudate is dimensional precision. However, the dimensions of the extrudate are highly influenced by a phenomenon called extrudate swell, where the extrudate starts to expand due to internal stresses in the polymer once it leaves the die [1]. Examples of products that are produced with polymer extrusion are pipes, tubes, sheets, films, structural parts and tires.

In industry, a regularly applied practice is to adjust the dimensions of the die by trial-and-error until the desired extrudate shape is obtained. Due to internal stresses the resulting extrudate to a given die shape may be difficult to predict. Therefore, especially for complex extrudate shapes, this is a time consuming and therefore costly process resulting in a desire to streamline this process in the form of a numerical optimization procedure without the need for actual experiments. Various articles have been published on die shape optimization, both using heuristic techniques, i.e. rules of thumb that limit extrudate deformation, such as die temperature and extrudate velocity through the die [2], and e.g. in a numerical environment that optimize finite element method (FEM) models while simulating extrudate behavior [3, 4].

Although the optimization methods presented here are applicable to the general die optimization problem, this thesis will relate to the works resulting from [1]. In this article, an extrudate model that is based on a FEM model is presented, and the die shape is optimized using feedback control, such that the die shape is optimized online. Furthermore, this article's introduction contains an extensive review of previous works that treat die optimization. The article presents a FEM framework for extrudates with one or more design variables that determine the die shape, i.e. inputs. In the case of just one design variable the extrusion process is modeled in two dimensions and only the height of the die is subject to optimization, see Fig. 1.1 [1]. The goal of this optimization process is to obtain an extrudate with a desired height, i.e. output. The polymer is pushed

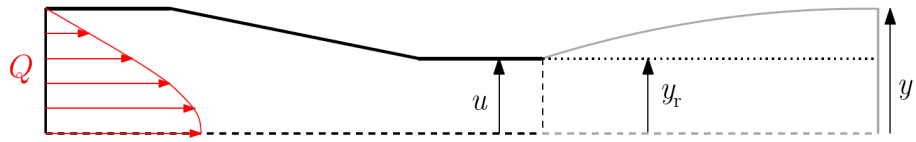


Figure 1.1: Side view of SISO extrusion process for a fixed flowrate Q with die height u and resulting extrudate height y. Due to internal stresses, the extrudate height is not constant and reaches steady state only after it has traveled a certain distance.

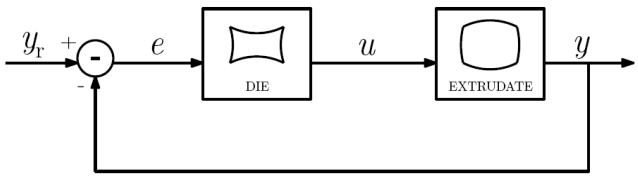


Figure 1.2: Control loop of the die shape optimization problem as implemented in [1]. The control action resembles an I-controller, such that the error is summed over time and multiplied by a gain to update the control action.

through the die, after which internal stresses make the extrudate deform until it reaches a steady state. For three dimensional extrudates, this optimization problem is extended to multiple inputs and multiple outputs. Rather than just the die height, the die corners and the die surface are also subject to optimization, see e.g. Fig. 1.3. 4-fold rotational symmetry around the desired extrudate center is assumed, as shown in the figure. In the simulations, the extrusion particles exiting the die that are in contact with an input position u_1, u_2, u_3 and u_4 (red dots in Fig. 1.3) travel in longitudinal direction. After a specified distance from the die, the polymer is assumed to be relaxed and therefore has reached a steady state shape. The particles that have traveled that distance are then measured and treated as the outputs of the system: y_1, y_2, y_3 and y_4. For both single-input single-output (SISO) and multiple-input multiple-output (MIMO) optimization problems, this transportation delay causes a significant time delay between inputs and measured outputs, a so-called dead time, resulting in conservatively tuned control parameters for the feedback controllers in [1], see Fig. 1.2.

Although [1] is able to successfully optimize the die shape to achieve the desired extrudate shape, the stability analysis is only observation based, and not based on any control theory. Multiple simulations are performed for different control parameters and their effect on stability and performance is depicted in the form of extrudate dimensions as a function of time. Moreover, although relations between inputs and outputs, i.e. die shape and extrudate shape respectively, are assumed, they are not used to form any mathematical relations, such as differential equations. Additionally, the work only explores a conservatively

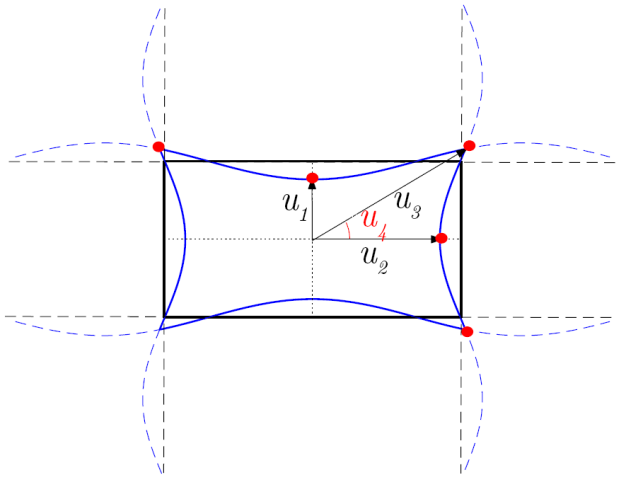


Figure 1.3: Front view of a MIMO die shape optimization process. In black, a cross section of the original die and desired extrudate shape, and in blue, the optimized die shape. The resulting optimal inputs/design variables are u_1, u_2, u_3 and u_4.

tuned I-controller as control structure, limiting rate of convergence.

In this thesis, the above mentioned shortages are addressed, meaning that the relation between inputs and outputs are explored and identified in the form of low-order approximations. This identification is then used to increase the performance compared to that of the existing solution. Model-based performance enhancements are achieved via dead-time compensators using PID control [5]. Dead-time controllers account for the transportation delay that is present in the control loop, but require knowledge of the system dynamics. More accurate knowledge of these dynamics, allows for a more aggressive controller, and thus faster convergence speed. Furthermore, several data-driven control methods are employed that, have a trade-off of decreased performance, but can guarantee stability under certain assumptions without the need for an estimate of the system. Data-driven control methods are attractive, since an accurate model may not always be available.

Although the MIMO die shape optimization problem is described in this thesis, the optimization methods are not applied to the MIMO problem. However, these methods are described in a general sense, such that they can be applied to the MIMO problem under validity of certain assumptions.

The outline of this thesis is as follows. In Chapter 2 the optimization problems are formulated. Next, in Chapter 3 the chosen optimization methods and algorithms are described, for which the implementation is then given in Chapter 4. In this chapter the simulation comparisons and results are also provided. Lastly, in Chapter 5 conclusions and a discussion for future work can be found.

Chapter 2

Problem formulation

In this chapter, the control problem for die shape optimization is introduced. First, the die shape optimization problem is described, after which the state of the art solution is briefly summarized. Finally, the design goals of this work are presented.

Die shape optimization in the environment described in [1] is considered. This environment uses partial differential equations (PDE's) to calculate the extrudate shape, such that it matches a finite element method (FEM) model. In [1], two scenario's are considered: (1) SISO die shape optimization such that only the die height is optimized, i.e. a single design variable and (2) MIMO die shape optimization such that 4 design variables are subject to optimization. In the second case, these 4 variables determine the cross section of the die. In both cases, the shape of the extrudate needs to be steered to a desired shape by adjusting the design variables. However, only SISO die shape optimization is considered in this thesis.

However, there is a significant delay between an adjustment of the design variables and the resulting effect on the extrudate shape at the measurement position. This is the result of not measuring the extrudate shape at the same location as that of the die, see Fig. 1.1. Instead, after the polymer is pushed through the die, it will be transported in the longitudinal direction, where it is subject to internal stresses, such that the shape of the extrudate changes as it is being transported. Only after a given distance, is the shape of the extrudate measured, after which the shape remains fairly constant. As will become apparent in the later chapters, this delay is the main challenge in achieving fast convergence to the optimal die shape.

The FEM is a novel and complex model to simulate polymer extrusion, with partial differential equations being solved to form the shape of the extrudate. The optimization of the die shape is not being directly integrated in this FEM model, by means of e.g. topology optimization. Rather, die shape optimization is treated as a separate problem. As such, the inputs where chosen to be positions on the die that can be changed freely, and the outputs are a position on the extrudate surface positioned at a specified distance from the die exit. The

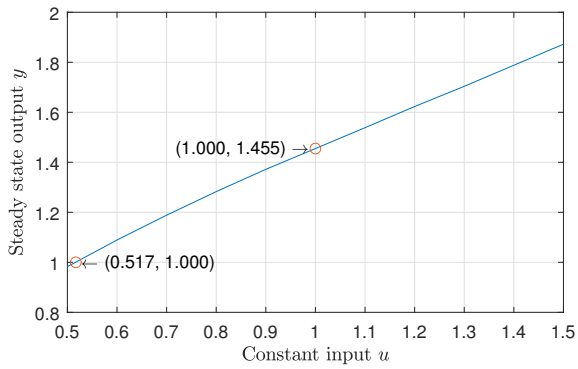


Figure 2.1: Surface of operating points for a single design variable.

FEM model is a discrete-time process with the following dynamics:

$$\mathbf{x}(k+1) = f_{\text{FEM}}(\mathbf{x}(k), \mathbf{u}(k)), \tag{2.1}$$

$$\mathbf{y}(k) = h_{\text{FEM}}(\mathbf{x}(k)) \tag{2.2}$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ is the state vector of unknown dimension, $\mathbf{u} \in \mathbb{R}^n$ is the input vector and $\mathbf{y} \in \mathbb{R}^n$ is the output vector, with n the number of inputs and outputs in the die shape optimization problem.

The dynamics of the FEM model are treated as unknown, resulting in a black box problem, where only the inputs and outputs can be controlled and measured respectively. Due to the transportation delay, this black box problem is expensive to evaluate, and certain assumptions will have to be imposed to find a control law that is able to let the system converge to the desired extrudate dimensions quickly and robustly.

The SISO die optimization problem is the simpler of the two problems. In this situation, only one input u is subject to optimization of which the effect can be measured by the single output y, see Fig. 1.1. This extrusion process is a stable process, in the way that for any bounded input, the output is also bounded. Figure 2.1 shows the surface of operating points for this SISO problem. This surface was obtained by providing multiple linearly spaced inputs to the FEM model in the interval $[0.5, 1.5]$ with steps of 0.1 for a given polymer. Obviously, this surface of operating points may differ for different polymers, but it is assumed that this surface is one-to-one for each polymer, i.e. monotonically increasing, in the sense that each input can be linked to a unique output. The polymer for which the surface is depicted, will be referred to as the *standard* polymer

Since the extrudate is transported in longitudinal direction, and the output is measured at a different location than the input, the corresponding output to a given input will be measured only after some time. This phenomenon is referred to as dead-time and is fairly common in the process industry, such as a heating system or traffic control systems, even to the extent that dead-times are

considered as an integral part of process control [5]. Dead-time compensators aim to reduce the effect of the delay. This is achieved by model based control, such that both a system estimate without delay and an estimate of the dead-time are available. Linear controllers of this class for SISO systems are well analyzed and robustness margins are available for PID-controllers. A mismatch of the estimated dead-time and the actual dead-time has a far greater influence on robustness than a mismatch of the system's estimate and the true system [5].

The numerical environment is subject to the limitation that, adjusting the die height too rapidly, results in a numerical instability associated to the way the PDE's are solved, and thereby the FEM simulation will terminate. Consequently, there is a constraint on the die height adjustment rate. Since this FEM simulation runs in discrete time, the following constraint is imposed:

$$|u(k+1) - u(k)| \leq \delta u_{\max}, \tag{2.3}$$

where $\delta u_{\max}$ is a conservatively chosen parameter or it is obtained through trial-and-error.

The control loop in the state of the art [1] is depicted in Fig. 1.2 [1]. The controller resembles an I-controller, such that the error is summed over time and multiplied by a gain to update the control action, i.e.

$$u(k+1) = u(k) + K_i e(k) \tag{2.4}$$

where K_i is the user-configured control gain for the integrator action, $e(k) = y_{\mathrm{r}} - y(k)$ and k is a FEM step. However, this control action does not necessarily take place every FEM step. Instead, the interval between control actions is a user-configured parameter. Since this interval directly relates to the rate of convergence and stability of the system, the control gain K_i should be chosen accordingly, i.e. smaller gain for shorter intervals and vice versa.

Due to the transportation delay present in the system, the gain should be smaller than the case where there would be no transportation delay. That is to say, dead-time compensators can be more efficient and robust than controllers without this characteristic [5]. Moreover, the controller in [1] is tuned without any concern for the dynamic behavior of the extrudate height. As such, the controller has been tuned conservative to prevent instability of the dynamic process, while sacrificing convergence speed.

Figure 2.2 shows the control performance of the state of the art as presented in [1]. For a different polymer, the performance may be different, and the transportation delay is also not constant, but depends on e.g. fluid velocity and die height. The interval between each control actions 1 second and the loop has a control gain $K_i = 0.05$. With these parameters, the extrudate height takes 146 seconds to reach within 1% of the desired extrudate height, i.e. 1.01. It is desired to find one or more solutions that can reduce the time needed to reach the desired extrudate height, while still being confident that the solution is robust to errors in knowledge about the system dynamics.

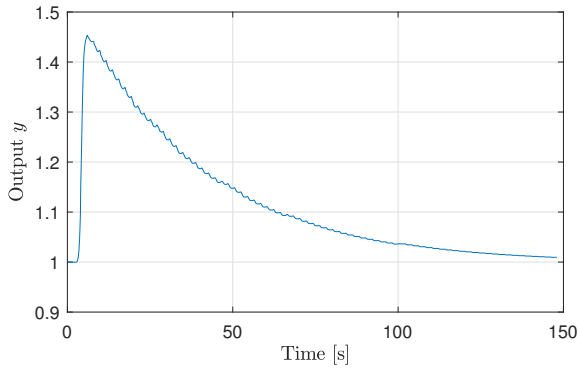


Figure 2.2: State of the art extrudate height optimization for the standard polymer, with a start height of the die $u(t_0) = 1.0$ at time $t_0 = 0$ and desired extrudate height $y_r = 1.0$.

Chapter 3

Optimization

In this chapter, several methods are presented to solve the die shape optimization problem, thereby resulting in the cross section of the die that achieves the desired extrudate shape. These methods are divided into two main approaches, namely data-driven control and model based control. Furthermore, Subsection 3.1.2 discusses a hybrid between data-driven control and model based control. The data-driven optimization methods are described for general optimization problems, to aid in extension to MIMO die shape optimization. For the model based control method, the extension is straightforward, and not described here.

3.1 Data-driven optimization

Data-driven optimization is a model-free approach to minimize an objective function with just experimental data. As such, it has applications in situations where a model is not available, i.e. black box such as die optimization problem in [1]. Still, data-driven approaches usually rely on assumptions such as convexity and continuity to guarantee stability of the convergence.

In this section, the control problem is translated from a black box problem to a mathematical problem that can be solved using data-driven control methods. As such, some mathematical concepts are presented, which will later be linked to the die optimization problem, and thus providing a method to achieve the desired extrudate dimensions. This is presented in the following way. At first, assumptions about the objective function are given and two data-driven methods are presented to solve the die shape optimization problem. Then, the first method, extremum seeking control, estimates the derivatives of an objective function to locate its optimizer, i.e. the design variables that result in the optimal die shape. The second method tries to create an estimate of the objective function that is to be optimized and subsequently find the argument that optimizes the objective function estimate.

3.1.1 The objective function

For the data-driven optimization methods that are utilized to solve the die optimization problem, the goal will be to find the minimizer of an objective function. As such, the optimization problem needs to be translated into this structure. That process is described below, along with some assumptions on the behavior of the FEM model such that convergence can be guaranteed.

Consider the optimization problem

$$\min_{\mathbf{u}} f(\mathbf{u}), \tag{3.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is the objective function to be minimized, and $\mathbf{u} \in \mathbb{R}^n$ are the design variables. In the context of die shape optimization, the objective function f is a function that maps the design variables to a scalar, with the aim that a minimizer of f coincides with the desired extrudate dimensions. To aid this process, f is a composite function such that

$$f(\mathbf{u}) = g \circ h(\mathbf{u}), \tag{3.2}$$

where $g : \mathbb{R}^n \to \mathbb{R}$, and $h : \mathbb{R}^n \to \mathbb{R}^n$ is a function that maps the design variables to the measured outputs $\mathbf{y} \in \mathbb{R}^n$, i.e. measured steady state extrudate shape. Note that this is also the surface of operating points, see e.g. Fig. 2.1. To this end, the following is assumed.

Assumption 3.1.1. *For each input $\mathbf{u}_{eq}$ there exists an equilibrium state $\mathbf{x}_{eq}$ such that $x(k+1) = f_{FEM}(\mathbf{x}_{eq}, \mathbf{u}_{eq}) = x(k)$. Moreover, for each $\mathbf{u}_{eq}$, the equilibrium $\mathbf{x}_{eq}$ is asymptotically stable. As such, each $\mathbf{u}_{eq}$ can be directly linked to a system output that corresponds to the equilibrium via the function h, such that $\mathbf{y}_{eq} = h(\mathbf{u}_{eq})$.*

The function g is a user-defined function, e.g. the quadratic cost function

$$g(\mathbf{y}) = (\mathbf{y} - \mathbf{y}_{\mathrm{r}})^\top (\mathbf{y} - \mathbf{y}_{\mathrm{r}}), \tag{3.3}$$

with $\mathbf{y}_{\mathrm{r}} \in \mathbb{R}^n$ the desired, or reference, measured outputs.

The goal of this data-driven optimization approach is to let the design variables $\mathbf{u}$ converge to the minimizer $\mathbf{u}^*$ of (3.1), i.e.

$$\mathbf{u} \to \mathbf{u}^* = \arg\min_{\mathbf{u}} f(\mathbf{u}). \tag{3.4}$$

Proper selection of g may result in $\mathbf{u}^*$ being unique, such that

$$f(\mathbf{u}) > f(\mathbf{u}^*), \ \forall \mathbf{u} \neq \mathbf{u}^*. \tag{3.5}$$

In the case of the SISO optimization problem where $n = 1$, this can be shown by imposing the following assumption.

