THESIS

# SOME EFFICIENT OPEN-LOOP CONTROL SOLUTION STRATEGIES FOR DYNAMIC OPTIMIZATION PROBLEMS AND CONTROL CO-DESIGN

Submitted by

Athul Krishna Sundarrajan

Department of Systems Engineering

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Summer 2021

Master's Committee:

Advisor: Dr. Daniel R. Herber

Dr. James Cale
Dr. Karan Venayagmoorthy

ABSTRACT

SOME EFFICIENT OPEN-LOOP CONTROL SOLUTION STRATEGIES FOR DYNAMIC OPTIMIZATION PROBLEMS AND CONTROL CO-DESIGN

This thesis explores strategies to efficiently solve dynamic optimization (DO) and control co-design (CCD) problems that arise in early-stage system design studies. The task of design optimization of dynamic systems involves identifying optimal values of the physical elements of the system and the inputs to effectively control the dynamic behavior of the system to achieve peak performance. The problem becomes more complex when designing multidisciplinary systems, where the coupling between disciplines must be accounted for to achieve optimal performance. Developing tools and strategies to efficiently and accurately solve these problems is needed.

Conventional design practices involve sequentially optimizing the plant parameters and then identifying a control scheme for the given plant design. This sequential design procedure does not often produce system-level optimal solutions. Control co-design or CCD is a design paradigm that seeks to find system-level optimal design through simultaneous optimization of the plant and control variables. In this work, both the plant and controls optimization are framed as a integrated DO problem. We focus on a class of direct methods called direct transcription (DT) to solve these DO problems.

We start with a subclass of nonlinear dynamic optimization (NLDO) problems for the first study, namely linear-quadratic dynamic optimization problems (LQDO). For this class of problems, the objective function is quadratic, and the constraints are linear. Highly efficient and accurate computational tools have been developed for solving LQDO problems on account of their linear and quadratic problem elements. Their structure facilities the development of automated solvers. We identify the factors that enable creating these efficient tools and leverage them towards

solving NLDO problems. We explore three different strategies to solve NLDO problems using LQDO elements, and analyze the requirements and limits of each approach.

Though multiple studies have used one of the methods to solve a given CCD problem, there is a lack of investigations identifying the trade-offs between the nested and simultaneous CCD, two commonly used methods. We build on the results from the first study and solve a detailed active suspension design using both the nested and simultaneous CCD methods. We look at the impact of derivative methods, tolerance, and the number of discretization points on the solution accuracy and computational times. We use the implementation and results from this study to form some heuristics to choose between simultaneous and nested CCD methods.

A third study involves CCD of a floating offshore wind turbine using the levelized cost of energy (LCOE) as an objective. The methods and tools developed in the previous studies have been applied toward solving a complex engineering design problem. The results show that the impact of optimal control strategies and the importance of adopting an integrated approach for designing FOWTs to lower the LCOE.

# ACKNOWLEDGEMENTS

At the outset, I would like to express my gratitude to Dr. Daniel Herber for giving me the opportunity to work on these fascinating topics and for his guidance and advice in navigating this field. His passion and the attention to detail he has towards his work has been a constant source of inspiration for me.

I want to thank my family for their endless support and encouragement. I am also thankful to Jayesh, Sanket, Muhundan, Tater, Robbie, Ryan, and the community at Wild Horizons for the wonderful memories.

Finally, I would like to acknowledge Dr. Alan Wright, Dr. Dan Zalkind, Dr. Garett Barter, and the team at NREL and Dr. Yong Hoon Lee from UIUC for their valuable feedback and suggestions in the final case study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

# Chapter 1

# Introduction

There is an increasing demand for advanced dynamic multidisciplinary systems that can satisfy tight performance requirements and exist and operate in adverse environments. A floating offshore wind turbine (FOWT), aircraft thermal management system, and the drive train of electric vehicles are examples of such systems [4–6]. The task of designing these systems involves identifying optimal physical parameters of the system and an effective control strategy to satisfy the performance requirements and ensure these systems' safety in such environments [7].

Because of the complex nature of these systems, the optimal design is often non-obvious and identifying the optimal design requires non-conventional methods [8]. Also, conventional design practices often do not take into account the dynamic nor the multidisciplinary nature of these systems. Design optimization is the process of identifying the optimal design by using numerical optimization techniques on a mathematical model of the system [9]. This model of the system is commonly developed using first-principle methods. For controls, the model can be developed using system-identification methods. The objective or utility function for the optimization problem can represent the quantities that must be minimized or maximized. This objective function is subject to constraints that represent physical behavior, functional limits, time, resources, and failure modes [10].

Multidisciplinary design optimization (MDO) provides a framework to combine each discipline's analysis and design considerations by taking into account the coupling that exists between them. Traditional application of MDO is to design static systems. Thus, the numerical optimization problem solved for MDO is a static optimization problem [11]. Though MDO can be used to design dynamic systems, the complications that arise due to the coupling between time-dependent quantities can be a challenge. Multidisciplinary dynamic system design optimization (MDSDO) is a framework defined by Allison and Herber [11]. As defined by the authors, MDSDO:

> "Deals with systems where the evolution of system state through time is a critical element of performance; where multiple disciplines, energy domains, models or sub-systems must be integrated; and where the unique properties of dynamic systems are exploited to improve system performance and yield efficient problem solutions."

To capture the dynamic behavior of the system, a family of ordinary differential equations is commonly used. Black box simulations of the system dynamics are a common alternative to ODEs. The constraints that are placed on the system can be simple upper and lower bound constraints to limit the range of the trajectories or complex algebraic constraints to impose specific trajectories that the system states must follow. The goals of the system are expressed in terms of an objective function. The resulting MDSDO problem formulations are dynamic optimization problems. When these problems involve finding a control strategy, they are also known as optimal control problems. Thus, the task of design optimization of dynamic systems involves solving a dynamic optimization problem that is subject to a set of constraints on the system's behavior. The problem becomes more complex when designing multidisciplinary systems, where the coupling between disciplines must be accounted for to achieve optimal performance. Efficient tools and strategies to accurately solve these problems are needed.

The primary application of these integrated design methods has been on structural design applications, but recently, the applicability of these methods to design renewable energy systems has been identified by experts [6, 12, 13]. These integrated design methods have recently been applied to modern renewable energy systems like land-based wind turbines, hydrokinetic turbines, wave energy extractors, and airborne wind energy systems [14–17]. These renewable energy systems are highly dynamic, and there are strong interactions between the different aspects of these systems. As the size of these systems increases as the technology advances, the complexities of the interactions within the systems and between other systems in the same environment increases, such as grid integration, offshore deployment, and environmental impacts. To account for these considerations, a systems engineering approach needs to be adopted towards the design of these systems [18].

## 1.1 Dynamic System Design Optimization

Using optimization techniques to solve a given design problem involves reformulating the system's design goals and requirements into elements appropriate for inclusion in an optimization problem. Mathematically, the full problem is represented as:

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \tag{1.1a}$$

$$\text{subject to: } \boldsymbol{h}(\boldsymbol{x}) = \mathbf{0} \tag{1.1b}$$

$$\boldsymbol{g}(\boldsymbol{x}) \leq \mathbf{0} \tag{1.1c}$$

A given optimization problem has the following elements:

1. A set of variables that represent the different aspects of the system. These are also called design variables. Ultimately these are choices that the designer can make.

2. The objective function that represents overarching goals for the system.

3. The constraint equations that represent the inherent restrictions in the system and limits that are externally placed. There are two kinds of constraints, equality constraints, where the constraint equation must be equal to a given value, and inequality constraints where the constraint equation must be lesser than or greater than a given value.

Using an example of design of floating offshore wind turbine (FOWT), the design tasks are reformulated in terms of an optimization problem as follows:

1. Design variables for a FOWT design problem can be the tower height, tower thickness, blade length, and platform mass. The control variables for the problem are the generator torque and the blade pitch.

2. The objective function is to minimize the cost of the tower and maximize the power captured by this system.

3. The constraints can be for example to limit the tower bending moment, limit the rotor speed etc.

where $\boldsymbol{x}$ is the vector of design variables, $f$ denotes the objective function, and $(\boldsymbol{h}, \boldsymbol{g})$ represent the equality and inequality constraints, respectively.

Dynamic optimization problems have additional different elements as opposed to their static counterparts, and these elements are different because of their dynamic nature. The next section describes the elements in a dynamic optimization problem.

### 1.1.1 Dynamic Optimization Problem Elements

**Problem variables.**

In our treatment of dynamic systems and their design, we concentrate on the following kinds of variables:

1. Time is expressed as $t$ and describes the time horizon under consideration $t \in [t_0, t_f]$. Where $t_0$ describes the initial time and $t_f$ denotes the final time.

2. The state variables of the system, denoted by $\boldsymbol{\xi}(t)$. State variables are time dependent, and their first-order time derivative $\dot{\boldsymbol{\xi}}(t)$ describes the rate of change of system states. The states capture the system evolution over time.

3. Control or input variables to the system, denoted by $\boldsymbol{u}(t)$. These variables are traditionally time dependent, but they can have constant values as well as is the case of controller gains.