Assumption 3.1.2. *In the case of a single design variable, i.e. $n = 1$, the function h is concave.*

Note that $\mathbf{y}$ is a measured function. Consequently, defining functions

$$h_{-}(\mathbf{u}) := -h(\mathbf{u}), \text{ and } g(h_{-}(\mathbf{u})) := (h_{-}(\mathbf{u}) + \mathbf{y}_{\mathrm{r}})^{\top}(h_{-}(\mathbf{u}) + \mathbf{y}_{\mathrm{r}}), \tag{3.6}$$

where h_{-} is convex and therefore g is strictly convex. Next, it is shown that, for $n = 1$, the composition of these functions is also strictly convex, i.e. that for all $x, y \in \Omega$, with $x \neq y$, $(g \circ h)(\lambda x + (1-\lambda)y) < \lambda(g \circ h)(x) + (1-\lambda)(g \circ h)(y)$, where $\Omega = \mathbb{R}$ is the domain of both g and h and $\lambda \in (0, 1)$.

$$\begin{aligned}
(g \circ h)(\lambda x + (1-\lambda)y) &= g(h(\lambda x + (1-\lambda)y)) && (3.7)\\
&\leq g(\lambda h(x) + (1-\lambda)h(y)) && (3.8)\\
&< \lambda g(h(x)) + (1-\lambda)g(h(y)) && (3.9)\\
&= \lambda(g \circ h)(x) + (1-\lambda)(g \circ h)(y). && (3.10)
\end{aligned}$$

With $f = g \circ h$ being strictly convex, it has a global and unique minimizer at the argument that corresponds to the desired die height. This is an important property since extremum seeking, presented in the next section, aims to locate that minimizer.

3.1.2 Extremum seeking control

Extremum seeking (ES) control is an optimization method that uses derivative information to let the input converge to the objective function's optimizer. Typically, the following three elements are present in the control loop: (1) The objective function, possibly containing a dynamic process and measurement disturbance, (2) derivative estimation of the objective function, and (3) the optimizer, see Fig. 3.1. Although not always the case, e.g. [6, 7], most often sinusoidal perturbation is used to collect the local derivative information. In the analysis of most ES frameworks, the timescale separation principle is used to show stability. Three timescales are considered ordered from slowest to fastest: (1) the objective function, possibly containing a dynamic process, (2) derivative estimation, and (3) the optimizer. This means that convergence speed is limited by the process' phase at the frequency of the perturbation signal. Although implementations vary, the proofs of stability and convergence in many schemes are based on the tuning of certain parameters, typically leading to slow convergence. However, using such implementations eliminates the need for models of the system.

In [8] a classical extremum seeking scheme for single variable optimization with local stability properties is presented. This scheme uses a low-pass and high-pass filter to extract local gradient information of the objective function. However, this scheme is unable to asymptotically converge to the exact optimizer [9]. Various variations of this scheme have appeared, e.g. [10] [11] [12] [9], see Appendix A, some of which can handle several variables, as opposed to a single variable. Various variations of this scheme have appeared. For instance, [10] presents a framework where that results in convergence to the true optimizer as time goes to infinity. In [13] non-local stability properties of [8] are explored.

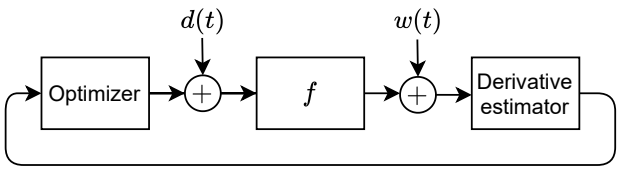


Figure 3.1: Typical scheme of an extremum seeking loop.

Recently, several ES implementations have appeared in literature that converge in finite time [14, 15]. Extensions of [8] utilize a bank of filters to obtain more accurate gradient information , e.g. low-pass filters [11] and moving average (MA) filters [12] [9], some of which can handle several variables, as opposed to a single variable.

The framework presented in [12] is summarized here. To ensure convergence of the optimization process, Assumption 3.1.1 is required.

Assumption 3.1.3. *The mapping $f : \mathbb{R}^n \to \mathbb{R}$ is $N+1$, $N \geq 2$ times continuously differentiable with respect to the control input $\mathbf{u} \in \mathbb{R}^n$ and there exist a constant input $\mathbf{u}^* \in \mathbb{R}^n$ such that the gradient equals*

$$\tilde{g}(\mathbf{u}) := \left[\frac{\partial f(\mathbf{u})}{\partial u_1}, \frac{\partial f(\mathbf{u})}{\partial u_2}, ..., \frac{\partial f(\mathbf{u})}{\partial u_n}\right]^\top = \mathbf{0}, \tag{3.11}$$

if and only if $\mathbf{u} = \mathbf{u}^*$. Here, $N \in \mathbb{N}_{\geq 2}$ is a user-defined parameter.

Definition 3.1.1. *Let f be a function of many variables defined on the set S. For any $a \in \mathbb{R}$, the set*

$$P_a = \{\mathbf{u} \in S \mid f(\mathbf{u}) \leq a\} \tag{3.12}$$

is called the lower level set of f for a.

Definition 3.1.2. *The function f of many variables defined on a convex set S is quasiconvex if every lower level set of f is convex, i.e. P_a is convex for every a.*

Assumption 3.1.4. *The objective function $f : \mathbb{R}^n \to \mathbb{R}$ is quasiconvex.*

The requirement of Assumption 3.1.3 and Assumption 3.1.4 ensures that $\mathbf{u}^*$ is the unique minimizer of f.

Derivative estimation framework

This subsection presents a framework to estimate the derivatives of the objective function f. With the above mentioned assumptions the objective function f is quasiconvex and maps the design variables $\mathbf{u}$ to a scalar, with $\mathbf{u}^*$ being the unique minimizer of $f(\mathbf{u})$. This map $f = g \circ h$ is a composite of (1) the surface of operating points h, see e.g. Fig. 2.1 and (2) a user-defined cost function. In essence

$$f : \mathbf{u} \to \mathbb{R}, \mathbf{u} = [u_1, u_2, ..., u_n]^\top \in \mathbb{R}^n \tag{3.13}$$

at the constant input $\hat{\mathbf{u}}$. The (higher-order) derivatives of f are denoted by:

$$D_{u_k}^{\alpha_k} := \left.\frac{\partial^{\alpha_k} f}{\partial u_k^{\alpha_k}}\right|_{\hat{\mathbf{u}}}. \tag{3.14}$$

To clarify this notation, e.g. the third-order derivative with respect to input u_2 is denoted as $D_{u_2}^3$.

Sinusoidal perturbation, also called dithering, is assumed

$$\mathbf{d}(t) = [d_1(t), d_2(t), ..., d_n(t)]^\top \in \mathbb{R}^n, \tag{3.15}$$

where $d_k(t) = a_k \cos(\omega_{d_k} t)$ and $a_k, \omega_{d_k} \in \mathbb{R}_{>0}, k = 1, 2, ..., n$ are the dither amplitude and frequency respectively. This yields the inputs $\mathbf{u}(t) = \hat{\mathbf{u}} + \mathbf{d}(t)$. This extremum seeking framework assumes timescale separation, meaning that the dither frequencies should be chosen small enough to have a negligible phase lag between $h(t)$ and the dither signal $d_k(t)$ for all k. Under Assumption 3.1.3, Taylor's theorem states that $f(\mathbf{u}(t))$ can be expressed as

$$f(\mathbf{u}(t)) = f(\hat{\mathbf{u}}) + \sum_{r=1}^{N} \frac{1}{r!} \left(\sum_{k=1}^{n} D_{u_k}^1 d_k(t) \right)^r + R_N, \tag{3.16}$$

with R_N the remainder term, which is a function of the derivatives of order $> N$. Now, the following expression is introduced

$$f(\mathbf{u}(t)) = \mathbf{p}(t) A \mathbf{g}_{\hat{\mathbf{u}}} + R_N \tag{3.17}$$

to describe (3.16), with $\mathbf{p}(t) \in \mathbb{R}^{1 \times n_g}$ a function of $\mathbf{d}(t)$, A a constant diagonal matrix, and $g_{\hat{\mathbf{u}}} \in \mathbb{R}^{n_g \times 1}$ the derivative vector, which contains the map output and derivatives at $\mathbf{u} = \hat{\mathbf{u}}$, up to order N.

Example 3.1.1. *For $n = N = 2$, it holds that $n_g = 6$, and one can select*

$$\mathbf{p}^\top(t) = \begin{bmatrix} 1 \\ \cos(\omega_{d_1} t) \\ \cos(\omega_{d_2} t) \\ \cos^2(\omega_{d_1} t) \\ \cos(\omega_{d_1} t)\cos(\omega_{d_2} t) \\ \cos^2(\omega_{d_2} t) \end{bmatrix}, \quad \mathbf{g}_{\hat{u}} = \begin{bmatrix} f(\hat{u}) \\ D_{u_1}^1 \\ D_{u_2}^1 \\ D_{u_1}^2 \\ D_{u_1}^1 D_{u_2}^1 \\ D_{u_2}^2 \end{bmatrix},$$

$$A = \mathrm{diag}(1, a_1, a_2, {}^1\!/\!_2 a_1^2, a_1 a_2, {}^1\!/\!_2 a_2^2), \tag{3.18}$$

to let (3.17) *correspond with* (3.16) *for* $\mathbf{d}(t)$.

To obtain an estimate $\tilde{\mathbf{g}}_{\hat{\mathbf{u}}}$ of $\mathbf{g}_{\hat{\mathbf{u}}}$, the objective function is multiplied with the demodulation signal $\mathbf{m}(t)$, such that

$$\mathbf{m}(t) f(\mathbf{u}(t)) = \mathbf{m}(t)\mathbf{p}(t) A \mathbf{g}_{\hat{\mathbf{u}}} + \mathbf{m}(t) R_N, \tag{3.19}$$

with $\mathbf{m}(t)$ selected as

$$\mathbf{m}(t) = \begin{bmatrix} \cos(\omega_{m_1} t) \\ \cos(\omega_{m_2} t) \\ \vdots \\ \cos(\omega_{m_{n_g}} t) \end{bmatrix}, \mathbf{m}(t) \in \mathbb{R}^{n_g} \tag{3.20}$$

with $\omega_{m_j} \in \mathbb{R}_{>0}$, $j = 1, 2, ..., n_g$ related to the dither frequencies w_{d_k}, $k = 1, 2, ..., n$. Under the assumption that R_N is small, it follows from (3.19) that the derivative estimate $\tilde{\mathbf{g}}_{\hat{\mathbf{u}}}$ could be obtained by pre-multiplying $\mathbf{m}(t)f(\mathbf{u}(t))$ with the inverse of matrix $\mathbf{m}(t)\mathbf{p}(t)A$. However, the matrix $\mathbf{m}(t)\mathbf{p}(t)$ is singular by construction. To cope with this problem, consider the integral of (3.19) over a finite time period T

$$\int_{t-T}^{t} \mathbf{m}(\tau) f(\mathbf{u}(\tau))\, d\tau - \int_{t-T}^{t} \mathbf{m}(\tau) R_N\, d\tau = \underbrace{\int_{t-T}^{t} \mathbf{m}(\tau)\mathbf{p}(\tau)\, d\tau}_{K} A\mathbf{g}_{\hat{\mathbf{u}}}. \tag{3.21}$$

When T is equal to the smallest common period time of all harmonic signals in the matrix $\mathbf{m}(t)\mathbf{p}(t)$, the matrix K is constant.

Example 3.1.2. *Take $n = N = 1$ and $\mathbf{p}(t) = \begin{bmatrix} 1 & \cos(\omega_{d_1} t) \end{bmatrix}$ accordingly. With $\mathbf{m}(t) = \begin{bmatrix} 1 & \cos(\omega_{m_1} t) \end{bmatrix}^\top$, matrix $\mathbf{m}(t)\mathbf{p}(t)$ is singular by construction. Matrix K in (3.21) is*

$$K = \int_{t-T}^{t} \begin{bmatrix} 1 & \cos(\omega_{d_1}\tau) \\ \cos(\omega_{m_1}\tau) & \cos(\omega_{m_1}\tau)\cos(\omega_{d_1}\tau) \end{bmatrix} d\tau. \tag{3.22}$$

For $\omega_{m_1} = \omega_{d_1} = 2\pi$ and $T = {}^{2\pi}/_{\omega_{d_1}}$

$$K = \begin{bmatrix} 1 & 0 \\ 0 & {}^1/_2 \end{bmatrix} \tag{3.23}$$

which is full rank.

When K is full rank and constant, and assuming estimation error $\int_{t-T}^{t} \mathbf{m}(\tau) R_N\, d\tau$ to be small, the estimate $\tilde{\mathbf{g}}_{\hat{\mathbf{u}}}$ is obtained according to the derivative estimation framework derived from (3.21):

$$\tilde{\mathbf{g}}_{\hat{\mathbf{u}}}(t) = G_0 \int_{t-T}^{t} \mathbf{m}(\tau) f(\mathbf{u}(\tau))\, d\tau, \quad G_0 = A^{-1}K^{-1}, \tag{3.24}$$

with $G_0 \in \mathbb{R}^{n_g \times n_g}$ a constant matrix.

Converging to the optimizer of f

$$\dot{\hat{\mathbf{u}}} = -K_{DE}\tilde{\mathbf{g}}_{\hat{\mathbf{u}}}, \quad \hat{\mathbf{u}}(t_0) = \hat{\mathbf{u}}_0, \quad t_0 = 0, \tag{3.25}$$

where K_{DE} is a matrix with the individual elements positive if an only if that element corresponds to a first derivative. All other elements are zero.

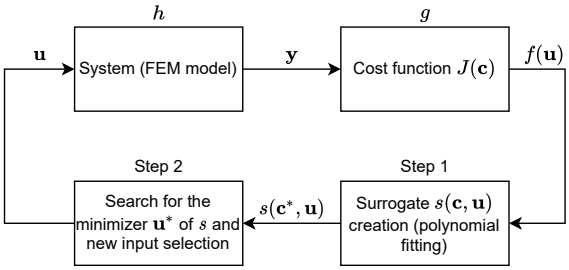


Figure 3.2: Schematic overview of the surrogate modelling algorithm. This loop resembles one iteration of the loop, where the input $\mathbf{u}$ remains constant until the system reaches steady-state after the transportation delay. Only then is the output $f(\mathbf{u})$ measured.

Example 3.1.3. *For* $\mathbf{g}_{\hat{\mathbf{u}}}$ *in* (3.18) *in Example 3.1.1, the matrix*

$$K_{DE} = \begin{bmatrix} 0 & k_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & k_2 & 0 & 0 & 0 \end{bmatrix}, \tag{3.26}$$

where $k_1, k_2 > 0$.

3.1.3 Surrogate modeling

Surrogate modeling is the term for algorithms where the outcome of a system is estimated with a so called surrogate model, e.g., based on polynomial or cubic splines, using very little evaluations of the objective function, typically without derivative information. This method is useful when the system output is not subject to noise, deterministic, and the system is a black box model that is expensive to evaluate, e.g., in the economical sense, time duration [16]. Moreover, algorithms that achieve global optimization exist, e.g. univariate [17] and multivariate [18]. Generally, surrogate modeling algorithms consist of two phases: (1) surrogate construction, and (2) searching for a minimum of the surrogate. The method summarized below is a relatively straightforward algorithm that (1) uses polynomial regression to estimate the objective function and create the surrogate. Then (2) the optimizer of the surrogate model is located such that it estimates the optimizer of the real objective function. This routine is depicted schematically in Fig. 3.2.

The first step is to construct the surrogate model s, that approximates the objective function f. Consider the polynomial surrogate model, a so-called response surface model

$$s(\mathbf{c}, \mathbf{u}) := \mathbf{c}^\top \mathbf{s}(\mathbf{u}) \tag{3.27}$$

where $\mathbf{c} = [c_0, c_1, ..., c_{K-1}]^\top \in \mathbb{R}^K$ is a coefficient vector, $s_k : \mathbb{R}^n \to \mathbb{R}$ is a monomial and $\mathbf{s}(\mathbf{u}) = [s_0(\mathbf{u}), s_1(\mathbf{u}), ..., s_{K-1}(\mathbf{u})]^\top \in \mathbb{R}^K$, with $K \in \mathbb{N}$. The structures of monomials used are a design choice. The goal is to find the coef-

Table 3.1: Number of output measurements for the number of design variables for a central composite design. With the exception for a single design variable, the number of required output measurements equals $2^n + 2n + 1 \leq M$.

# Variables n	# Measurements
1	3
2	9
3	15
4	25

ficient vector $\mathbf{c}$ such that the surrogate model s is the best estimate of the real objective function f. Therefore, again, assume that Assumption 3.1.1 holds.

The best estimate is obtained by minimizing the linear least squares residuals of the surrogate model to the objective function, i.e. a linear least squares fit. Define the cost function

$$J(\mathbf{c}) := \sum_{m=j}^{i} \left[f(\mathbf{u}_m) - s(\mathbf{c}, \mathbf{u}_m)\right]^2, \tag{3.28}$$

where $m \in 0, 1, ..., i$ is an evaluation of the objective function, $i \in \{0, 1, ..., N-1\}$ is the current objective function evaluation, N the total number of objective function evaluations and $j = \max\{0, i - M + 1\}$, with M the desired number of objective function evaluations to contribute to the fit. Note that the objective function is evaluated by giving an input $\mathbf{u}_i$ to the system and waiting for the output $f(\mathbf{u}_i)$ to converge to a steady state value, taking the transport delay into account. For this cost function J, it is required that $N \geq M \geq K$. From this requirement it follows that J should only be evaluated when $i - j + 1 \geq M$. To find the parameter vector $\mathbf{c}^*$ that minimizes J, the following should hold.

$$\frac{\partial J(\mathbf{c})}{\partial c_k} = 0, \ \forall k = 0, 1, ..., K - 1 \tag{3.29}$$

Define the following matrix:

$$S := [\mathbf{s}(\mathbf{u}_j), \mathbf{s}(\mathbf{u}_{j+1}), ..., \mathbf{s}(\mathbf{u}_i)]^\top, \tag{3.30}$$

where again $j = \max(\{0, i + 1 - M\})$, and $M \in \mathbb{N}_{\geq K}$ is the chosen number of function evaluations that are used to create the fit. Then, if S has rank larger than or equal to $K - 1$, the following unique coefficients will minimize the cost function.

$$\mathbf{c}^* = \arg\min_{\mathbf{c}} J(\mathbf{c}) = (S^\top S)^{-1} S^\top \mathbf{f}_{j:i}, \tag{3.31}$$

where $\mathbf{f} = [f(\mathbf{u}_0), f(\mathbf{u}_1), ..., f(\mathbf{u}_{l-1})]^\top$. Note that this is only possible after at least $K - 1$ evaluations of the objective function. As a result, the surrogate model $s(\mathbf{c}, \mathbf{u})$ evaluated at the point $(\mathbf{c}^*, \mathbf{u}_i)$ is the surrogate's best estimate of $f(\mathbf{u}_i)$.