4. Plant variables denoted by $\boldsymbol{x}_p$ that describe the physical design variables. They are considered static variables.

These problem variables will be referred to as optimization variables in this thesis.

**State derivative function.**

Dynamic systems describe the evolution of the system states over time. Mathematically this behavior is often expressed as an ordinary differential equation:

$$\dot{\boldsymbol{\xi}}(t) = \boldsymbol{f}(\boldsymbol{\xi}(t), \boldsymbol{u}(t), \boldsymbol{x}_p, t) \tag{1.2}$$

This equation is also referred to as the state derivative function. Using the example of a FOWT, some the system states are usually the generator speed and platform pitch.

In addition to the state derivative function, there are other types of equations that are used to describe the system and limitations on its behavior.

**Boundary constraints.**

Since dynamic systems are time-varying, additional boundary constraints are also prescribed. They specify the optimization variable values at the start and end of the time horizon. They can be expressed as equality or inequality constraints:

$$\boldsymbol{\Phi}(\boldsymbol{\xi}(t_0), \boldsymbol{u}(t_0), t_0, \boldsymbol{\xi}(t_f), \boldsymbol{u}(t_f), t_f) = \mathbf{0} \tag{1.3}$$

$$\boldsymbol{\psi}(\boldsymbol{\xi}(t_0), \boldsymbol{u}(t_0), t_0, \boldsymbol{\xi}(t_f), \boldsymbol{u}(t_f), t_f) \leq \mathbf{0} \tag{1.4}$$

Practically, boundary constraints specify the values of the optimization variables at the initial and final time points. For example, in the design of a thermal management system, these constraints can be used to set the desired value of the working fluid's temperature at the start and end of the time horizon.

**Path constraints.**

Mathematically, path constraints are algebraic constraints specified to restrict the optimization variables' trajectories throughout the time horizon. They can be expressed as equality or inequality constraints:

$$\boldsymbol{h}(\boldsymbol{\xi}(t), \boldsymbol{u}(t), \boldsymbol{x}_p, t) = \mathbf{0} \tag{1.5}$$

$$\boldsymbol{g}(\boldsymbol{\xi}(t), \boldsymbol{u}(t), \boldsymbol{x}_p, t) \leq \mathbf{0} \tag{1.6}$$

For example, when designing a wind turbine, one common constraint is limiting the maximum power generated by the turbine to be at the rated power or below it. This is done by placing path constraints on the variables that are used to calculate the power.

**Objective function.**

The final element of a dynamic optimization problem is the objective function. The objective term represents the quantity of interest that we would like to minimize or maximize. There are two kinds of objective function terms encountered in dynamic optimization problems, Lagrange terms $\mathcal{L}(\cdot)$, and Mayer terms $\mathcal{M}(\cdot)$. Lagrange terms denote the running cost, and the Mayer terms denote the terminal costs. Correspondingly, Lagrange terms depend on the value optimization variables throughout the time horizon, while the Mayer term depends only on the optimization variable values at the initial and final time points. They are represented as:

$$\phi = \mathcal{M}(\boldsymbol{\xi}(t_0), t_0, \boldsymbol{\xi}(t_f), t_f, \boldsymbol{x}_p) + \int_{t_0}^{t_f} \mathcal{L}(t, \boldsymbol{\xi}(t), \boldsymbol{u}(t), \boldsymbol{x}_p) dt \tag{1.7}$$

The full dynamic optimization problem with all the elements present is given by:

$$\min_{\boldsymbol{\xi}, \boldsymbol{u}, \boldsymbol{x}_p} \quad \boldsymbol{\phi} = \mathcal{M}(\boldsymbol{\xi}(t_0), t_0, \boldsymbol{\xi}(t_f), t_f, \boldsymbol{x}_p) + \int_{t_0}^{t_f} \mathcal{L}(t, \boldsymbol{\xi}(t), \boldsymbol{u}(t), \boldsymbol{x}_p) dt \tag{1.8a}$$

$$\text{subject to:} \quad \dot{\boldsymbol{\xi}}(t) - \boldsymbol{f}(\boldsymbol{\xi}(t), \boldsymbol{u}(t), \boldsymbol{x}_p, t) = \boldsymbol{0} \tag{1.8b}$$

$$\boldsymbol{h}(\boldsymbol{\xi}(t), \boldsymbol{u}(t), \boldsymbol{x}_p, t) = \boldsymbol{0} \tag{1.8c}$$

$$\boldsymbol{g}(\boldsymbol{\xi}(t), \boldsymbol{u}(t), \boldsymbol{x}_p, t) \leq \boldsymbol{0} \tag{1.8d}$$

$$\boldsymbol{\psi}(\boldsymbol{\xi}(t_0), \boldsymbol{u}(t_0), t_0, \boldsymbol{\xi}(t_f), \boldsymbol{u}(t_f), t_f) \leq \boldsymbol{0} \tag{1.8e}$$

Unlike static optimization problems, DO problems are defined by the optimization variables' value at the infinite points in a given time horizon $[t_0, t_f]$. For this reason, they are also called infinite-dimensional optimization problems [19, 20]. This aspect of DO problems can make it harder to solve them. Though various solution methods have been investigated for solving dynamic optimization problems, a class of methods called direct transcription (DT) methods have proven to be useful [7, 8, 19, 21, 22]. DT methods are part of a family of direct numerical methods that transcribe the infinite-dimensional problem to finite-dimensional problems as given by Eq. (1.1) by approximating the continuous functions with piecewise continuous functions in the given time interval.

In this thesis, we consider two design domains: the physical plant and the control system design. The next section elaborates on how a system level optimal solution can be found, by taking into account the coupling between the plant design and the control design.

## 1.2 Control Co-Design

Control co-design involves simultaneously optimizing different plant and control design decision associated with a given system [8,23]. Traditional engineering practices often involve designing the plant first and then designing a suitable control system for the given plant design [11,14,24]. However, this approach will only yield system-optimal designs if there exists no coupling between the plant and the control systems. Many systems exhibit tight coupling between the physical and control systems. This approach was primarily applied to structural optimization problems [25]. But studies have shown the applicability of this approach to other domains as well [7, 23, 26–28].

This work deals with the two main approaches to control co-design, nested control co-design and simultaneous control co-design. It is essential to look at the physical system design and control system design processes, before a discussion of CCD can be presented.

### 1.2.1 Optimal Physical System Design

The primary goal of the physical system design is to identify the optimal plant parameters $\boldsymbol{x}_p$. In the case of plant design, commonly a static model is used[1].

$$\min_{\boldsymbol{x}_p} \quad \boldsymbol{\phi}_p(\boldsymbol{x}_p) \tag{1.9a}$$

$$\text{subject to:} \quad \boldsymbol{h}_p(\boldsymbol{x}_p) = \boldsymbol{0} \tag{1.9b}$$

$$\boldsymbol{g}_p(\boldsymbol{x}_p) \leq \boldsymbol{0} \tag{1.9c}$$

Solving this problem will yield the optimal set of plant variables $\boldsymbol{x}_p$. Then for this optimal plant design, a control system is designed as shown in the next section.

[1] A dynamic model can also be used as discussed in Ref. [14].

### 1.2.2 Optimal Control Systems Design

The control system objective, in a sequential approach, is to identify the input $\boldsymbol{u}(t)$ that would produce the desired performance for the given plant design $\boldsymbol{x}_p$ from Eq. (1.9). Additional factors like robustness and cost are also commonly included. Similar to Eq. (1.9), the control system design uses a model that depends on the control input and the state trajectories. An open-loop control design problem can be formulated as:

$$\min_{\boldsymbol{u}(t)} \quad \phi_c = \mathcal{M}_c(\boldsymbol{\xi}(t_0), t_0, \boldsymbol{\xi}(t_f), t_f) + \int_{t_0}^{t_f} \mathcal{L}_c(\boldsymbol{u}(t), \boldsymbol{\xi}(t), t)dt \tag{1.10a}$$

$$\text{subject to:} \quad \dot{\boldsymbol{\xi}}(t) - \boldsymbol{f}_c(\boldsymbol{u}(t), \boldsymbol{\xi}(t), t) = 0 \tag{1.10b}$$

$$\boldsymbol{h}_c(\boldsymbol{u}(t), \boldsymbol{\xi}(t), t) = \boldsymbol{0} \tag{1.10c}$$

$$\boldsymbol{g}_c(\boldsymbol{u}(t), \boldsymbol{\xi}(t), t) \leq \boldsymbol{0} \tag{1.10d}$$

$$\boldsymbol{\psi}_c(\boldsymbol{\xi}(t_0), \boldsymbol{u}(t_0), t_0, \boldsymbol{\xi}(t_f), \boldsymbol{u}(t_f), t_f) \leq \boldsymbol{0} \tag{1.10e}$$

Solving the plant and control design optimization problems sequentially produces system-optimal results only if there exist no coupling between the two disciplines. However, if there exist a coupling then the sequential approach does not produce system-optimal results. To achieve system-optimal performance, this coupling must be accounted for, and the physical and control design optimization must be done in a integrated optimization problem [24].