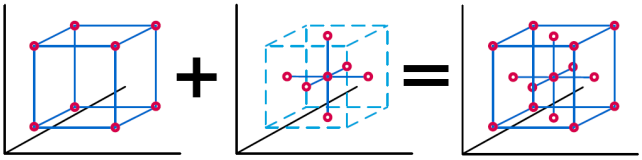


Figure 3.3: Example of a 3D central composite design, captured from [16].

Subsequently, the minimizer of the surrogate $s(\mathbf{c}^*, \mathbf{u})$ will be the best estimate of the minimizer of $f(\mathbf{u})$ and step 2 of the algorithm can commence, namely searching for the minimizer of the surrogate. The choice of the optimization algorithm should depend on the optimization problem. In the case of surrogate modeling, it is attractive that Assumptions 3.1.3 and 3.1.4 also hold, since this allows for the use of local optimization techniques, which are generally faster than algorithms that look for a global optimum, especially in case of multiple design variables. For now, it is assumed that both assumptions hold.

Depending on the number of objective function evaluations, the algorithm may jump back to step 1 using a new input. This input is chosen as the minimizer of $s(\mathbf{c}^*, \mathbf{u})$ obtained in step 2. Possibly, a disturbance can be added to this new input to avoid singularities. Next, revisiting step 1, the process is repeated until objective function evaluation $l - 1$, or an optimality criterion is reached.

Example 3.1.4. *Assume one design variable, i.e. SISO optimization, and a quadratic surrogate model $s(\mathbf{c}, u)$, then the following form for s could be chosen: $\mathbf{c} = [c_0, c_1, c_2]^\top$ and $\mathbf{s}(u) = [s_0(u), s_1(u), s_2(u)]^\top = [u^2, u, 1]$ This means that $K = 3$. When using a central composite design, at least 3 measurements are required, see Fig. 3.3 and Table 3.1. Let's say that the minimizing argument u^* for f is expected to lie on the interval $[a, b]$, such that $a < u^* < b$, then a central composite design would yield the following three initial inputs: $u_0 = a, u_1 = \frac{a+b}{2}, u_2 = b$. It is desired that, directly after evaluating the objective function $f(u)$ to these inputs, the argument $\mathbf{c}^*$ that uniquely minimizes $J(\mathbf{c})$ is found, leading to $M = 3$. For this example, the initial guess of $\mathbf{c}^*$ is satisfactory, so $N = 3$. Then, using $\mathbf{c}^*$, the minimizing argument u^* for $s(\mathbf{c}^*, u)$ can be found using a suitable optimization method. For this case, with s quadratic, the complete algorithm is presented in Algorithm 1 (page 26).*

Remark 3.1.1. *From Table 3.1 it follows that the number of objective function evaluations needed becomes significantly larger when the number of dimensions increases, making this method less suitable for the MIMO die shape optimization.*

Remark 3.1.2. *One could alter this algorithm, by only trying to get an estimate of the function h instead of the function $f = g \circ h$, but then still locating the minimum of the estimate of f. This is advantageous, since a polynomial fit of h will, generally, be more accurate than a polynomial fit of the composition f. However, by experiments, it was found that for this application the advantage is marginal.*

Algorithm 1 Optimization of the die shape using surrogate modeling

```
U ← central composite design with 2^n + 2n + 1 rows and n columns, where
n is the number of design variables. Each row of this matrix contains the
corresponding input vector, i.e. U_i = u_i^⊤
f ← 0_N
for i ← 0 to N − 1 do                                   ▷ Start of step 1.
    u_i ← U_{i,:}
    f_i ← f(u_i) Evaluate the objective function.
    if i ≥ M − 1 then
        j ← max({0, i + 1 − M})
        S ← [0]_{N×K}
        for m ← j to i do
            for k ← 0 to K − 1 do
                S_{m,k} ← s_k(U_{m,:})
            end for
        end for
        c* ← (S^⊤ S)^{−1} S^⊤ f          ▷ Find the optimal coefficients for s(c, u).
        u* ← arg min_u s(c*, u)                         ▷ Start of step 2.
        if i ≥ 2^n + 2n + 1 then
            u_{i+1} ← u* + small random value
            U_{i+1,:} ← u_{i+1}^⊤
            if Optimality criterion met then
                break
            end if
        end if                                          ▷ End of step 2.
    end if
end for
                                                        ▷ End of step 1.
```

3.2 Model based control

As opposed to data-driven optimization, in model based control, a suitable controller is designed using an estimate of the system's behavior. An estimate of the system can be procured in various ways, e.g. first principles design or system identification techniques. Once the system estimate is available, this allows for a design process in which the system's behavior can be simulated to obtain the desired system robustness and performance, without the need for the real system.

In control theory, there is a vast selection of model based control techniques, both linear and nonlinear. Examples of linear control are Linear Quadratic Regulators (LQR) using state feedback [19], root locus [20], model predictive control [21] and design in the frequency domain [20]. Examples of nonlinear model based control are (1) feedback linearization and (2) feedback control using a linear model of the system [22].

Another type of a linear model based control technique is called the dead-time compensator. These techniques are suitable for processes exhibiting transportation delays. Typically, dead-time compensators try to steer a copy of the system to the desired set-point. The dynamics of this copied system are then split into (1) the non-delayed dynamics and (2) the delay itself. Moreover, the actual system with dead-time receives the exact same input. In this way, with a correct system estimate, both the true system and the copied system will be steered towards the reference faster compared to regular feedback control.

In this section, first the process of acquiring an estimate of the system is explained. Then the use for a type of dead-time compensator, a so-called Smith Predictor, is motivated. Lastly, an extension to the extremum seeking controller is presented, that uses an estimate of the systems dead time to shorten one of the timescales of the loop.

3.2.1 Acquiring a system estimate

It is assumed that the nonlinear dynamical FEM system can be represented in the form of a Wiener model structure, such that a (discrete-time) linear system is followed by a static nonlinearity placed in series, i.e. a transformation from $\mathbf{x}(k+1) = f_{\text{FEM}}(\mathbf{x}(k), \mathbf{u}(k))$ to

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\boldsymbol{\Delta}\mathbf{u}(k) \tag{3.32}$$

$$\boldsymbol{\Delta}\mathbf{y}(k) = C\mathbf{x}(k) + D\boldsymbol{\Delta}\mathbf{u}(k) \tag{3.33}$$

$$\mathbf{y}(k) = h_w(\mathbf{y_l}(k)), \tag{3.34}$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ is the state vector of the linearized model with transfer function G, $\boldsymbol{\Delta}\mathbf{u} \in \mathbb{R}^n$ is the input vector, $\boldsymbol{\Delta}\mathbf{y} \in \mathbb{R}^n$ and is the output vector, with n the number of inputs and outputs of the die shape optimization problem. Also, $\boldsymbol{\Delta}\mathbf{u}(k) = \mathbf{u}(k) - \mathbf{u}_{\text{eq}}$ and $\mathbf{y_l}(k) = \boldsymbol{\Delta}\mathbf{y}(k) + \mathbf{y}_{\text{eq}}$ and $h_w : \mathbb{R}^n \to \mathbb{R}^n$ is the static nonlinearity. This complete Wiener is depicted in Fig. 3.4. To estimate a model

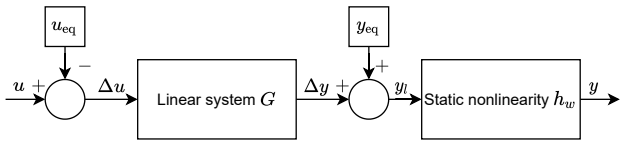


Figure 3.4: Structure of a Wiener system. A linear model is composed with a static nonlinearity.

of the true system, both the dynamic behavior needs to be linearized around an operating point and the static nonlinearity need to be estimated.

In case of just a single design variable, and the static nonlinearity being a on-to-one function, the following identification procedure will identify the discrete-time Wiener system.

1. Choose a linearization point. Figure 2.1 contains a map with a surface of operating points for the SISO case. The selected operating point yields the parameters u_{eq} and y_{eq}, see Fig. 3.4.

2. A model set $\mathcal{M}$ for the discrete-time linear model $G(z, \theta)$ has to be selected. No noise and disturbances are used in the FEM system, and any disturbances are regarded as nonlinear, and will therefore not be modeled. This leaves two viable options for the model structures: (1) Finite Impulse Response (FIR), and (2) Output Error (OE). The model set has the following structure:

$$\mathcal{M} = \{(G(z, \theta), H(z, \theta)) \mid \theta \in \Theta \subset \mathbb{R}^d\}, \tag{3.35}$$

   where θ is the coefficient vector of the polynomial models G and H. Since, as just mentioned, the errors are not modeled for a FIR and OE polynomial model, the error dynamics are selected as 1, i.e. $H(z, \theta) = 1$.

3. One has to design an input signal to feed to the true FEM system. Due to the nonlinear nature of the FEM system, and also the transportation delay, a *slow* multisine signal is desired. This will minimize any averaging effect of the extrudate height caused by the transportation delay. Moreover, (2.3) mentions a maximum change in input signal value in between samples. The Matlab function `idinput` has an input parameter that allows the user to specify this maximum change in signal value. It is required that the multisine contains at least half the number of frequencies of the number of parameters to be estimated in θ [23]. In essence, if $f_{\text{freq}} \in \mathbb{R}^z_{>0}$ were a vector of frequencies of the multisine, with every element of f_{freq} unique, then for d in (3.35) it is required that $d \leq 2z$. This is caused by the criterium for persistence of excitation.

4. The selected operating input point is then added to the generated input signal: if $\Delta u(k)$ is the generated multisine, then the input to the FEM system would be $u(k) = \Delta u(k) + u_{\text{eq}}$. When subjecting the system to

this input signal $u(k)$, the output of the system $y(k)$ is captured. For this signal, the operating point x_{eq} is subtracted from the output signal $y(k)$, such that $\Delta y(k) = y(k) - y_{\text{eq}}$.

5. Now that this data Δu and Δy is available, the model $G(z, \theta)$ can be identified using an appropriate algorithm, e.g. using the System Identification Toolbox from Matlab. For convenience of writing, from now on, the model $G(z, \theta)$ is denoted by G.

6. The last step in the identification process is the estimation of the static nonlinearity h_w. As discussed in Chapter 2, the surface of operating points depicted in Fig. 2.1 was obtained by giving the constant inputs u in the sequence $0.5, 0.6, ..., 1.4, 1.5$ to the FEM system and measuring the steady state output y for each of those inputs. As such, using these 11 input-output pairings, it is possible to fit a polynomial with a maximum order of 10 through these points. However, it is not the relation from u to y that is of interest, but the relation from y_l to y to form the static nonlinearity h, see again Fig. 3.4. In order to obtain the values for y_l, one could simulate the output of G to the inputs $\Delta u = u - u_{\text{eq}}$, with the sequence of u given above, and then measuring $y_l = \Delta y + y_{\text{eq}}$. Another approach, one that does not require any simulations, would be to use following, related to the final value theorem.

   If it is desired to calculate the steady state gain of a discrete time transfer function to a constant input signal, this is essentially measuring the output of the system to a step function when time goes to infinity. Take the discrete-time step signal $u(k)$ defined as follows:

$$u(k) = \begin{cases} a, \text{ if } k \geq 0. \\ 0, \text{ else.} \end{cases} \tag{3.36}$$

   This signal has the z-transform

$$U(z) = \sum_{k=0}^{\infty} u(k) z^{-k} = a \sum_{k=0}^{\infty} z^{-k} = \frac{a}{1 - z^{-1}}. \tag{3.37}$$

   Now taking the z-transform of the output $y(k)$:

$$Y(z) = U(z)G(z) = \frac{aG(z)}{1 - z^{-1}}. \tag{3.38}$$

   Then applying the final value theorem to the output $y(k)$ gives

$$\lim_{k \to \infty} y(k) = \lim_{z \to 1} (1 - z^{-1}) Y(z) = \lim_{z \to 1} aG(z). \tag{3.39}$$

Example 3.2.1. *Assume the discrete-time transfer function of a first-order low-pass filter.*

$$G(z) = \frac{0.00995 z^{-1}}{1 - 0.99005 z^{-1}}. \tag{3.40}$$

Applying a step signal with gain a and using (3.39) *gives that the steady state value of* $\frac{0.00995a}{1-0.99005} = a$*, which is expected from a low-pass filter.*

In essence, since the inputs $\Delta u = u - u_{\text{eq}}$ are known from the above defined sequence, the outputs Δy can be directly computed using the final value theorem. In turn, $y_l = \Delta y + y_{\text{eq}}$ is also known. Note that in the above example $a = \Delta u$.

Now that this information is available, the input-output pairings y_l and y can be used to estimate the static nonlinearity h, using e.g. least squares fit to obtain h in the form of a polynomial. Although least squares polynomial fits are fairly straightforward, Matlab provides the function `polyfit` to accomplish this. Note again that it is assumed that h is strictly increasing in its argument.

3.2.2 Dead-time compensation using a Smith Predictor

Although the previously presented solutions have the advantage that they are data-driven, and thus robust to uncertainties, they can be slow. Where Surrogate Modeling is only slow for the MIMO case, Extremum Seeking will be slow for both the SISO and the MIMO case. Consequently, there is a need for a robust method that will have comparable speed for both the SISO and the MIMO case, that is robust to model uncertainties as well. As a reminder, the speed limitation for both data-driven optimization methods is mainly caused by the transportation delay. As it turns out, transportation delays are fairly common in process control, and is said to even be an integral part of this subject [5]. As was the case in the state of the art die shape optimization [1], controllers that steer systems exhibiting dead-times to a desired reference need to be tuned conservatively, to allow for smooth convergence and prevent instability.

Controllers that attempt to overcome this limitation are so-called Dead-time compensators (DTC), where dead-time refers to transportation delay. One such DTC is a linear controller called the Smith Predictor. The concept of a Smith Predictor is fairly straightforward. Instead of only steering the actual system to a desired reference, two other systems are present in the control loop as well: (1) a copy of the model where the transportation delay is removed G, and (2) a model of the plant Ge^{-Ls}, where e^{-Ls} is the delay and L is the delay in seconds, see Fig. 3.5. Additionally, a low-pass filter F_{LP} can be added to improve robustness, making this control loop a so-called Filtered Smith Predictor. Intuitively, the control loop of the Smith Predictor can be interpreted as follows. A reference y_r is selected by the user, after which the controller, which may be tuned based on the model G, instead of the system P, will attempt to steer the model G to the reference. Since G is the only of the three systems that is not subject to a transportation delay, it can be expected to react the quickest. Essentially, it is the aim of the Smith Predictor to steer the non-delayed model G to the reference, after which the other two models will follow. If the model is correct, the error e_p, which is the difference in the measured output of the actual system P and the model of the system Ge^{-Ls} will always be zero. If the model

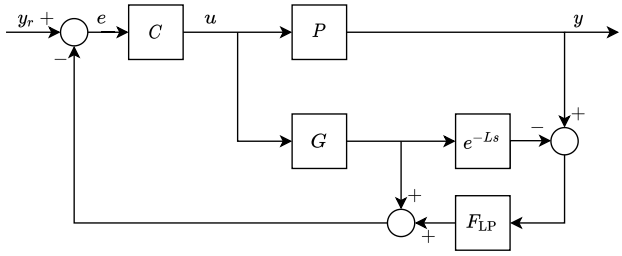


Figure 3.5: SISO control loop of a type of dead-time compensators called the Filtered Smith Predictor.

is incorrect however, this error will not be zero and is added to the error e. When one has to choose whether DTC's are a viable option to control a process containing a transportation delay, the knowledge about the transportation delay should play the biggest role in that decision. That is, not the length, but the variability of the delay. As it turns out, the mismatch between the true and estimated delay is a far greater cause for instability than a mismatch between the non-delayed transfer functions of P and G [5].

Remark 3.2.1. *Although there is literature available on DTC's for nonlinear processes, e.g. [24, 25], most literature on the Smith Predictor is based on linear controllers. The presented DTC implementation in this thesis is also based on a linear system. As such, only the linear part G of the Wiener model is used to control the die shape.*

This description of the Filtered Smith Predictor is for SISO control. However, the MIMO case is a generalization of the SISO case, and the condition for robust stability of the MIMO case are derived from those of the SISO case [5]. To describe this robustness condition, first the following concepts about model uncertainty in the frequency domain are introduced. To account for model uncertainty, it is assumed that the system $P(j\omega)$ can be described by a family of transfer functions, in such a way that the magnitude and phase can vary in a disk with radius of maximum $\overline{\Delta P}(\omega)$, as shown in Fig. 3.6. As such, each system $P(s)$ can be written in the frequency domain as

$$P(j\omega) = P_n(j\omega) + \Delta P(j\omega), \quad |\Delta P(j\omega)| \leq \overline{\Delta P}(\omega) \quad \forall \omega \geq 0, \tag{3.41}$$

where $P_n(j\omega)$ is the nominal model and $\Delta P(j\omega)$ is defined as the additive description of the modeling errors. Equivalently

$$P(j\omega) = P_n(j\omega)(1 + \delta P(j\omega)), \quad |\delta P(j\omega)| \leq \overline{\delta P}(\omega) \quad \forall \omega \geq 0, \tag{3.42}$$

where $\delta P(j\omega)$ is a multiplicative description of the modeling errors and

$$\delta P(j\omega) = \frac{\Delta P(j\omega)}{P_n(j\omega)}, \quad \overline{\delta P}(\omega) = \frac{\overline{\Delta P}(\omega)}{|P_n(j\omega)|} \quad \forall \omega \geq 0. \tag{3.43}$$

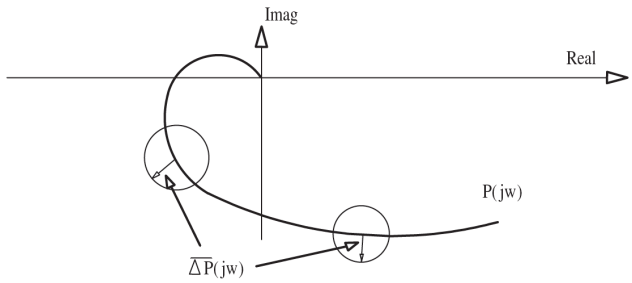


Figure 3.6: Model uncertainty disk in the frequency domain, obtained from [5].

Now that these concepts have been introduced, the robust stability condition for the Filtered Smith Predictor can be presented:

$$\overline{\delta P}(\omega) < \frac{|1 + C(j\omega)G(j\omega)|}{|C(j\omega)G(j\omega)F_{\mathrm{LP}}(j\omega)|}, \quad \forall \omega > 0. \tag{3.44}$$

From this condition, it becomes clear why the low-pass filter adds robustness. For higher frequencies, the gain of this filter will decrease, and in turn making the bound on $\overline{\delta P}(\omega)$ in (3.44) increase. Moreover, the addition of this filter does not decrease performance in case Ge^{-Ls} is a perfect representation of P, since the error e_p will be zero, and therefore the filter output will remain zero as well.