Optimal control and open-loop control trajectories assume all information, past and future about all possible disturbances are known apriori. This information could also be an estimation based on predicted values. Although it is challenging to incorporate these considerations in implementable controllers, this approach will provide an upper limit on what is physically achievable for the control system [29]. This is aligned with the goal of early-stage design studies, where we want to understand what is theoretically the best possible performance the system can achieve.

### 1.2.3 Simultaneous CCD

The simultaneous CCD approach optimizes the physical system design and control system design in a single formulation. The formulation for simultaneous CCD problem has been presented

in Eq. (1.8). This formulation is a combination of Eq. (1.10) and Eq. (1.9). The dynamic model used in this problem incorporates the effects of both the plant variables and the control variables on the system states. This model can be used to identify system-optimal results as it takes into account the design coupling between the physical system design and control system design.

Grouping all the elements into single formulation has its benefits and drawbacks. On a positive note, it guarantees that the obtained solution is a system-level optimal solution under the condition that a global solution can be found. But, a single formulation increases the problem size and complexity. As the problem size and model/constraint complexity increase, it becomes much harder to solve the problems. For example, in some cases of FOWT design when a full model of the system is used, the problem contains a hundred states! For such problems, it is not always possible to construct and solve a single formulation of all problem elements. Sec. 4.3 provides a much deeper discussion on the advantages and disadvantages of the simultaneous CCD problem.

Nested co-design is a specific reformulation of the simultaneous co-design problem. Nested co-design uses nested or bilevel optimization where the outer loop solves the plant optimization problem and the inner loop solves for control optimization problem. The outer-loop optimization problem is:

$$\min_{\boldsymbol{x}_p} \quad o(\mathcal{I}(\boldsymbol{x}_p, \boldsymbol{\xi}, \boldsymbol{u}), \boldsymbol{x}_p) \tag{1.11a}$$

$$\text{subject to:} \quad \boldsymbol{h}_o(\boldsymbol{x}_p) = \boldsymbol{0} \tag{1.11b}$$

$$\boldsymbol{g}_o(\boldsymbol{x}_p) \leq \boldsymbol{0} \tag{1.11c}$$

where $\mathcal{I}(\boldsymbol{x}_p, \boldsymbol{\xi}, \boldsymbol{u})$ is the solution to the following inner-loop optimization problem:

$$\min_{\boldsymbol{\xi}, \boldsymbol{u}} \quad o(\boldsymbol{\xi}, \boldsymbol{u}, \boldsymbol{x}_p^\dagger) \tag{1.12a}$$

$$\text{subject to:} \quad \boldsymbol{\xi}(t) - \boldsymbol{f}_i(\boldsymbol{\xi}(t), \boldsymbol{u}(t), \boldsymbol{x}_p^\dagger, t) = \boldsymbol{0} \tag{1.12b}$$

$$\boldsymbol{h}_i(\boldsymbol{\xi}(t), \boldsymbol{u}(t), \boldsymbol{x}_p^\dagger, \boldsymbol{\xi}(t_0), \boldsymbol{\xi}(t_f), t) = \boldsymbol{0} \tag{1.12c}$$

$$\boldsymbol{g}_i(\boldsymbol{\xi}(t), \boldsymbol{u}(t), \boldsymbol{x}_p^\dagger, \boldsymbol{\xi}(t_0), \boldsymbol{\xi}(t_f), t) \leq \boldsymbol{0} \tag{1.12d}$$

$$\boldsymbol{\psi}(\boldsymbol{\xi}(t_0), \boldsymbol{u}(t_0), t_0, \boldsymbol{\xi}(t_f), \boldsymbol{u}(t_f), t_f) \leq \boldsymbol{0} \tag{1.12e}$$

where $\boldsymbol{x}_p^\dagger$ is a candidate plant design from the outer loop in Eq. (1.11). References [23, 26] show that the under certain conditions the solution of the nested problem is equivalent to the simultaneous problem's solution.

One major advantage with nested CCD is that it allows the problem to be broken into different parts, and facilitates easier solution. For example, in the breakup of variables considered in Eqs. (1.11,1.12), the plant design optimization, which is a static optimization problem, can be solved using a specific NLP solver while the inner loop which solves for the states and controls can be solved using a DO solver. If the control design problem is in a specific form like the linear-quadratic regulator (LQR), then optimality conditions can be applied to obtain a practical solution. Such a approach is not feasible when solving a simultaneous CCD problem. Additionally, for large-scale problems where multiple subsystems are considered, the nested method can help decrease the difficulty in solving the problem. Sec. 4.3 again provides a much deeper discussion on the advantages and disadvantages of the nested CCD formulation.