The properties of the Filtered Smith Predictor suggest the following two-step general procedure for tuning the controller:

1. Compute $C(s)$ in order to obtain the desired closed-loop performance of the model $G(s)$.

2. Estimate the uncertainties of the system and compute the filter F_{LP} in order to obtain robust stability or robust performance. The filter can be defined using known filter design techniques.

Obviously, the FEM model P is not exactly known, and the assumption is made that this model can be represented as a Wiener system. As such, it can never be verified whether the condition in 3.44 is met. However, by tuning the filter F_{LP} conservatively, robustness can be added to the control loop.

Chapter 4

Implementation and results

The methods presented in Chapter 3 are applied to the die optimization problem. In this chapter, both the implementation of these methods and the results are presented and compared to the original results presented in [1]. At first, the system that results from the system identification procedure is presented. Next, the implementation and results of the data-driven optimization methods are discussed. Lastly, the model-based Smith Predictor is treated.

4.1 System identification

Although the identified system only serves a function for the model-based Smith Predictor, which is covered last in this chapter, an identified system will benefit the data-driven optimization methods as well. For example, identifying the static nonlinearity of the Wiener model helps to verify the concavity of the surface of operating points as shown in Fig. 2.1. Moreover, it presents an insight into the transportation delay, which may help to select suitable extremum seeking parameters to ensure time scale separation. An additional advantage of having an identified model early on in the process, is to aid in simulating the response of the system. Implementing and simulating the different controllers in the FEM system requires common Matlab functions and Simulink blocks to be rewritten in Fortran. This is a time consuming process, that is aggravated by the time needed to run a simulation in the FEM environment. By implementing the controllers in the Matlab/Simulink environment on the identified model, this process is more streamlined while showing similar performance to the true FEM system.

The Wiener system is identified using the steps that are laid out in Section 3.2.1. This same outline is followed again here, where the choices for these steps are motivated and the results depicted.

1. The system is linearized around the operating point $(u_{\text{eq}}, y_{\text{eq}})$ $= (1.000, 1.455)$. Again, see Fig. 2.1.

2. An Output Error model structure is selected, that has the following form.

$$G(z,\theta) = z^{-n_k}\frac{B(z,\theta)}{F(z,\theta)} = z^{-n_k}\frac{b_0 + b_1 z^{-1} + ... + b_{n_b-1}z^{-(n_b-1)}}{1 + f_1 z^{-1} + ... + f_{n_f}z^{-n_f}}, \quad (4.1)$$

$$H(z,\theta) = 1, \quad (4.2)$$

$$\theta = [b_{n_b-1}, b_{n_b-2}, ..., b_0, f_{n_f}, f_{n_f-1}, ..., f_1]^\top \in \Theta \subset \mathbb{R}^{b_b+n_f}, \quad (4.3)$$

where $G(z,\theta)$ is monic, n_k is the number of delay samples, n_b and n_f are the number of parameters in the numerator and denominator of $G(z,\theta)$ respectively.

3. A multisine input signal $\Delta u(k)$ is generated using the Matlab function `idinput`. Using this function, it is possible to select the normalized frequency range for the input signal, where a normalized frequency of 1 represents the Nyquist frequency. Due to the desire for a *slow* input signal, the generated multisine is a sum of 4 sine waves such that for this signal the normalized frequencies contained in the set $\mathcal{F} = \{0.00314, 0.01257, 0.01885, 0.02827\}$ are all represented exactly one time, that $\min_n \Delta u(k) \geq -\delta u_{\max} = -0.1$ and $\max_n \Delta u(k) \leq \delta u_{\max}$, see (2.3), and that the mean of $\Delta u(k)$ is 0. Note again, that since $\Delta u(k)$ contains 4 sine waves with each a unique frequency, a maximum of 8 parameters can be estimated, i.e. $n_b + n_f \leq 8$. The FEM system has a sampling time of $T_s = 0.005$ seconds. Multiplying each element in $\mathcal{F}$ by $\frac{\pi}{2\pi T_s}$ yields the frequencies in Hz.

4. The resulting input signal is

$$u(k) = \begin{cases} u_{\text{eq}}, & \text{if } 0 \leq k < 3000 \text{ or } k \geq 5000, \\ u_{\text{eq}} + \Delta u(k-3000), & \text{if } 3000 \leq k < 5000, \\ 0, & \text{else.} \end{cases} \quad (4.4)$$

In essence, at first the input signal is constant to allow the system to settle to a constant value. After 3000 samples = 15 seconds the multisine is added to the signal. This signal has a length of 2000 samples = 10 seconds.

The signal $y[n]$ is collected from which $\Delta y[n]$ is then derived, such that

$$\Delta y(k) = \begin{cases} y(k+3000) - y_{\text{eq}}, & \text{if } 0 \leq k < 2000, \\ 0, & \text{else.} \end{cases} \quad (4.5)$$

5. Now that $\Delta u(k)$ and $\Delta y(k)$ are available, the linear part $G(z,\theta)$ from the Wiener system can be estimated. From experiments it follows that the input-output delay around the selected operating point is approximately 4 seconds. For the FEM system a short sampling time of $T_s = 0.005$ s is selected, ensuring that the dynamics accurately represent their continuous-time counterparts [26]. As such, a delay of 4 seconds amount to the number of delay samples $n_k = 800$. By trial-and-error it was found that the the number of estimation parameters $n_b = 4$ and $n_f = 4$ returned parameters

that best fitted the measured data to the simulated data, see Fig. 4.2a. Table 4.1 gives these parameters and Fig. 4.1 also depicts the Bode plot and pole-zero map of G.

6. Section 3.2.1 mentioned that the FEM system was subjected to the constant inputs u_{ss} from the sequence $0.5, 0.6, ..., 1.5$ and the corresponding constant outputs y_{ss} were measured. The surface of operating points depicted in Fig. 2.1 was obtained from these measurements. With this information, the static nonlinearity h_w can be derived. The constant input sequence follow from u_{ss} and u_{eq}, such that $\Delta u_{\text{ss}} = u_{\text{ss}} - u_{\text{eq}}$. Using (3.39), where $a = \Delta u$, and the above identified G gives the steady state Δy_{ss}. Note that for $G(z, \theta)$ with the optimal parameters

$$\Delta y_{\text{ss}} = \lim_{k\to\infty} \Delta y(k) = \lim_{z\to 1} aG(z) = \Delta u_{\text{ss}} \frac{\sum_{i=0}^{3} b_i}{1 + \sum_{j=1}^{4} f_j} \approx 0.8189 \Delta u_{\text{ss}}. \tag{4.6}$$

Now that Δy_{ss} and y_{ss} are available, the static nonlinearity h_w can be fitted using the Matlab function `polyfit`. The following 5th order polynomial was obtained:

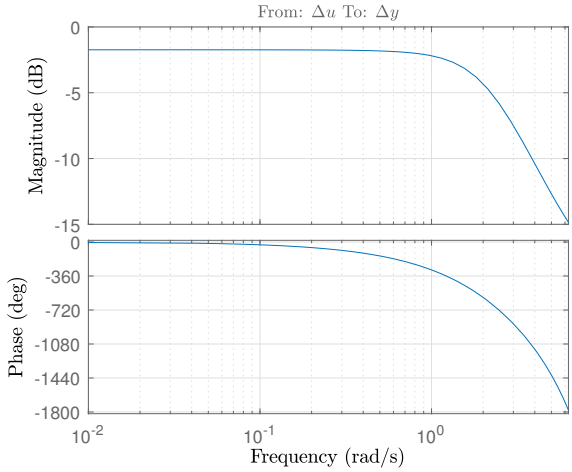
$$h_w(y_l) = -0.6517 y_l^5 + 4.5896 y_l^4 - 12.4226 y_l^3 + 15.8551 y_l^2 - 8.1645 y_l + 1.7149, \tag{4.7}$$

see Fig. 4.1c. One should be careful extrapolating using this fitted polynomial, and attempt to only evaluate this polynomial on the appropriate domain, i.e. $[0.8, 1.8]$. On this domain, the derivative of h_w has no real-valued roots, and as such is positive everywhere. This confirms the concavity required for the system identification process and the implemented extremum seeking and surrogate modeling schemes, at least when the identified Wiener model is used.

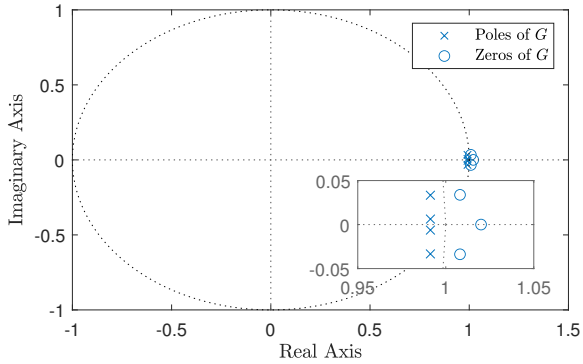
The identified model is validated by the following: (1) Subjecting the identified linear part G to the multisine input and comparing the output with the output of the true FEM system, and (2) applying the I-controller from the state of the art [1] on the identified Wiener system and comparing output and convergence properties. The outputs of these simulations have been depicted in Fig. 4.2. The dynamics of the Wiener system exhibit non-minimum phase behavior, as can be derived from Fig. 4.2a and can be explained by the zeros of G that lie outside the unit circle, see Fig. 4.1b.

4.2 Extremum seeking control

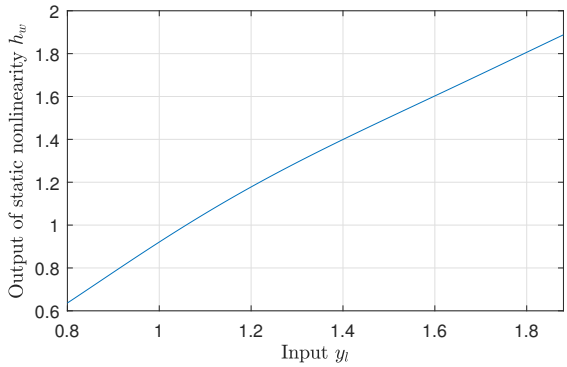
The data-driven optimization method Extremum seeking control is a robust optimization method, should the parameters be chosen correctly. As such, stability can be guaranteed, provided one accepts the principle of time-scale separation. Due to the delay between input and output, the system take a long time to



(a)

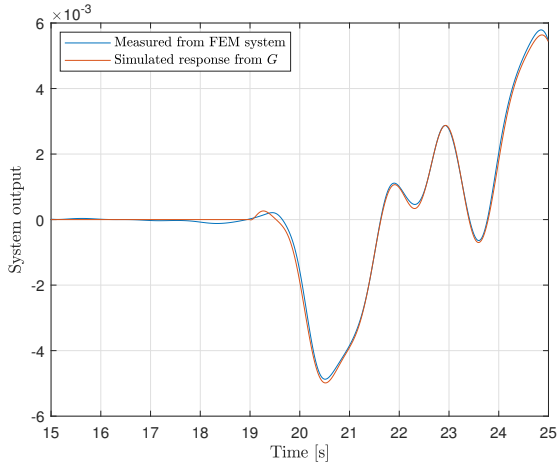


(b)

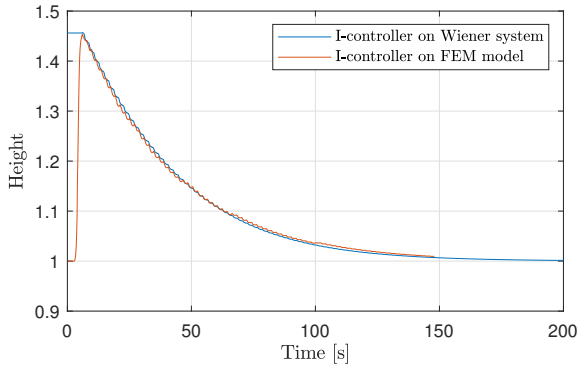


(c)

Figure 4.1: Identified Wiener model. (a) Bode plot of the linear part G, (b) pole-zero map of G, and (c) static nonlinearity h_w.



(a)



(b)

Figure 4.2: Model validation of the identified Wiener model. (a) Output from the multisine to identify the system, and (b) the identified Wiener model subjected to the I-controller from the state of the art [1].

Table 4.1: Optimal parameters for the model from (4.1) to (4.3). Only 4 significant figures are shown.

Parameter	Optimal value
b_0	-0.004637
b_1	0.01408
b_2	-0.01425
b_3	0.004812
f_1	-3.965
f_2	5.897
f_3	-3.899
f_4	0.9667

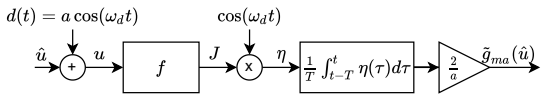


Figure 4.3: Extremum seeking derivative estimation based on a moving average filter. The gradient $\tilde{g}_{\mathrm{ma}}$ of f is estimated at $\hat{u}$.

reach steady state, resulting in the necessity for slow derivative estimation and even slower optimization.

Moreover, the cost function that is subject to minimization needs to be quasiconvex to avoid converging to local minima, with its minimum at the system's reference output. However, if certain assumptions about the system can be made, this condition can still be guaranteed without the need for a model of the system, resulting in guaranteed convergence the reference output.

In the case of SISO optimization, there is just a single design variable that needs to be optimized. In this case, the following extremum seeking parameters are chosen: dither amplitude $a = 0.1$, dither frequency $\omega_d = \frac{\pi}{20}$ rad/s and integrator gain $K_{\mathrm{DE}} = 0.01$. It is chosen that for this numerical case study only the first derivative of the function f is of interest. Consequently the derivative estimation equals that of the loop presented in Fig. 4.3. This loop uses a moving average filter, which is not standard for extremum seeking, see Appendix A.

The following quadratic cost function $g : \mathbb{R} \to \mathbb{R}$ is selected:

$$g(y) = (-y + y_r)^2, \tag{4.8}$$

where $y = h(u)$. Note that this is a strictly convex cost function with a global minimum at y_r and let Assumption 3.1.2 hold, stating that h is concave. Consequently, as shown in Section 3.1.1 the composite function $f = g \circ h$ is convex. Then, if a, ω_d and K_{DE} are selected sufficiently small, the estimated optimizer $\hat{u}$ will converge to the true optimizer u^* when time goes to infinity.

Numerical case study

This extremum seeking loop is implemented on the SISO identified Wiener system as presented in Section 4.1. However, due to extremum seeking being a data-driven optimization method, this system is treated as black box. The effect of extremum seeking parameters with respect to time scale separation and convergence is investigated, and expectations for practical use of extremum seeking are discussed. Revisiting the three time scales in extremum seeking from fast to slow: (1) the dynamic system, (2) derivative estimation, and (3) optimization. To conform to the time scale separation requirement, it is necessary that the phase difference between the sinusoidal input to the system and sinusoidal output of the system, i.e. the delay of the frequency response, is as small as possible. In this case, the system mapping the die height to the extrudate height has a significant transportation delay, resulting in a quickly decreasing phase of the system for increasing dither frequency ω_d. Below, three simulations with the presented extremum seeking framework are depicted and discussed. These simulations differ in the choice of extremum seeking parameters, i.e. dither amplitude, dither frequency and integrator gain, as to illustrate the effect of time scale separation. For these simulations both the initial condition $\hat{u}_0 = 1.0$ and the desired extrudate height $y_r = 1.0$ are consistent. Although of course unknown to the extremum seeking scheme, it follows from the Wiener system that for this reference the true optimizer $u^* \approx 0.517$.

Remark 4.2.1. *The extremum seeking control scheme is implemented in discrete-time in Simulink. This environment facilitates the conversion of a continuous-time process to a discrete-time process, and as such, this section does not elaborate on this conversion, except for the following. An integrator is needed to obtain the gradient estimate* $\tilde{\mathbf{g}}_{\hat{\mathbf{u}}}$ *of the objective function* $f(\mathbf{u})$ *at* $\hat{\mathbf{u}}$ *in* (3.24). *This integrator is implemented in discrete-time using the forward-Euler method.*

Slow extremum seeking This first case upholds the time scale separation principle is good as possible. This means that for the first time scale the dither frequency is chosen in such a way that there is minimal phase difference with the plant output. For the second time scale this means that the demodulation signal is also in phase with the plant output, and for the third time scale this means that the optimizer is slow enough as to not disrupt the system output too much or cause significant overshoot with respect to the set point. To achieve this, the following parameters were chosen for the simulation: dither amplitude $a = 0.1$, dither frequency $\omega_d = 0.01$ rad/s and integrator gain $K_{\mathrm{DE}} = 0.0002$. Although the identified Wiener system is treated as unknown, its linear systems frequency response has a phase delay of approximately 30 degrees at the dither frequency, which is sufficiently small to maintain time scale separation.

Remark 4.2.2. *In the context from the matrices and vectors introduced in Section 3.1.2, these extremum seeking parameters would result in the following.*

$$A = 0.1, p(t) = m(t) = \cos(\omega_d t) \implies K = 100\pi \implies G_0 = {}^1\!/_{10\pi}, K_{DE} = 0.0002. \tag{4.9}$$

Figure 4.4a depicts the output of the extremum seeking loop $\hat{u}(t)$, i.e. expected optimal die height, over time. This simulation shows that it takes more than 10000 seconds to converge to the optimum die height. This is very slow compared to the state of the art solution, see Fig. 2.2. However, the choice of extremum seeking parameters has ensured the conservation of time scale separation, and as such, convergence to the optimizer is guaranteed.

Relaxation of third time scale For this simulation, the third time scale is relaxed, such that the integrator gain is selected higher than withe the previous simulation. Consequently, the system output may not closely resemble a sinusoid and therefore, the derivative estimation may fail. The following extremum seeking parameters are chosen: $a = 0.1$, $\omega_d = 0.01$ rad/s and $K_{\mathrm{DE}} = 0.0007$. Figure. 4.4b shows the extremum seeking output for this simulation. It shows that although convergence to the optimal die height is achieved much faster in comparison to the first simulation, the output is much less smooth. This is the result of the relaxation of the third time scale, leading to poor derivative estimation. Even higher integrator gains would eventually result in instability of the loop.

Relaxation of the first time scale For this third and final simulation, the dither frequency is increased to $\omega_d = \pi$ rad/s. Again, the identified Wiener system is treated as black box, but inspecting the frequency response of the linear system, it is found this system has a phase lag of approximately 184 degrees at the dither frequency. This poses a significant problem for the time scale separation, in the sense that the derivative estimation is completely disturbed, because the system and output and demodulation signal are completely out of phase. In practice, means that a positive derivative of the system output with respect to the system input would be measured as a negative derivative and vice versa. On the other hand, a higher dither- and demodulation frequency can decrease the time needed to get an estimate of the derivative, since the derivative is estimated over the dither period, and as such converge to the optimizer faster. In [27] the estimated phase lag between the sinusoidal input and output of the dynamic system is used as a phase for the demodulation signal. That same procedure is followed in this simulation, i.e. each element in the vector $m(t)$ gets a phase shift of approximately -184 degrees, but it is noted that for the calculation of the matrix K, the vector $m(t)$ with no phase shift is used. The other extremum seeking parameters are as follows: $a = 0.1$ and $K_{\mathrm{DE}} = 0.07$. From Fig. 4.4c it becomes clear that this procedure allows for considerably faster convergence to the desired die height. In essence, if the phase lag of the system at the dither frequency is known, the dither frequency can be chosen much larger, and faster convergence can be achieved. Unfortunately, this phase lag needs to be estimated from e.g. a low order model, making this extremum seeking loop a hybrid between data-driven and model based optimization.

However, the Wiener model is estimated around an operating point, i.e. die height. Around this operating point, the transportation delay can be considered constant. Around a different operating point, the dimensions of the die are either

larger or smaller. For a constant flow speed of the polymer going into the die, this means that the same volume of the polymer is pushed through a larger or smaller die, resulting in the extrudate flowing slower or faster respectively when exiting the die. This means that the transportation is not and cannot be considered as constant.

Figure 4.1a contains the Bode plot of the linear part G of the identified Wiener system. It shows that the phase quickly reduces when the frequency increases, caused by the transportation delay. As such, any model mismatch between true system and identified system may result in a phase estimation that is completely off. For the most accurate extremum seeking, it is necessary that the phase is estimated as good as possible.

In Fig. 4.8 the output of the true system to multiple tuning configurations of a Smith Predictor is depicted. Figure 4.8 shows that the more aggressive controllers initially show a shorter transportation delay. A more aggressive controller will decrease the die height quicker compared to a less aggressive controller. As such, the fluid with a constant flow speed will have to travel through the die faster and will therefore reach the measurement point quicker. From this figure it becomes clear that the transportation delay cannot be accurately captured with the identified Wiener model. As a result, unless the phase lag can be properly estimated, this last extremum seeking implementation is not a viable solution.

4.3 Surrogate modeling

Just as extremum seeking, surrogate modeling is a data-driven control approach. In extremum seeking, the input signal that perturbs the system is slow, and as such the measured output signal is close to being steady state. In surrogate modeling, the input does not perturb the system, but remains constant for a period of time, as to ensure an output that is in steady state. After collecting multiple steady state input-output pairings, a pattern is estimated, e.g. in the form of a polynomial that explains the measured input-output pairings. In turn, this pattern allows for an estimation of the optimal input to let the system converge to the desired output.

Surrogate modeling is implemented as described in Section 3.1.3 on the estimated Wiener model from Section 4.1. It should be noted that, since the transient behavior does not contribute to the measured input-output pairings, implementing this solution on the Wiener model should be almost identical to implementing the solution on the true FEM system in Fortran. Essentially, only a mismatch in the estimated static nonlinearity h has an influence on the difference between the true and estimated performance, since the steady state inputs u_{ss} can be directly linked to the steady state outputs y_{ss}, see again Section 3.1.3. The selected reference $y_{\text{ref}} = 1.0$ is the same as with extremum seeking. The monomials s_k are chosen as $s_0(u) = u^2, s_1(u) = u, s_0(u) = 1$, such that $s(\mathbf{c}, u) = c_0u^2 + c_1u + c_2$, i.e. a quadratic polynomial. This leads to the number of monomials $K = 3$ in s. The number of evaluations of the objective

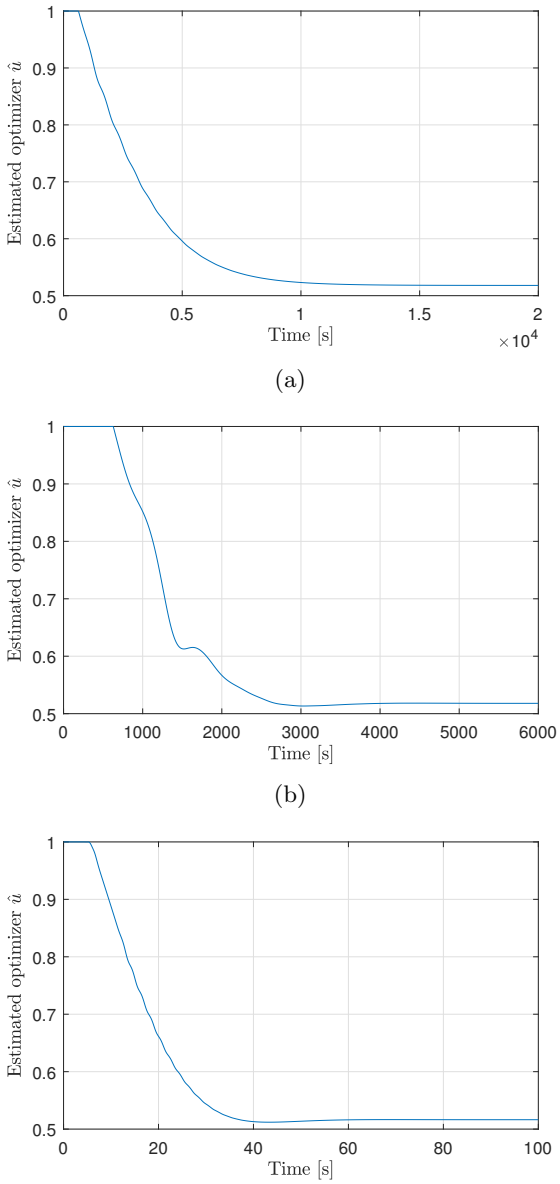


Figure 4.4: Implementation of Extremum Seeking on the SISO optimization problem. (a) Slowest simulation, but time scale separation holds and stability is thus guaranteed. (b) Relaxation of the second time scale. Faster than simulation (a), but nears instability, and (c) shows the fastest convergence to the desired die height. However, this method required accurate knowledge of the system's phase lag for the dither frequency. For higher frequencies, the phase is very sensitive to an estimation error of the transportation delay.

function f that contribute to the fit is chosen as $M = 3$. Although the objective function f is convex, it is not quadratic, and as such s will not be an exact fit of f. By increasing the number of objective function evaluations, i.e. $N > M$, the domain on which s is estimated is reduced as N increases. Subsequently, earlier objective function evaluations that are essentially outliers are ignored. The total number of objective function evaluations is chosen as $N = 10$. The initial inputs are chosen by means of a central composite design on the interval $[a, b] = [0.5, 1.0]$, as it is expected that the optimal input u^* will lie in this interval. This leads to $u_0 = 0.5, u_1 = 0.75$ and $u_2 = 1.0$. For each iteration, the constant input is held for 10 seconds. This should give the system sufficient time to reach steady state. After M iterations, the optimal coefficients $\mathbf{c}^*$ for $J(\mathbf{c})$ can be found. Then using the Matlab function `fmincon`, the optimal input $\hat{u} \approx u^*$ can be estimated. Then based on $\hat{u}$, a new input u can be selected as $u_{i+1} = \hat{u}_i + \epsilon_i$, where ϵ_i is a small random value to prevent singularity of the matrix S. For these simulations, ϵ_i is selected to lie on the interval $[-0.01, 0.01]$ for all i.

Remark 4.3.1. *Note that, since $s(\mathbf{c}, u)$ is quadratic in u, the optimizer u^* could be found in a similar fashion as the optimizer $\mathbf{c}^*$ of $J(\mathbf{c})$ is found, i.e. by using the pseudo-inverse of the matrix S as in Algorithm 1. However, the function* `fmincon` *allows to set a lower- and upper bound, which can be selected as a and b respectively.*

Remark 4.3.2. *Since h is concave and g is quadratic, making f convex, the quadratic polynomial s that approximates f can be assumed to be convex as well. However, to guarantee convexity of s, bounds can be set on the coefficients $\mathbf{c}^*$ using e.g.* `fmincon` *to estimate $s(\mathbf{c}^*, u)$.*

Figure 4.5 shows the estimated optimal input $\hat{u}$ after each simulation. Note that, since $M = 3$, it takes 3 simulations to obtain a first estimate. After 5 simulations, $\hat{u}$ is estimated sufficiently well, and after 6 simulations, $\hat{u}$ converges to the true optimum input u^* and remains tolerably constant. Compared to the state of the art die shape optimization which takes 146 seconds to converge, this is a significant reduction in convergence time.

4.4 Dead-time compensation using a Smith Predictor

The Smith Predictor attempts to steer an estimated model of the true system to the desired reference. This estimated model has the transportation delay removed, and as such, if the model is correct, convergence to the reference should be faster. This section covers the implementation of the Smith Predictor. The tuning parameters are explained and the performance on not only the identified model, but also the true FEM model is shown.

The Smith Predictor is implemented according to Section 3.2.2. As mentioned, this controller uses linear estimation G and the delay e^{-Ls} of the system

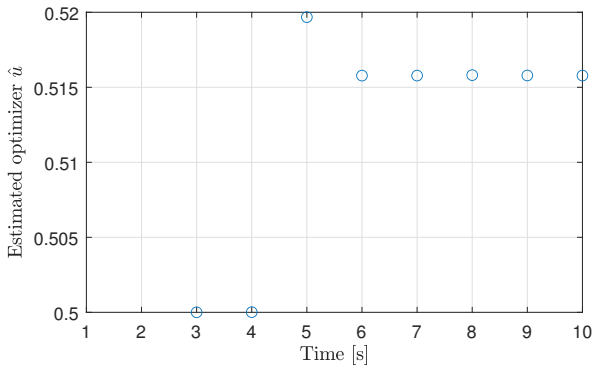


Figure 4.5: Estimated optimal die height as a function of the number of simulations run for SISO optimization. The reference extrudate height is 1.0. Each simulation takes 10 seconds, meaning that 5 simulations take 50 seconds. Since the objective function f is fitted using a quadratic polynomial, 5 objective function evaluations are necessary and as such, the optimum die height can be estimated after 50 seconds.

P. Consequently, when looking at the estimated Wiener model in 4.1 of the true system P, the static nonlinearity h is omitted.

The main focus of this implementation of the Smith Predictor is an analysis on its convergence properties, rather than the selction of a controller type that maximizes the convergence rate of the die shape. As such, the controller C is chosen as a PID controller that is tuned based on the model G using the Matlab command `pidtune (sys,type,wc)`, that tunes the model `sys` with a controller of type `type` (e.g. PID) with a crossover frequency of `wc` radians per second for the open-loop response. This command uses an algorithm that has the following three objectives: (1) closed-loop stability, (2) adequate performance, and (3) adequate robustness. For a process that is modeled as a transfer function, there always exists a state feedback law that stabilizes the closed loop system. The cutoff frequency `wc` needs to be chosen sufficiently small to prevent numerical instability, see 2.3.

Since the Smith Predictor is implemented on the true FEM system in Fortran, a discrete-time environment, the PID controller C needs to be discretized. A discretization of the PID controller is given here. The overall control function is

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de(t)}{dt}, \tag{4.10}$$

where the control parameters K_p, K_d and K_i are obtained with the function `pidtune` as mentioned above. A discrete time implementation of this controller,

obtained using backward finite-differences approximations [26], is

$$u(n) = u(k-1) + \left(K_p + K_i T_s + \frac{K_d}{T_s}\right) e(k) + \left(-K_p - \frac{2K_d}{T_s}\right) e(k-1) + \frac{K_d}{T_s} e(k-2), \quad (4.11)$$

where T_s is the sampling time. Note that the tuning parameters K_p, K_d and K_i remain unchanged.

The Smith Predictor is implemented on the following three systems: (1) just the identified linear model, i.e. in Fig. 3.5 the system P is chosen as G, (2) the identified Wiener model, i.e. P is set to G followed by the static nonlinearity h, and (3) on the true FEM system. All these systems used the controller obtained from the function `pidtune (sys,type,wc)` with `sys` chosen as the model G, `type` chosen as PID and cutoff frequencies `wc` chosen as $0.05, 0.1, 0.2, 0.3$. From numerical simulations it follows that these cutoff frequencies are small enough that no numerical instabilities occur, see 2.3. The desired extrudate height is selected as $y_r = 1.0$. For all models, a low-pass filter is included to increase robustness to modeling errors. This is a first-order filter with the following transfer function in both continuous-time and discrete-time respectively:

$$F_{\mathrm{LP}}(s) = \frac{k_{\mathrm{LP}}\omega_{\mathrm{LP}}}{s + \omega_{\mathrm{LP}}}, \quad F_{\mathrm{LP}}(z) = \frac{k_{\mathrm{LP}}\omega_{\mathrm{LP}}}{\frac{z-1}{T_s} + \omega_{\mathrm{LP}}}, \quad (4.12)$$

where $F_{\mathrm{LP}}(z)$ is the result of the discretization of $F_{\mathrm{LP}}(s)$ using the Forward Euler Method. The cut-off frequency ω_{LP} is chosen as 0.1 rad/s and the gain $k_{\mathrm{LP}} = 1$. The choice of these parameters is motivated later in this section.

Remark 4.4.1. *As to remain consistent with [5] and the introduction of the Smith Predictor in Section 3.2.2, the analysis for the Smith Predictor on the die shape optimization problem will be treated in continuous-time. As such, all units are in e.g. rad/s and seconds, rather than rad/sample and samples respectively. The effect on robustness is given in Remark 4.4.2.*

Remark 4.4.2. *The robustness of the Smith Predictor has been investigated in Section 3.2.2, and will be complimented in this section, for continuous-time processes. However, the Smith Predictor is implemented in discrete-time for the die shape optimization problem. The work [5] discusses the usage of these continuous-time robustness properties for discrete-time processes. It concludes that these concepts remain valid, and that the focus of maintaining robustness should lie on the magnitude of the modeling errors resulting from the discretization process. For this die shape optimization problem, both the controller C and low-pass filter F_{LP} are converted from continuous-time to discrete time, while the G is already a discrete-time process.*

Situation 1: Linear model At first, the system P in Fig. 3.5 is chosen as just the identified linear model G, i.e. $P = Ge^{-Ls}$. This is also the model to which

the controller C has been tuned. As such, using this implementation, the Smith Predictor can be expected to have the best performance when an aggressive controller is used, compared to the other two implementations. Figure 4.6a depicts Smith Predictor's performance for different crossover frequencies and in Table 4.2 in terms of time needed to converge to a region of a maximum error of 1% of the reference. From both the figure and the table it can be concluded that the higher the open-loop crossover frequency, the shorter is the time needed to converge to the desired reference.

Section 3.2.2 mentions that the cause that contributes the most to decreased robustness is a mismatch between the transportation delay present in P and the estimated delay L for the estimation of $P \approx Ge^{-Ls}$. In Fig. 4.7a this effect is investigated.

In support of these simulations, and using the concepts from (3.41) to (3.43), the robustness condition (3.44) can be expanded as follows:

$$\frac{|\Delta P(j\omega)|}{|P_n(j\omega)|} \leq \overline{\delta P}(\omega) < \frac{|1 + C(j\omega)G(j\omega)|}{|C(j\omega)G(j\omega)F_{\mathrm{LP}}(j\omega)|}, \quad \forall \omega > 0, \tag{4.13}$$

where $\Delta P(j\omega) = P(j\omega) - P_n(j\omega)$. Using this expanded robustness condition, the robustness of the simulations in Fig. 4.7a can be analyzed. Note that the right-hand side of (4.13) only depends on the estimated model G, the controller C and the filter F_{LP}. As such, the controller and filter can be tuned independently of the actual system to increase robustness.

It is desirable to have the right-hand side of this equation to be as large as possible to guarantee robustness. If F_{LP} is a low-pass filter, this right hand side will be larger for ω above the cutoff frequency ω_{LP} of F_{LP}. As such, it would make sense to have ω_{LP} as small as possible. However, having a conservatively tuned filter results in a lower rate of convergence. Therefore, one must attempt to find a filter that makes the right hand side of (4.13) as small as possible, while satisfying the sufficient robustness condition. Note that the robustness of the Smith Predictor is also influences by the controller. However, the tuning of the controller will always influence the convergence rate, while the filter tuning will only have an influence if there is a model mismatch, i.e. $P \neq Ge^{-Ls}$.

Using a numerical case study, (4.13) is numerically validated, and the effect of different parameters ω_{LP} and k_{LP} for F_{LP} in (4.12) is investigated. From Fig. 4.7b it follows that the system dynamics become unstable when the left-hand side and right-hand side of (4.13) cross. Choosing different filter parameters ω_{LP} and k_{LP} will however alter the right-hand side of (4.13), resulting in stable system dynamics.

Situation 2: Wiener model In this situation, the system P in Fig. 3.5 is chosen as the identified Wiener model which includes both the identified linear model G and the identified static nonlinearity h. Consequently, the models P and G are not the same and as such $e_p(t)$ in Fig. 3.5 will not always be zero, resulting in the need for a conservatively tuned filter F_{LP}. Unfortunately, no literature was discovered that treats robustness of a Wiener system on a Smith

Predictor. It is expected however, that a filter gain $k_{\mathrm{LP}} = 1$ and crossover frequency $\omega_{\mathrm{LP}} = 0.1$ rad/s will deliver the expected robustness.

As shown in both Table 4.2 and Fig. 4.6b , convergence of $y(t)$ to the desired reference y_r is slower than when only the linear model is used, i.e. $P = G$. This happens since the system output P is different than that of the model G, and a correction has to be made. The figure shows a clear contrast between Situation 1 and Situation 2: where Situation 1 has no overshoot and is able to converge to the desired die height in an efficient manner, Situation 2 needs to correct for the overshoot, ultimately leading to slower convergence.

Situation 3: FEM model The Smith Predictor has been implemented on the true FEM system, such that the true FEM system is P. Although the linear dynamics of the estimated Wiener model was found to behave fairly similar tot the true FEM system close to the linearization point, this does not have to be the case when looking at the global system behavior. Moreover, the Smith Predictor uses the linear model G as its main model to be controlled, and the difference between the actual output P and the expected output Ge^{-Ls} to correct any model mismatch. The fact that the model G is linear, will only make this model mismatch between P and G larger, compared to Situation 2. However, for this situation, the low-pass filter keeps the original tuning parameters, such that $k_{\mathrm{LP}} = 1$ and $\omega_{\mathrm{LP}} = 0.1$ rad/s. Again, Table 4.2 shows the time needed for the output to converge to the reference extrudate height y_r and Fig. 4.8 shows the simulations for different open-loop bandwidths ω_c.