Even though there are other methods for solving CCD problems, nested and simultaneous CCD methods are the most common strategies used [30]. However, only a few studies offer detailed comparisons between the nested and simultaneous CCD methods. A critical comparison study is paramount as CCD is being applied to increasingly complex problems. Such a study could help decide an appropriate strategy for solving a given problem instead of domain-specific suggestions, and elucidate much needed early insights into the appropriate choice of strategy and implementation techniques.

## 1.3 Research Questions

The primary focus of this thesis is to find efficient methods to solve certain DO and CCD problem classes. As the behavior of the system under consideration becomes more complex, highly nonlinear models need to be used to capture this behavior [5, 7, 31]. Using these detailed models in design optimization studies and solving them becomes computationally intensive. Many design optimization studies use simplified models of the system dynamics or construct surrogate models

of the system behavior in order to simplify and solve the design problem [32–34]. However, these simplified models cannot capture the complex system behavior and do not provide sufficient design insights. There is a pressing need to find efficient methods and strategies to solve these complex problems.

The CCD problems discussed in this thesis are posed as DO problems. In the context of CCD, the use of these simplified models can be counterproductive as the use of these simplified models will result in certain quantities being overpredicted or underpredicted. Efficient methods to solve DO problems will also help in solving CCD more efficiently while capturing the different interactions in the system.

The choice of CCD method used to solve a given problem has been based on the ease of implementation and literature-based recommendations. As of now, there is not a comprehensive or even a partial set of heuristics that provide clear, evidence-based guidelines on how to choose between nested and simultaneous CCD strategies for a given problem. It is crucial to keep in mind that the guidelines can be based on different aspects of a given strategy. For example, the choice of the method can vary based on the complexity of the physics involved, the nature of the coupling between the plant and control variables, how the model is constructed, and the efficient numerical implementation. A common denominator among these different categories is efficient numerical implementation. A study that can provide guidelines based on efficient implementation of nested and simultaneous CCD methods will provide insights that can be broadly applicable to different types of CCD problems.

## 1.4 Thesis Overview

Based on the previous discussions about the possible difficulties in solving CCD and DO problems, three studies have been carried out to find efficient strategies to solve them.

### 1.4.1 Study 1: Using LQDO Elements in Solving NLDO Problems

The first study aims to leverage the problem structure of dynamic optimization problems to develop efficient methods and tools to solve them. We start with a subclass of NLDO problems, namely linear-quadratic dynamic optimization problems (LQDO). For this class of problems, the objective function is quadratic, and the constraints are linear. Highly efficient and accurate computational tools have been developed for solving LQDO problems on account of their linear and quadratic problem elements. We identify the factors that enable creating these efficient tools and leverage them towards solving NLDO problems. We discuss three different strategies to solve NLDO problems using LQDO elements and analyze the requirements and limits of each approach.

### 1.4.2 Study 2: Comparison of Nested and Simultaneous CCD

In the first study, we identify the factors that lead to efficient solutions of simultaneous and nested optimization problems. We use these results and solve a CCD problem of an active vehicle suspension design first presented in Ref. [7], using both the nested and simultaneous CCD methods. We look at the impact of derivative methods, tolerances, and the number of discretization points on the solution accuracy and computational times. We use the solution approach and the results from this study to form some much needed early insights into the appropriate choice of strategy and implementation techniques. While guidelines applicable to broad classes of CCD problems will not be made, the existing literature-based comparisons and a detailed case study will better demonstrate the state-of-the-art understanding and how a thorough investigation can be conducted to yield the desired implementation insights.

### 1.4.3 Study 3: CCD of Floating Offshore Wind Turbines

In the third study, we leverage the insights gained from the first two studies and solve a large and complex engineering problem of the design of floating offshore wind turbines (FOWT). FOWT is a concrete example of a system where the coupling between the plant and controller must be taken into account to obtain an optimal design. Traditional FOWT design has followed a sequential

process where the physical plant parameters are designed first and a controller is then designed for this plant [35]. Not accounting the strong interactions in FOWTs between the plant systems and control system yields sub-optimal and unstable systems, reducing the economic viability of offshore wind systems [36]. Recent studies have identified the need for integrated design of FOWT systems [3, 35]. We study the effect of CCD and optimal control strategies on the design of the system.

The rest of the thesis is organized as follows:

1. Chapter 2 discusses and establishes the elements of NLDO and LQDO problems and explains how the direct transcription method can be used to solve DO problems.
2. Chapter 3 explores the uses of LQDO problem elements in solving NLDO problems.
3. Chapter 4 provides comparison between nested and simultaneous CCD methods using a the CCD problem of an active vehicle suspension.
4. Chapter 5 defines and solves a CCD of a floating offshore wind turbine (FOWT).
5. The final chapter summarizes thesis and provides directions for future work.

# Chapter 2

# Dynamic Optimization and Direct Transcription

This chapter explains the different solution approaches for solving dynamic optimization problems. The topics covered in this chapter will set up the arguments for the studies previously mentioned. We start with the general nonlinear dynamic optimization (NLDO) problems, then discuss the direct transcription (DT) method and how it is used to solve the NLDO problem. We then talk about linear-quadratic dynamic optimization problems (LQDO), a subclass of NLDO, and the factors that facilitate the development of efficient solvers for LQDO problems.

## 2.1 Dynamic Optimization

Dynamic optimization (DO), or the optimization of systems with time-varying behavior, has proven to be instrumental to the advancement of many domains, including aerospace [37,38], agricultural [39], chemical [20,21], energy [40], financial [41], mechanical [7,34], and medical [42] applications. Also, frequently termed optimal control problems, the infinite-dimensional nature of DO necessitates specialized solution strategies. A variety of numerical approaches have been developed, including those based on the optimality conditions [37,43–45] and direct methods based on transcribing the DO problem into a finite-dimensional optimization problem. Consider the general NLDO problem given by Eq. (1.8). There are two main families of methods that are used to solve DO problems. Direct methods and indirect methods [7,14,19,21].

### 2.1.1 Indirect Methods

Indirect methods use optimality conditions derived from Pontryagins' maximum principle to find the optimal trajectory for the problem [46]. The application of these optimality conditions results in a two-point boundary value problem (TPBVP). These TPBVP problems are then solved analytically or numerically. The solution of this BVP will yield the extremal trajectories, but under

some conditions, these trajectories can be optimal [47]. The numerical indirect methods follow the "optimize-then-discretize" approach.

To understand the approach, consider a simple example of an optimal control problem without inequality constraints of the form:

$$\min_{\boldsymbol{u}} \quad J = \int_0^{t_f} \mathcal{L}(\boldsymbol{x}(t), \boldsymbol{u}(t))dt + \mathcal{M}(\boldsymbol{x}(t_f)) \tag{2.1a}$$

$$\dot{\boldsymbol{x}} - \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{0} \tag{2.1b}$$

$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0 \tag{2.1c}$$

where $(\boldsymbol{x}, \boldsymbol{u})$ are the states and controls.

The quantity $H$, known as the Hamiltonian is defined as:

$$H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}) = \mathcal{L}(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\lambda}^T \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{2.2}$$

where $\boldsymbol{\lambda}$ is known as costate vector or adjoint.

If $(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*)$ are the optimal trajectories, the first order optimality conditions states that:

$$\frac{\partial H(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*)}{\partial \boldsymbol{u}} = \boldsymbol{0} \tag{2.3a}$$

$$\dot{\boldsymbol{\lambda}}^*(t) = -\frac{\partial H(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*)}{\partial \boldsymbol{x}} \tag{2.3b}$$

$$\boldsymbol{\lambda}(t_f) = \left[\frac{\partial \mathcal{M}}{\partial \boldsymbol{x}}\right]^T_{t=t_f} \tag{2.3c}$$

$$\dot{\boldsymbol{x}}^*(t) = \frac{\partial H(\boldsymbol{x}^*, \boldsymbol{u}^*, \boldsymbol{\lambda}^*)}{\partial \boldsymbol{\lambda}} \tag{2.3d}$$

$$\boldsymbol{\lambda}^T \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{0} \tag{2.3e}$$

the state values have specified initial conditions, and the costate values have final conditions, resulting in a TPBVP. The solution of these sets of equations will yield the optimal trajectories.

There are several issues with indirect methods that make them unsuitable for solving complex problems. The equations given by the first-order optimality conditions can rarely be solved analytically, and therefore require numerical methods to solve the BVP. Evaluation of the partial derivatives of the Hamiltonian is complicated and requires extensive experience with optimal control. Additionally, in some problems, black-box functions and lookup tables are used, and de-

termining the derivative information for such problems is highly complex [14, 19]. An initial guess value for the costate vector $\boldsymbol{\lambda}$ is required to solve these problems. These quantities are not physical, and therefore providing accurate estimates is a process of trial and error. Even with a good guess, the problem can be ill-conditioned. The addition of inequality constraints to the problem increases the difficulty in finding the initial guess of the states, costates, and the appropriate switching structure [21]. When using direct methods, all these difficulties are avoided, as direct methods do not use these optimality conditions.