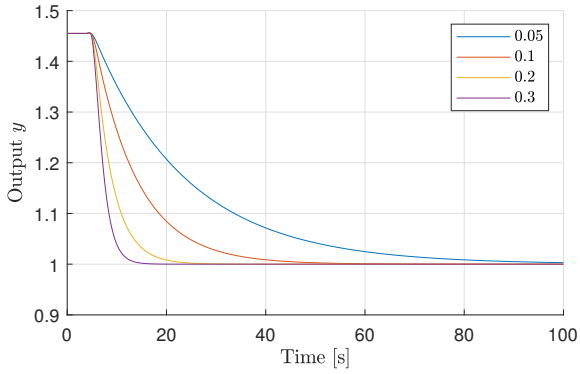
Interestingly, convergence time does not necessarily decrease when ω_c increases, which was the case for the other two situations. Instead, the convergence time appears to be decreasing at first for an increase ω_c, but seems to increase again when $\omega_c > 0.1$ rad/s. When looking at Fig. 4.8b, it is observed that the transportation delay seems to get shorter for increasing ω_c. This is most likely caused by the aggressiveness of the controllers with a higher bandwidth, resulting in the die height to decrease more rapidly than is the case with the more conservative controllers. Consequently, since the fluid flow velocity of the polymer is constant, the same volume of polymer is pushed through a smaller die, resulting in a higher fluid velocity of the polymer once it exits the die. As such, the polymer reaches the measuring point for the extrudate height faster then it would with the more conservative controllers with a smaller bandwidth. As mentioned in Section 3.2.2, the negative control performance resulting from a difference between the estimated transportation delay and the actual transportation delay far outways the negative performance resulting from a mismatch in the expected dynamics of the model.

Figure 4.8b shows that the controller with an open-loop bandwidth of $\omega_c = 0.1$ rad/s initially approximates the estimated transportation delay best, i.e. 4 seconds. Although this may not be the case later on in the simulation, when the die height has decreased and the transportation delay is therefore different, the initial control action does not disrupt the estimated output of the Smith Predictor in such a way that a large correction has to be made, resulting in oscillatory behavior [5], as is the case with the more aggressive controllers.

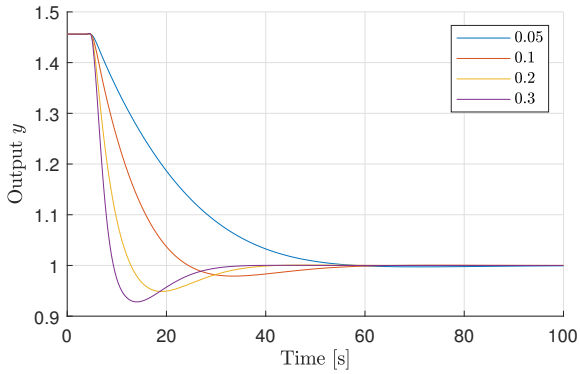
Table 4.2: Performance in the form of time needed to converge to the reference $y_r = 1.0$ for the Smith Predictor on both the estimated Linear model, Wiener model and the true FEM system.

ω_c [rad/s]	Time Linear [s]	Time Wiener [s]	Time FEM [s]
0.05	77	48	33
0.1	38	46	26
0.2	19	33	28
0.3	12	27	32

Remark 4.4.3. *Robustness to modeling errors must be considered a requirement in the application of the Smith Predictor. As shown above, the transportation delay is not constant and is dependent on the current shape of the die, i.e. the area of the die through which the polymer is pushed. Furthermore, the identified Wiener model from Table 4.1 and* (4.7) *is to play a key role in the die shape optimization for other polymers. Namely, it is not desired to identify a Wiener model for each polymer subject to optimization, as the desired die shape can be directly derived from the surface of operating points, see Fig. 2.1. As such, for each polymer and using the identified Wiener model from Table 4.1 and* (4.7)*, the Smith Predictor will quickly steer the die shape close to the optimal one, and the robustness properties ensure it will eventually converge to the optimal die shape.*

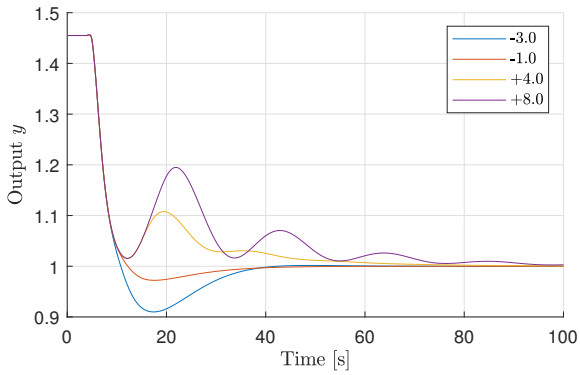


(a)

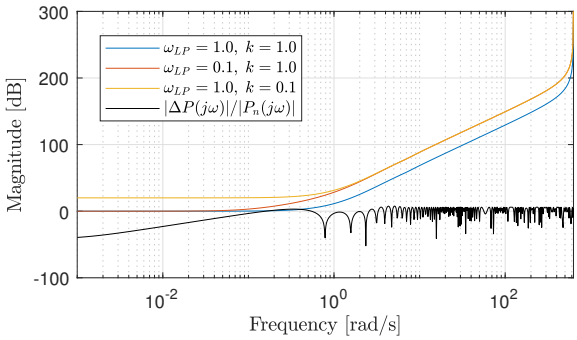


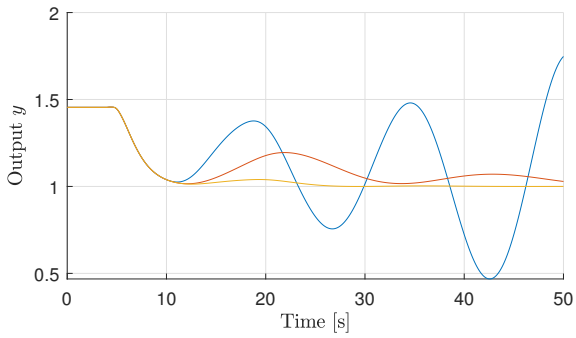
(b)

Figure 4.6: The Smith Predictor simulated on the estimated Wiener model. (a) Only the linear model G is used and the static nonlinearity h is omitted. The open-loop bandwidth is depicted in the legend in rad/s. (b) Simulations on the full Wiener model, such that the static nonlinearity h is no longer omitted. Again, the legends shows the open-loop bandwidths in rad/s.



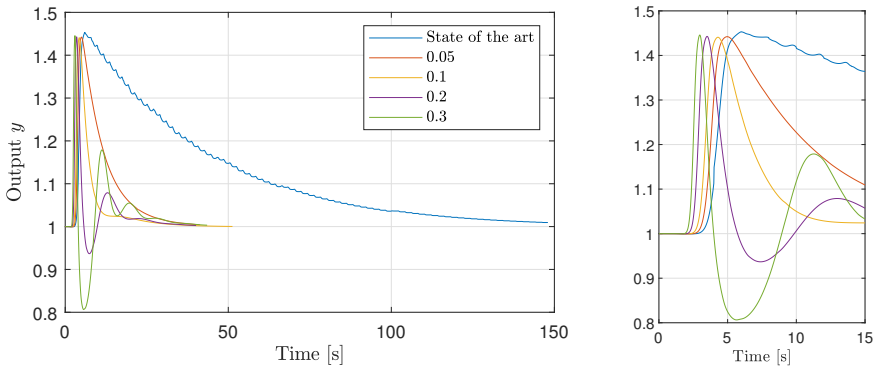
(a)





(b)

Figure 4.7: Numerical case study on the effect of a mismatch between the estimated transportation delay L for G and the true transportation delay of P and the effect of a low-pass filter to add robustness. (a) Effect of the transportation delay mismatch with a controller bandwidth `wc` of 0.3 rad/s, low-pass filter cut-off frequency $\omega_{\mathrm{LP}} = 0.1$ rad/s and gain $k_{\mathrm{LP}} = 1$. The legend contains the delay mismatch in seconds for each simulation, where a positive mismatch means the true delay is longer than L. (b) Effect of filter parameters on robustness. Here too, `wc` is 0.3 rad/s.



(a) (b)

Figure 4.8: Simulation of multiple tuning configurations of the Smith Predictor on the SISO FEM system. The number in the legend means the bandwidth of the closed loop for that simulation. (a) All simulations fully captured, and (b) all simulations only captured for the first 15 seconds.

Chapter 5

Conclusion and future work

In this final chapter, the main conclusions of this thesis are presented and recommendations for future work are given. These conclusions focus on the control problem and how certain assumptions and process knowledge are used to guarantee stability or improve performance. The recommendations for future work regard extensions to higher dimension control problems, such that MIMO die shape optimization can be improved upon as well.

5.1 Conclusions

Die shape optimization using classic feedback control is limited by slow dynamics and a significant transportation delay. These properties result in a need for conservative tuning parameters to prevent instability of the optimization process. Moreover, the dynamics of the extrudate process are not exactly known, and rely on assumptions. The work on which the state of the art was based did not utilize or explicitly state any assumptions of the process, apart from it being a stable process, to guarantee stability or improve convergence speed. The research conducted for this thesis, explored three types of optimization techniques, each with their own advantages and disadvantages.

The first method, extremum seeking, perturbs the system to obtain local derivative information of a cost function that describes the location of the input that optimizes the process, and subsequently let the system converge to that point. The applied method utilizes the principle of timescale separation, making it a relatively slow optimization method. However, under certain assumptions which are valid for the die shape optimization problem and with the tuning parameters chosen conservatively, convergence to a small neighborhood of the optimizing input is guaranteed. Precise knowledge on the transportation delay could be used to significantly improve convergence time.Extremum seeking is also suitable for higher dimension control problems, i.e. MIMO optimization problems, since the highest possible convergence rate depends on the speed of the process dynamics, rather than the number of objective function evaluations.

However, the applied extremum seeking method relies on the objective function having a single extremum, which may not be guaranteed for the MIMO die shape optimization problem.

Surrogate modeling, the second method, estimates the optimal input by evaluating the objective function a limited number of times, and subsequently estimates the optimal input using assumptions on the system's steady state input-output pairings. As such, this process will not smoothly converge to the true optimizer, but is able to guess the optimizer with sufficient accuracy in a relatively short time span compared to extremum seeking. However, in contrast to extremum seeking, the applied surrogate modeling method lacks a stability guarantee. Furthermore, surrogate modeling is less suitable for scaling to higher dimensions, since the number of required objective function evaluations to identify the input-output pairings increases drastically with each added dimension, to ensure uniqueness of the solution. However, it is expected that this process is still acceptably fast for a 4-dimensional problem.

The Smith Predictor is the third applied method. Where extremum seeking and surrogate modeling only rely on process assumptions, the Smith Predictor needs a model of the system. The requirement of a model limits flexibility in the use of different polymers for the die shape optimization. However, with an available robustness requirement and the usage of a low-pass filter in the control loop, the controller and filter can be tuned conservatively to gain robustness, at the trade-off of decreased performance. In this sense, a controller that was tuned for a standard polymer, can be used for other polymers as well, limiting the required process knowledge just as with extremum seeking and surrogate modeling. However, to guarantee robustness, the modeling errors have to be known, and this is not the case for this problem. In terms of convergence speed, this process was the fastest of the three solutions, but required more efforts in tuning the controller. If a higher dimension model is available, this method is also suitable for the MIMO die shape optimization, since the MIMO robustness properties are an extension of the SISO robustness properties.

5.2 Recommendations for future work

From the conclusions drawn above, the conducted research can be followed up by:

- For the MIMO die shape optimization problem, the surface of operating points is multivariate and is not necessarily convex. In that case, optimization methods that rely on the objective function having a unique extremum, e.g. extremum seeking, are not suitable for MIMO die shape optimization. Future research could entail the design of a cost function that composites the surface of operating points, such that the overall objective function is made convex, and thus contains a local minimum.

- From applying extremum seeking, it was found that the transportation delay has a significant impact on the first timescale, and as such, the

entire optimization loop is slow. Future research could focus on applying extremum seeking where convergence time is improved in the presence of a transportation delay.

Knowledge on the phase shift between the perturbation signal and demodulation signal can potentially be used to significantly increase the frequency of the perturbation signal, and increasing the speed with which the gradient of the objective function can be estimated, thereby ignoring the first timescale. An attempt to estimate this phase shift could be made by real time knowledge of (1) the flow rate of the polymer, and (2) the area of the die opening. As such the transportation speed of the polymer can be calculated resulting in the transportation delay, and consequently the phase shift. A method to identify a model using this concept is described in [5].

Bibliography

[1] M.M.A. Spanjaards, M.A. Hulsen, and P.D. Anderson. "Die shape optimization for extrudate swell using feedback control". In: *Journal of Non-Newtonian Fluid Mechanics* 293 (2021), p. 104552.

[2] J.W. Summers and R.J. Brown. "Practical principles of die design—a simplified procedure, in table form, for rigid PVC". In: *Journal of Vinyl Technology* 3.4 (1981), pp. 215–218.

[3] J.F.T. Pittman. "Computer-aided design and optimization of profile extrusion dies for thermoplastics and rubber: a review". In: *Proceedings of the Institution of Mechanical Engineers, Part E: Journal of Process Mechanical Engineering* 225.4 (2011), pp. 280–321.

[4] S. Elgeti, M. Probst, C. Windeck, M. Behr, W. Michaeli, and C. Hopmann. "Numerical shape optimization as an approach to extrusion die design". In: *Finite Elements in Analysis and Design* 61 (2012), pp. 35–43.

[5] J.E. Normey-Rico and E.F. Camacho. *Control of dead-time processes*. Springer, 2007.

[6] B.G.B. Hunnekens, M.A.M. Haring, N. van de Wouw, and H. Nijmeijer. "A dither-free extremum-seeking control approach using 1st-order least-squares fits for gradient estimation". English. In: *Proceedings of the 53rd IEEE Conference on decision and control (CDC 2014), 15-17 December 2014, Los Angeles, California*. United States: Institute of Electrical and Electronics Engineers, 2014, pp. 2679–2684.

[7] Dragan Nesic, Ying Tan, and Iven Mareels. "On the Choice of Dither in Extremum Seeking Systems: a Case Study". In: *Proceedings of the 45th IEEE Conference on Decision and Control*. 2006, pp. 2789–2794.

[8] M. Krstic and H. Wang. "Design and stability analysis of extremum seeking feedback for general nonlinear systems". In: *Proceedings of the 36th IEEE Conference on Decision and Control*. Vol. 2. 1997, 1743–1748 vol.2.

[9] M.J. Kortenhoeven and T.A.C. Keulen. "Recursive Least Squares Derivative Estimation for Fast and Accurate Extremum Seeking Control". In: *IEEE Control Systems Letters* (2022). Paper submitted for review.

[10] L. Wang, S. Chen, and H. Zhao. "A novel fast extremum seeking scheme without steady-state oscillation". In: *Proceedings of the 33rd Chinese Control Conference*. 2014, pp. 8687–8692.

[11] D. Nesić, Y. Tan, C. Manzie, A. Mohammadi, and W. Moase. "A unifying framework for analysis and design of extremum seeking controllers". In: *2012 24th Chinese Control and Decision Conference (CCDC)* (2012), pp. 4274–4285.

[12] R. van der Weijst, T.A.C. van Keulen, and F.P.T. Willems. "A generalized framework for perturbation-based derivative estimation in multivariable extremum-seeking". English. In: *IFAC-PapersOnLine* 50.1 (July 2017). 20th World Congress of the International Federation of Automatic Control (IFAC 2017 World Congress), IFAC 2017 ; Conference date: 09-07-2017 Through 14-07-2017, pp. 3148–3153.

[13] Y. Tan, D. Nešić, and I. Mareels. "On non-local stability properties of extremum seeking control". In: *Automatica* 42.6 (2006), pp. 889–903.

[14] M. Guay. "Finite-time extremum seeking control for a class of unknown static maps". In: *International Journal of Adaptive Control and Signal Processing* 35 (June 2020).

[15] J.I. Poveda and M. Krstić. "Fixed-Time Gradient-Based Extremum Seeking". In: *2020 American Control Conference (ACC)*. 2020, pp. 2838–2843.

[16] L.F.P. Etman. "Surrogate modeling for engineering optimization: a tutorial". Lecture slides for the course 4DM20: Engineering Optimization at the Eindhoven University of Technology.

[17] R.G. Regis and C.A. Shoemaker. "A Stochastic Radial Basis Function Method for the Global Optimization of Expensive Functions". In: *INFORMS J. Comput.* 19 (2007), pp. 497–509.

[18] H.M. Gutmann. "A Radial Basis Function Method for Global Optimization". In: *JOURNAL OF GLOBAL OPTIMIZATION* 19 (1999), p. 2001.

[19] J. Hespanha. *Linear Systems Theory*. Princeton University Press, Sept. 2009.

[20] G.F. Franklin, J.D. Powell, and A. Emami-Naeini. *Feedback Control of Dynamic Systems (7th Edition)*. Pearson, 2014.

[21] L. Wang. *Model Predictive Control System Design and Implementation Using MATLAB®*. Advances in Industrial Control. Springer London, 2009.

[22] H.K. Khalil. *Nonlinear systems; 3rd ed.* The book can be consulted by contacting: PH-AID: Wallet, Lionel. Upper Saddle River, NJ: Prentice-Hall, 2002.

[23] P.M.J. van den Hof. "System Identification - Data-driven model learning of dynamic systems". Lecture slides for the course 5SMB0: System Identification at the Eindhoven University of Technology.

[24] J. Normey-Rico, D.M. Lima, and T. Santos. "Robustness of Nonlinear MPC for Dead-time Processes**This work was financed by CNPq-Brasil (Conselho Nacional de Desenvolvimento Científico e Tecnológico)." In: *IFAC-PapersOnLine* 48 (Dec. 2015), pp. 332–341.

[25] R. Wright and C. Kravaris. "Nonlinear decoupling with deadtime compensation". In: *IFAC Time Delay Systems* (2003).

[26] K. J. Aström and B. Wittenmark. *Computer Controlled Systems: Theory and Design*. Prentice Hall, 1984.

[27] M.A.M. Haring, N. van de Wouw, and D. Nesic. "Extremum-seeking control for nonlinear systems with periodic steady-state outputs". English. In: *Automatica* 49.6 (2013), pp. 1883–1891.

Appendix A

Extremum seeking paper submitted for review

Recursive Least Squares Derivative Estimation for Fast and Accurate Extremum Seeking Control

Martijn Kortenhoeven[1] and Thijs van Keulen[1,2]

***Abstract*— This paper investigates accurate and fast online optimization of unknown nonlinear systems in the presence of measurement noise. Classical Extremum Seeking (ES) approaches seek an average system to guarantee convergence to a neighborhood of the extremum. The convergence speed is limited due to time-scale separation requirements associated to averaging while precise convergence to the true optimizer is limited by a trade-off between the signal-to-noise ratio and the steady-state error. This paper advocates to employ a Moving Average (MA) filter for Derivative Estimation (DE) in ES. An analysis demonstrates that the MA filter approach can achieve superior DE accuracy compared to finite-order DE filtering and avoids the need for averaging with the associated delay. Also, the analysis shows that the DE with a MA filter coincides with a recently developed DE based on linear Least-Squares (LS) fitting. By further exploiting this insight, this paper provides a novel DE based on high-order LS fitting in a receding horizon fashion which reaches superior steady-state convergence accuracy in the presence of measurement noise. A numerical case study demonstrates the strength of the approach in terms of accuracy and convergence speed.**

I. INTRODUCTION

Extremum Seeking (ES) is a data-driven control method for online performance optimization of systems. A typical ES feedback interconnection consists of the following elements: An unknown mapping of the system input to a scalar objective function output, an online Derivative Estimation (DE) algorithm that estimates the local gradient of the measured output with respect to the input, and an online optimization method that utilizes the gradient information to find the optimizer of the unknown objective function, see Fig. 1.

The ES framework in [10] exploits time-scale separation to achieve convergence towards a neighborhood of the true optimizer in the sense that the optimizer update rate is low compared to the delay introduced by the DE. A key ingredient for the proper functioning of ES is thus to obtain accurate and fast estimation of local gradient information of the unknown objective function. In practice, however, the system output measurement is corrupted with external disturbances which reduce the correlation between system input and measured cost output.

To provide robustness of the DE, to, e.g., measurement noise, a sinusoidal perturbation signal is commonly added to the system input. While injection of a sinusoidal dither signal improves the signal-to-noise ratio between input and output, there are unfortunately also disadvantages to the use of active perturbation. Firstly, the frequency of the dither signal is directly related to the delay introduced by the DE and, therefore, limits the convergence speed of the optimization. Secondly, continuous perturbation in the neighborhood of the optimum can lead to a steady-state error in case the unknown objective function is not symmetric around the optimizer, which occurs in various applications, such as anti-lock braking systems [2].

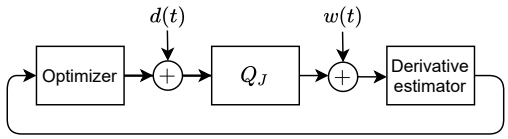


Fig. 1: Overview of the ES implementation with cost mapping Q_J, perturbation signal d, and noise disturbance w.

Against this background, the DE approach in classic ES [10] employs, subsequently, high-pass filtering, demodulation, and low-pass filtering to the measured objective output. Generalizations of the classical ES approach exist, both to multiple input ES and to ES using higher-order derivatives, see, for instance [11]. Recent developments includes extensions of [11] to non-local stability properties [13] and to the area of gradient descents with finite-time convergence [3, 12]. The stability analysis in all these approaches seek an average system to guarantee convergence to a neighborhood of the extremum. The application of averaging, however, requires time-scale separation and thus causes delay in the DE.

Regarding the steady-state error due to continuous perturbation, it is shown in [10] that, small dither amplitudes will reduce the steady-state error. The tuning of the dither amplitude, hence, introduces a trade-off between the signal-to-noise ratio and the achievable steady-state error bound. Hence, multiple ES schemes are designed with the aim to deliver asymptotic convergence to the true optimizer by reducing the perturbation amplitude while converging to the optimum. For example, [15] proposed an extension to the classical ES scheme by adapting the amplitude of the dither signal based on the estimated derivative. However, in [1] it is shown that, this extended scheme does not posses a single equilibrium, but rather an equilibrium manifold, meaning that, if the scheme starts in, or close to, this manifold, it can never converge to the true optimum.

An extension of the scheme presented in [10] to remove the steady-state error is proposed in [4], which includes a convex uncertainty estimation set with a radius that con-

[1]Martijn Kortenhoeven and Thijs van Keulen are with the Eindhoven Univ. of Tech., Dep. of Mechanical Engineering, Control Systems Technology group, The Netherlands. t.a.c.v.keulen(at)tue.nl

[2]Thijs van Keulen is also with ASML, Veldhoven, The Netherlands.

verges to zero. Subsequently, the perturbation signal amplitude gets reduced recursively such that it roughly equals the radius of the convex uncertainty set. While this scheme can achieve precise convergence to the true optimizer of a measured unknown objective function, the scheme is still subject to the undesired delay in the DE and trade-off in the presence of measurement noise due to its dependency on perturbation amplitude reduction.

In sharp contrast to the classical ES scheme, a Moving Average (MA)-based DE framework is introduced in [5]. In this DE implementation, MA filtering is applied to the measured cost output modulated with the perturbation signal. Simulations in [5] indicate that the MA design leads to an improved performance with respect to a comparable ES controller with a low-pass filter. In [16], a generalization to both multiple-input and higher-order derivatives of this MA filtered DE is provided. Besides the modulation-based DE, the MA filter framework can also apply iterative DE by the local fitting of a polynomial functional on the measured input-output data. A linear Least-Squares (LS) fitting framework is proposed in [6]. It is shown in [14] that, the linear LS filter framework is equivalent to the modulation-based MA filter implementation.

In conclusion, the MA filter DE demonstrates significant improvements in transient performance over the classic DE implementation. However, until now, an analysis of the faster convergence has not been presented. Moreover, an analysis of the steady-state error dynamics of the MA-based DE implementation is not known to the authors. Also, the fundamental trade-off between signal-to-noise ratio and steady-state error is not removed by both classic or MA filtered ES.

The first contribution of this paper is to provide a stability and error analysis that shows that MA filter based DE does not require the application of averaging and hence avoids time delay in the DE. Also, it is shown that the steady-state error of the MA filter framework based on linear LS approaches the same steady-state error limit as classic ES. A second contribution of this paper is to extend the DE based on first-order LS with a MA filter implementation to higher-order polynomial fitting in a LS framework. In sharp contrast to existing ES methods, a high-order polynomial fit reduces the negative effect on the convergence efficiency caused by an increased perturbation amplitude. Finally, a third contribution of this paper is to provide a numerical case study that compares different ES schemes regarding their convergence speed and steady-state error for non-symmetric objective functions in the presence of measurement noise.

This paper is organized as follows. In Section II the problem is formulated. Section III gives an overview of the DE schemes including the novel high-order LS DE. Next, in Section IV, a numerical case study is presented. Finally, Section V provides conclusions.

II. PROBLEM FORMULATION

Consider the optimization problem

$$\min_{u\in\mathbb{R}} J(u,t), \tag{1}$$

with the unknown static cost functional:

$$J(u,t) = Q_J(u) + w(t). \tag{2}$$

Here, $J(u,t)$ is the measured cost as a function of the control input $u \in \mathbb{R}$, and $w(t) \in \mathbb{R}$ is the measurement disturbance at time $t \in \mathbb{R}_{\geq 0}$, and $Q_J : \mathbb{R} \to \mathbb{R}$ an unknown nonlinear static mapping. The white noise signal w is assumed to have zero mean and a constant power spectral density.

The goal of the ES framework is to let the input $u(t)$ converge to the optimizer of (1), i.e.:

$$u(t) \to u^* = \arg\min_{u\in\mathbb{R}} Q_J(u), \text{ when } t \to \infty. \tag{3}$$

To ensure convergence of the optimization process, the following assumption is required.

Assumption 1 *The mapping $Q_J : \mathbb{R} \to \mathbb{R}$ is N, $N \geq 2$ times continuously differentiable with respect to the control input u and there exists a constant input $u^* \in \mathbb{R}$ such that the gradient equals*

$$\tilde{g}(u) := \frac{\mathrm{d}Q_J(u)}{\mathrm{d}u} = 0, \tag{4}$$

if and only if $u = u^*$. Moreover

$$\frac{\mathrm{d}^2 Q_J(u)}{\mathrm{d}u^2} > 0, \text{ for all } u \in \mathbb{R}. \tag{5}$$

Assumption 1 ensures that $Q_J(u^*)$ is a global minimum and u^* is the unique minimizer of Q_J.

III. EXTREMUM SEEKING FRAMEWORK

This section outlines three ES implementations: classical DE, MA based DE, and LS based DE. Figure 1 provides a generic schematic overview of an ES framework.

Sinusoidal perturbation of the system input is assumed for each ES framework

$$d(t) = a\cos(\omega t), \tag{6}$$

in which, $a > 0$ is the dither amplitude, and ω the frequency. The dither signal d is added to the input of the performance map, leading to

$$u(t) = \hat{u}(t) + d(t), \tag{7}$$

where, $\hat{u}$ is the output of the optimization algorithm. Given that the optimizer u^* of Q_J applies and $\hat{u}$ is the actual output of the optimizer at time instance t, the optimization error is defined as:

$$\tilde{u}(t) := \hat{u}(t) - u^*. \tag{8}$$

The objective is to adapt input u such that the steady-state cost Q_J is minimized, i.e., to let $\tilde{u}(t) \to 0$ if $t \to \infty$. As such, the cost Q_J can be minimized using a continuous steepest-descent approach, where the local gradient of Q_J with respect to u is estimated and this function value is integrated, i.e., the differential equation

$$\dot{\hat{u}}(t) = -k\tilde{g}(t), \tag{9}$$

applies, where $k > 0$ is the optimizer gain, and $\tilde{g}$ is the gradient estimate at time instance t.

In the following subsections the different approaches to obtain $\tilde{g}$ are discussed. These approaches are later compared in a numerical case study.

A. Classical derivative estimation

Figure 2 depicts the DE of a classical ES scheme as given in [10]. This scheme employs a high-pass filter to remove the static contribution $Q_J(u)$ from $J(u,t)$ and thereby attempts to avoid distortion of the demodulation. For example, a first-order high-pass filter can be applied

$$\mathcal{H}_{HP}(s) = \frac{s}{s+\omega_{HP}}, \tag{10}$$

where, s is the Laplace variable and $\omega_{HP} > 0$ the tune-able filter pole.

Next, the high-pass filtered signal is demodulated to locally estimate the objective function's gradient. Subsequently, a low-pass filter is included to smooth the behavior of $\hat{u}$. For example, a first-order low-pass filter can be applied:

$$\mathcal{H}_{LP}(s) = \frac{\omega_{LP}}{s+\omega_{LP}}. \tag{11}$$

Here, $\omega_{LP} > 0$ is the tune-able filter pole. The stability analysis in [10] requires ω_{HP} and ω_{LP} small compared to dither frequency ω. Hence, the DE takes the following form

$$\dot{\tilde{g}}_{cl}(t) = \omega_{LP}(-\tilde{g}_{cl}(t) + (J(u,t) - \eta(t))a\cos(\omega t)), \tag{12}$$
$$\dot{\eta}(t) = \omega_{HP}(-\eta(t) + J(u,t)), \tag{13}$$

where η is the approximated static contribution of $Q_J(\hat{u})$. The estimated gradient $\tilde{g}_{cl}$ is a scaled version of the true gradient. For the gradient estimate $\tilde{g}_{cl}$ to be in the correct scale, $\tilde{g}_{cl}$ should be divided by $\frac{a^2}{2}$. However, one could choose to correct this deviation in the integrator gain k. Moreover, due to the oscillating behavior of $J(u,t)$, η in (13) is unable to track $J(u,t)$, i.e. $J(u,t) - \eta(t) \neq 0$. As a consequence, $\tilde{g}_{cl}(t)$ in (12) shows oscillating behavior and, therefore, compromises the DE of $J(u,t)$.

It is shown in [10] that, the steady-state error $\tilde{u}$ in (8) is approximated by

$$\tilde{u} \approx -\frac{Q_J'''(u^*)}{8Q_J''(u^*)}a^2, \tag{14}$$

in which, $Q_J''(u^*)$ and $Q_J'''(u^*)$ denote the second and third derivative of Q_J with respect to u at the optimizer u^* respectively. This error bound shows that, since $a > 0$, the error can not converge to zero when the third derivative of Q_J with respect to the input u is not equal to zero.

The approach can be generalized to higher-order derivatives, see [11]. Here, DE of Q_J are obtained by the application of a bank of low-pass filters. Let the function $g_{lp}(a,\eta)$ provide the DE

$$g_{lp}(a,\eta) := A_N^{-1}A_\alpha^{-1}A^{-1}\eta, \tag{15}$$

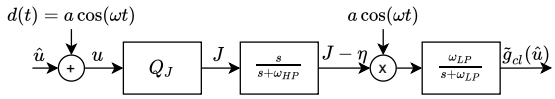


Fig. 2: Derivative estimation of a classical ES scheme.

with the matrices A_N, A_α, and A defined by:

$$A := \operatorname{diag}\begin{pmatrix}1 & a & a^2 & \cdots & a^N\end{pmatrix}, \tag{16}$$

$$A_\alpha := \begin{bmatrix}\alpha_0 & \alpha_1 & \cdots & \alpha_N \\ \alpha_1 & \alpha_2 & \cdots & \alpha_{N+1} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_N & \alpha_{N+1} & \cdots & \alpha_{2N}\end{bmatrix}, \tag{17}$$

$$A_N := \operatorname{diag}\begin{pmatrix}\frac{1}{0!} & \frac{1}{1!} & \frac{1}{2!} & \cdots & \frac{1}{N!}\end{pmatrix}. \tag{18}$$

Here, N is the highest derivative order, and α indicates the static contribution of each entry in η to the derivatives in g_{lp}. Opposed to the implementation (12) and (13), this implementation estimates the gradient with the correct scale.

B. Moving average based derivative estimation

Although the ES scheme presented in [10, 11] is intuitive and well analyzed, applying an MA filter instead of a high-pass filter provides more accurate DE and, therefore, can achieve faster convergence. See Fig. 3 for a depiction of the first-order DE implementation using MA filtering. This section provides a frequency domain analysis that this scheme obtains more accurate DE. Furthermore, an analysis demonstrates that convergence is guaranteed without time-scale separation between DE filter frequency and dither frequency.

Using Assumption 1, and for constant $\hat{u}$, a Taylor approximation of the steady-state response of Q_J is given by

$$\hat{Q}_J(\hat{u}+d(t)) = \sum_{r=0}^{N}\left(\frac{1}{r!}\left(d(t)D_u\right)^r Q_J(\hat{u})\right) + R_N, \tag{19}$$

in which R_N is the remainder term and the derivative of Q_J with respect to the input argument is defined by:

$$D_u^r := \left(\frac{\partial}{\partial u}\right)^r = \frac{\partial^r}{\partial u^r}. \tag{20}$$

It is convenient to rewrite (19) as

$$\hat{Q}_J(\hat{u}+d(t)) = m^\top(t)AA_N g_{ma}(\hat{u}) + R_N, \tag{21}$$

where the harmonics with the dither frequency up to order N are collected in vector $m(t) = [1, \cos(\omega t), \cos^2(\omega t), \ldots, \cos^N(\omega t)]$ and the constant matrices A and A_N, as defined in (16) and (18), collect the dither amplitude information.

Next, consider demodulation by multiplication of the measured cost with signal vector $m(t)$ and subsequent application of an MA filter:

$$\int_{t-T}^{t} m(\tau)\hat{Q}_J(\hat{u}+d(\tau))\mathrm{d}\tau = \int_{t-T}^{t} m(\tau)m^\top(\tau)AA_N g_{ma}(\hat{u})\mathrm{d}\tau + \int_{t-T}^{t} m(\tau)R_N\,\mathrm{d}\tau. \tag{22}$$

Rearranging of (22) provides the DE

$$g_{ma} \approx K_{\text{MA}}^{-1} \int_{t-T}^{t} m(\tau)\hat{Q}_J(\hat{u} + d(\tau))\, \mathrm{d}\tau, \tag{23}$$

in which $K_{\text{MA}} = \int_{t-T}^{t} m(\tau)m^{\top}(\tau)AA_N \ \mathrm{d}\tau$. Note that, the matrix $m(t)m^{\top}(t)$ is singular by construction unless the product is integrated over the perturbation time period $T = \frac{2\pi}{\omega}$. Hence, K_{MA} is full rank if $T = \frac{2\pi}{\omega}$.

1) Frequency domain analysis: Inspired by the work in [5, 7, 9], the higher accuracy of MA based DE over classic DE is studied in the context of frequency-based system identification. Using the fact that $Q_J(u)$ is approximately linear on the interval $u : [\hat{u} - a, \hat{u} + a]$ for small a and using the common ES assumption that k is sufficiently small such that $\hat{u}$ can be viewed as a constant, the signal $Q_J(u(t))$ can be approximated as follows:

$$\tilde{Q}_J(u(t)) \approx Q_J(\hat{u}) + \tilde{g}_{ma} a \cos(\omega t). \tag{24}$$

Using the complex exponential Fourier transform the signal $\tilde{Q}_J(u(t))$ can be transformed into

$$\tilde{Q}_J(u(t)) = \sum_{n=-\infty}^{\infty} C_n\, e^{\frac{j2\pi nt}{T}}, \tag{25}$$

where, $T = \frac{2\pi}{\omega}$ and:

$$\begin{aligned} C_n = &\underbrace{\frac{1}{T}\int_{-T/2}^{T/2} \tilde{Q}_J(u(t)) \cos\left(\frac{2\pi nt}{T}\right) \mathrm{d}t}_{\mathrm{Re}(C_n)} \\ &- \underbrace{\frac{j}{T}\int_{T/2}^{T/2} \tilde{Q}_J(u(t)) \sin\left(\frac{2\pi nt}{T}\right) \mathrm{d}t}_{\mathrm{Im}(C_n)} \ \forall n \in \mathbb{Z}. \end{aligned} \tag{26}$$

Here, the coefficients are ordered according to:

$$C_n = \begin{cases} Q_J(\hat{u}) & \text{for } n = 0, \\ \frac{1}{2}\tilde{g}_{ma} a & \text{for } n = \pm 1, \\ 0 & \text{for } |n| \geq 2. \end{cases} \tag{27}$$

Note that, the Fourier series coefficients C_n in (26) only produce real-valued constants. Therefore, it can be concluded that, the imaginary part $\mathrm{Im}(C_n)$ in (26) is zero for all n and, that as a result, (26) can be simplified to:

$$C_n = \frac{1}{T}\int_{-T/2}^{T/2} \tilde{Q}_J(u(t)) \ \cos\left(\frac{2\pi nt}{T}\right) \mathrm{d}t, \ \forall n \in \mathbb{Z}. \tag{28}$$

Using this insight, and provided that $\tilde{Q}_J(u(t))$ has the form (24), it can be shown that the scheme presented in Fig. 3 is able to extract the true gradient of Q_J at $\hat{u}$ and the true constant $Q_J(\hat{u})$ and it is able to do so on an arbitrarily short time interval $T = \frac{2\pi}{\omega}$ provided that an integer number of samples n fit in the time interval T. As such, it is concluded that the DE is more accurate than the classical DE, for which the DE exhibits oscillating behavior. Consequently, MA based DE leads to better transient behavior and is able to achieve faster convergence compared to the classical DE.

2) Stability and error analysis: of an ES scheme adopting an MA filter for first-order DE does not seem to appear in literature and is presented here. In this context, consider the MA based ES scheme described in Fig. 3 for which the DE takes the form

$$\dot{\hat{u}} = -\frac{2k}{aT}\int_{t-T}^{t} Q_J(\hat{u} + a\cos(\omega\tau)) \ \cos(\omega\tau)\, \mathrm{d}\tau, \tag{29}$$

with $T = \frac{2\pi}{\omega}$. In contrast to [10, 11], the application of the MA filter provides a closed-form solution for the perturbed system. Note that, the system description is similar to the average system description in [8, 10], but, crucially, without the need for an approximate solution hinging on time-scale separation. Now implementing the coordinate transformation (8), the ES dynamics are described with:

$$\dot{\tilde{u}} = -\frac{2k}{aT}\int_{t-T}^{t} Q_J(\tilde{u} + u^* + a\cos(\omega\tau)) \ \cos(\omega\tau)\, \mathrm{d}\tau. \tag{30}$$

This system has an equilibrium $\tilde{u}_{\text{e}}$ that satisfies:

$$\int_{t-T}^{t} Q_J(\tilde{u}_{\text{e}} + u^* + a\cos(\omega\tau)) \cos(\omega\tau)\, \mathrm{d}\tau = 0. \tag{31}$$

Note that, since $Q_J(u^*)$ is a constant and for $T = \frac{2\pi}{\omega}$, that $\int_{t-T}^{t} -Q_J(u^*)\cos(\omega\tau)\, \mathrm{d}\tau = 0$. Hence, define the error function

$$v(\tilde{u} + a\cos(\omega t)) = Q_J(\tilde{u} + u^* + a\cos(\omega t)) - Q_J(u^*), \tag{32}$$

and rewrite (31) to:

$$\int_{t-T}^{t} v(\tilde{u}_{\text{e}} + a\cos(\omega\tau)) \cos(\omega\tau)\, \mathrm{d}\tau = 0. \tag{33}$$

By Assumption 1, this implies that $v(0) = 0$ and $v'(0) = 0$. Also, following the approach in [10] assuming that $\tilde{u}_{\text{e}}$ can be approximated by a polynomial of the form $\tilde{u}_{\text{e}}(a) = b_1 a + b_2 a^2 + R_n(a)$, where a is the dither amplitude and approximating $v(\tilde{u}_{\text{e}} + a\cos(\omega t))$ by a Maclaurin series, using the time-scale $\sigma = \omega t$ and selecting $t = T$, giving:

$$\begin{aligned} &\int_0^{2\pi} v(\tilde{u}_{\text{e}} + a\cos\sigma)\cos\sigma\, \mathrm{d}\sigma = \\ &\qquad \int_0^{2\pi} \frac{v''(0)(b_1 a + b_2 a^2 + a\cos\sigma)^2 \cos\sigma\, \mathrm{d}\sigma}{2} \\ &+ \int_0^{2\pi} \frac{v'''(0)(b_1 a + b_2 a^2 + a\cos\sigma)^3 \cos\sigma\, \mathrm{d}\sigma}{6} = 0. \end{aligned} \tag{34}$$

Due to the presence of $\cos\sigma$, multiple terms in (34) equal zero, resulting in:

$$\begin{aligned} \int_0^{2\pi} v(\tilde{u}_{\text{e}} + a\cos\sigma)\cos\sigma\, \mathrm{d}\sigma &= \pi v''(0)(b_1 a^2 + b_2 a^3) \\ &+ \frac{\pi v'''(0)(b_1 + \frac{3}{4})a^3}{6} + R_n(a) = 0. \end{aligned} \tag{35}$$

This concludes to $b_1 = 0$ and $b_2 = -\frac{v'''(0)}{8v''(0)}$, such that:

$$\tilde{u}_{\text{e}}(a) = -\frac{v'''(0)}{8v''(0)}a^2 + R_n(a). \tag{36}$$

So, to summarize, the classic ES presented in Section III-A and the MA filter ES presented in Section III-B, share the same approximated steady-state error bound, i.e. (14).

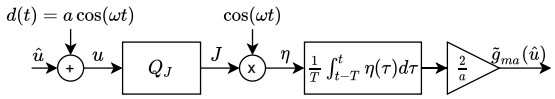


Fig. 3: Derivative estimation of a moving average filter.

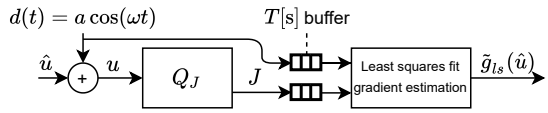


Fig. 4: Derivative estimation by recursive LS fitting.

Next, it is shown that the structure of the well-known stability analysis for the classical ES is also applicable to MA filtered ES.

Heretofore, rewrite the error dynamics (30) to

$$\dot{\tilde{u}} = -\frac{2k}{aT}\int_{t-T}^{t} v(\tilde{u} + a\cos(\omega\tau))\cos(\omega\tau)\,\mathrm{d}\tau, \tag{37}$$

that has the Jacobian evaluated at $\tilde{u}_{\mathrm{e}}$:

$$\mathcal{J} = -\frac{2k}{aT}\int_{t-T}^{t} v'(\tilde{u}_{\mathrm{e}} + a\cos(\omega\tau))\cos(\omega\tau)\,\mathrm{d}\tau. \tag{38}$$

This Jacobian $\mathcal{J}$ will be Hurwitz if and only if:

$$\int_{t-T}^{t} v'(\tilde{u}_{\mathrm{e}} + a\cos(\omega\tau))\cos(\omega\tau)\,\mathrm{d}\tau > 0. \tag{39}$$

Again using that $v'(0) = 0$ and also that $v''(0) > 0$ and performing similar Maclaurin approximation as in (34), it can be concluded that:

$$\int_{t-T}^{t} v'(\tilde{u}_{\mathrm{e}} + a\cos(\omega\tau))\cos(\omega\tau)\,\mathrm{d}\tau = \pi v''(0)a + R_n(a^2). \tag{40}$$

As such, it can be concluded that the equilibrium $\tilde{u}_{\mathrm{e}}$ is asymptotically stable for sufficiently small a.

C. Least squares based derivative estimation

A third implementation for DE in ES is given by [6], such that, by using a first-order LS polynomial fit to locally estimate the performance map, a DE can be achieved by taking the gradient of the fitted polynomial, see Fig. 4. Consider the following LS objective function

$$W_{LS}(t) = \int_{t-T}^{t}\left(Q_J(u(\tau)) - [c_0(t)\ c_1(t)\dots c_N(t)]\begin{bmatrix}1\\ d(\tau)\\ \vdots\\ d^N(\tau)\end{bmatrix}\right)^2 \mathrm{d}\tau, \tag{41}$$

with, $N \geq 1$ the selected polynomial order, $d(t)$ the perturbation signal according to (6) and $T = \frac{2\pi}{\omega}$, for the following optimization problem:

$$\min_{c_0, c_1, c_2, \dots, c_N} W_{LS}. \tag{42}$$

If the parameters $c_0, c_1, c_2, \dots$ are optimal, then the following conditions should hold: $\frac{\partial W_{LS}}{\partial c_0} = 0$, $\frac{\partial W_{LS}}{\partial c_1} = 0$, etc..

Next, the polynomial $p(u) = c_0 + c_1 u + \dots + c_N u^N$ is fitted using the input- and output data in the time interval $[t-T, t]$. Taking the common assumption in ES that the integrator gain k in the optimizer (9) is sufficiently small, this polynomial locally approximates the performance map Q_J on the interval $[\hat{u} - a, \hat{u} + a]$ shifted by $-\hat{u}$, where a is the dither amplitude, i.e. $p(e) \approx Q_J(\hat{u} + e)$ for $-a \leq e \leq a$. Subsequently, the DE is obtained by:

$$\tilde{g}_{ls} = \frac{dp}{du}(0) = c_1. \tag{43}$$

Note that, $p(0) \approx Q_J(\hat{u})$ implies that $\tilde{g}_{ls} \approx \frac{dQ_J}{du}(\hat{u})$.

The equivalence between the LS based and MA based DE is shown. Revisit the cost function (41). Rewriting $c(t) = [c_0(t), c_1(t), c_2(t), \dots, c_N(t)]^{\mathsf{T}}$ and $Am(\tau) = [d_0(\tau), d_1(\tau), d_2(\tau), \dots, d_N(\tau)]^{\mathsf{T}}$ and taking out the constant A matrix defined in (16), the unique minimizing argument c^* to this cost function is

$$c^* = A^{-1}\underbrace{\left(\int_{t-T}^{t} m(\tau)m(\tau)^{\mathsf{T}}\,\mathrm{d}\tau\right)^{-1}}_{K_{\mathrm{LS}}^{-1}}\int_{t-T}^{t} Q_J(u(\tau))m(\tau)\,\mathrm{d}\tau, \tag{44}$$

where the matrix K_{LS} is a constant Hankel matrix if and only if the integration time $T = \frac{2\pi}{\omega}$. As such, equivalence between the MA DE and LS DE is demonstrated and a generalization of the LS DE from [16] is obtained.

Fitting an Nth order polynomial allows for N^{th} order DE of Q_J making it possible to use, e.g., Newton based optimization. However, higher-order polynomial fitting is also advantageous in an ES scheme employing steepest-descent optimization, which is the focus of this paper. In particular, since the polynomial order may affect the optimum polynomial coefficients, employing a higher-order fit with $N > 2$ can result in more accurate first-order DE in case of a performance map which is non-symmetric around its extremum. However, an even polynomial order does not yield an advantage compared to the previous odd order. This is due to the following.

Matrix K_{LS} in (44) is a Hankel matrix in which each element $K_{i,j}$ in row i and column j is larger than zero if $i+j$ is an even number and equals zero if $i+j$ is an odd number. The inverse of K_{LS} has the same structure. Since only the second row of K, i.e. $K_{\mathrm{LS}_{2,[1,2,\dots,N]}} \in \mathbb{R}^{1\times N}$ contributes to the first order DE $\tilde{g}_{ls} = c_1$ in (43), rather than all rows in K_{LS}, only odd polynomial orders increase the accuracy of the polynomial fit. This also shows why a higher-order polynomial fit is more accurate than a first-order fit.

IV. NUMERICAL CASE STUDY

In this section, a case study is provided to compare the performance of the DE schemes presented in Section III. For each scheme, both the transient- and steady-state behavior is evaluated for the objective function $J(u,t) = -u\sin u + w(t)$ with initial estimate $\hat{u}(t_0) = 1.5$ at time $t_0 = 0$. This static map has an extremum at $u^* \approx 2.0288$ and is non-symmetric around the optimizer since $J(u^*+e) \neq J(u^*-e)$ for $0 < e \ll 1$.

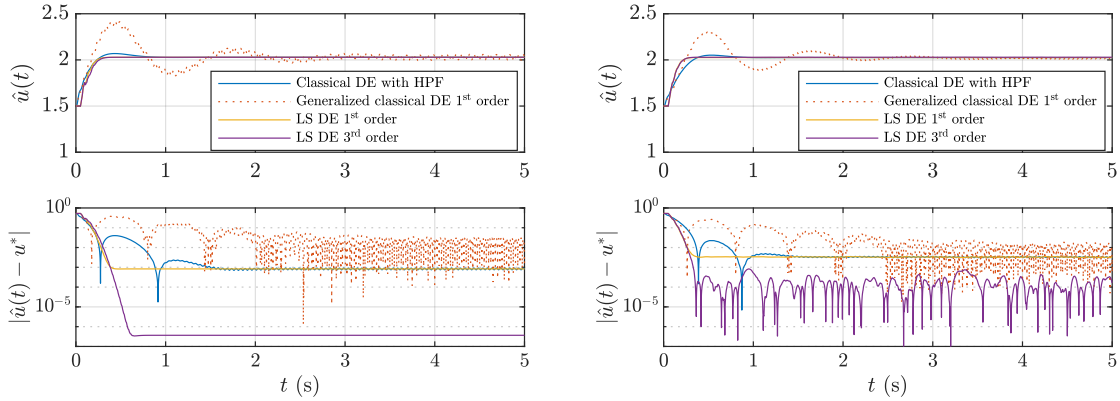


Fig. 5: Results of the investigated ES schemes: (left) without measurement noise, and (right) with measurement noise.

TABLE I: Extremum seeking parameters for simulations.

	Without noise			With noise					
	a	ω	k	a	ω	k	ω_{LP}	ω_{HP}	T
Cl DE 1	0.1	40π	350	0.2	40π	100	10	20	-
Cl DE 2	0.1	40π	5	0.2	40π	5	2.5	-	-
LS DE	0.1	40π	5	0.2	40π	5	-	-	0.05

Each ES scheme is implemented with a sampling frequency F_s = 1000 Hz, both with and without measurement noise $w(t)$, see Fig. 1. The measurement noise is white Gaussian noise, i.e. zero mean with constant power spectral density, with variance $\sigma_w^2 = 1 \cdot 10^{-7}$ and is a pre-generated time series, such that the same disturbance is applied to each ES scheme as to create a fair comparison.The selected ES parameters are provided in Table I and inspired by the results provided in the corresponding cited articles, see Section III.

A. Classical derivative estimation

The classical DE has been implemented according to both the scheme represented in Fig. 2 (Cl DE 1) and the generalized classical DE (Cl DE 2) as described in Section III-A. For Cl DE 1, the first derivative of $Q_J(\hat{u})$ is estimated using both a high-pass filter and a low-pass filter. It is emphasized that, as discussed in Section III-A, the DE is scaled, resulting in a differently tuned integrator gain k. For Cl DE 2, only low-pass filtering is applied. As a consequence of these filters, the DE is inaccurate when comparing it to the LS DE implementation. Moreover, the classical DE shows steady state oscillations around the approximated error according to (14). In the presence of measurement noise, the scheme provides robustness to these disturbances. However, since the steady-state error is, according to (14), a function of the dither amplitude, the classical DE implementation is unable to converge to the true optimum in the presence of measurement noise.

B. Least-Squares based derivative estimation

The LS DE is implemented with different polynomial orders N. As is shown in Section III, the first-order LS scheme converges to the same approximation error as the classical DE, however faster. Selecting a higher polynomial order allows for a better fit of the input to output data and, in turn, allows for a more accurate DE. As becomes evident from Fig. 5, choosing a higher-order polynomial results in a reduced approximation error. In the presence of measurement noise, this allows for a higher dither amplitude, while the DE remains more accurate than the classical DE.

V. CONCLUSIONS

This article advocates derivative estimation (DE) for extremum seeking (ES) based on recursive high-order least-squares (LS) fitting implemented with a receding horizon equal to the perturbation period. It is demonstrated through analysis and a numerical case study that this DE provides faster convergence compared to classical ES while is able to achieve superior steady-state accuracy on non-symmetric objective functions in the presence of measurement noise. Future work will focus on enhancing the convergence speed further by exploiting self excitation of the converging ES loop.

REFERENCES

[1] K.T. Atta and M. Guay. "Comment on "On stability and application of extremum seeking control without steady-state oscillation" [Automatica 68 (2016) 18–26]". In: *Automatica* 103 (2019), pp. 580–581.

[2] E. Dinçmen and T. Altinel. "An emergency braking controller based on extremum seeking with experimental implementation". In: *International Journal of Dynamics and Control* (2018).

[3] M. Guay. "Finite-time extremum seeking control for a class of unknown static maps". In: *International Journal of Adaptive Control and Signal Processing* 35 (June 2020).

[4] M. Guay. "Uncertainty Estimation in Extremum Seeking Control of Unknown Static Maps". In: *IEEE Control Systems Letters* 5.4 (Oct. 2021), pp. 1115–2021.

[5] M. Haring, N. van de Wouw, and D. Nešić. "Extremum-seeking control for nonlinear systems with periodic steady-state outputs". In: *Automatica* 49 (2013), pp. 1883–1891.

[6] B.G.B. Hunnekens, M.A.M. Haring, N. van de Wouw, and H. Nijmeijer. "A dither-free extremum-seeking control approach using 1st-order least-squares fits for gradient estimation". In: *53rd IEEE Conference on Decision and Control*. 2014, pp. 2679–2684.

[7] T. van Keulen, R. van der Weijst, and T. Oomen. "Fast extremum seeking using multisine dither and online complex curve fitting". In: *Proceedings of the 21st IFAC World Congress*. 2020, pp. 5362–5367.

[8] H. Khalil. *Nonlinear Systems*. Pearson, 2014.

[9] D. Krishnamoorthy. "On the design and analysis of multivariable extremum seeking control using fast fourier transform". arXiv preprint arXiv:2104.14365.

[10] M. Krstić and H.-H. Wang. “Stability of extremum seeking feedback for general nonlinear dynamic systems”. In: *Automatica* 36 (2000), pp. 595–601.

[11] D. Nešić, Y. Tan, C. Manzie, A. Mohammadi, and W. Moase. “A unifying framework for analysis and design of extremum seeking controllers”. In: *2012 24th Chinese Control and Decision Conference (CCDC)*. 2012, pp. 4274–4285.

[12] J.I. Poveda and M. Krstić. “Fixed-Time Gradient-Based Extremum Seeking”. In: *2020 American Control Conference (ACC)*. 2020, pp. 2838–2843.

[13] Y. Tan, D. Nešić, and I. Mareels. “On non-local stability properties of extremum seeking control”. In: *Automatica* 42.6 (2006), pp. 889–903.

[14] R. van der Weijst. “Extremum seeking for robust fuel-efficient control of diesel engines”. PhD thesis. Eindhoven University of Technology, Nov. 2019.

[15] L. Wang, S. Chen, and K. Ma. “On stability and application of extremum seeking control without steady-state oscillation”. In: *Automatica* 68 (2016), pp. 18–26.

[16] R. van der Weijst, T. van Keulen, and F. Willems. “A generalized framework for perturbation-based derivative estimation in multivariable extremum-seeking”. In: *Proceedings of the 20th IFAC World Congress*. 2017, pp. 7836–7841.