### 2.1.2 Direct Methods

There are two variants in the paradigm of direct methods, based on the optimization variable(s) that are discretized: sequential methods and simultaneous methods. The direct methods follow a "discretize-then-optimize" approach. The continuous time problem is first discretized to form a NLP, and this NLP is solved to obtain the optimal variables.

**Sequential methods.**

In sequential methods, only the control variables are discretized. For this reason, this approach is also known as control vector parameterization [21]. The control trajectory is discretized using piecewise polynomial functions, and the coefficients for these polynomials are the optimization variables. Given an initial condition and control input, the differential equation with algebraic constraints is solved using forward numerical simulation to obtain the trajectories. Even though sequential solvers are easy to set up, there are several issues. For every iteration of the optimization problem, a DAE must be solved. This becomes expensive for large scale problems [48]. Additionally, sequential approaches cannot handle open-loop instabilities, and path constraints can only be handled approximately [14].

**Simultaneous methods.**

Simultaneous methods discretize both the state and control trajectories using collocation at finite number of time points. These methods are also known as direct transcription (DT) meth-

ods [49]. This results in the creation of a large-scale NLP problem, as opposed to solving DAE for each simulation.

## 2.2 Direct Transcription

The following section elaborates on the DT method and shows how it can be used to solve a DO problem. Consider the NLDO problem given by Eq. (1.8). The given problem is solved through DT by using the following steps:

### 2.2.1 Discretization

The first step in the DT process is to discretize the continuous-time horizon $\boldsymbol{t} \in [t_0, t_f]$ into $n_t$ finite-time points, where $n_t$ is the number of discretization points. These discretized points are also known as the mesh:

$$\boldsymbol{t} = [t_0, t_1, t_2, \ldots, t_{n_t-1}, t_{n_t}] \tag{2.4}$$

There are different methods for choosing the discretization points. One straightforward way is to have an equidistant mesh, i.e., the points in the horizon $[t_0, t_f]$ are equally spaced.

$$h = \frac{t_f - t_0}{n_t} \tag{2.5}$$

and,

$$t_1 = t_0 + h, \quad t_2 = t_1 + h, \quad \ldots \quad , t_f = t_{n_t-1} + h \tag{2.6}$$

Legendre-Gauss-Lobatto (LGL) points, Legendre-Gauss (LG) points, and Legendre-Gauss-Radau (LGR) points are examples of these schemes. The location of these points correspond to the roots of an orthogonal polynomial [14, 19, 21].

Based on the discretized time points, the state and control trajectories are also discretized as:

$$\boldsymbol{\Xi} = \begin{bmatrix} \boldsymbol{\Xi}_1 \\ \boldsymbol{\Xi}_2 \\ \boldsymbol{\Xi}_3 \\ \vdots \\ \boldsymbol{\Xi}_{n_\xi} \end{bmatrix} = \begin{bmatrix} \xi_1(t_1) & \xi_2(t_1) & \xi_3(t_1) & \dots & \xi_{n_\xi}(t_1) \\ \xi_1(t_2) & \xi_2(t_2) & \xi_3(t_2) & \dots & \xi_{n_\xi}(t_2) \\ \xi_1(t_3) & \xi_2(t_3) & \xi_3(t_3) & \dots & \xi_{n_\xi}(t_3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \xi_1(t_{n_t}) & \xi_2(t_{n_t}) & \xi_3(t_{n_t}) & \dots & \xi_{n_\xi}(t_{n_t}) \end{bmatrix} \tag{2.7a}$$

$$\boldsymbol{U} = \begin{bmatrix} \boldsymbol{U}_1 \\ \boldsymbol{U}_2 \\ \boldsymbol{U}_3 \\ \vdots \\ \boldsymbol{U}_{n_u} \end{bmatrix} = \begin{bmatrix} u_1(t_1) & u_2(t_1) & u_3(t_1) & \dots & u_{n_u}(t_1) \\ u_1(t_2) & u_2(t_2) & u_3(t_2) & \dots & u_{n_u}(t_2) \\ u_1(t_3) & u_2(t_3) & u_3(t_3) & \dots & u_{n_u}(t_3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ u_1(t_{n_t}) & u_2(t_{n_t}) & u_3(t_{n_t}) & \dots & u_{n_u}(t_{n_t}) \end{bmatrix} \tag{2.7b}$$

where $(n_\xi, n_u)$ are the number of states and control variables, respectively.

Since the plant variables $\boldsymbol{x}_p$ are considered static, they are not discretized with respect to the mesh:

$$\boldsymbol{x}_p = [p_1, p_2, p_3, \dots p_{n_p}]^T \tag{2.8}$$

where $n_p$ denotes the number of plant variables.

### 2.2.2 Dynamic Constraints

The continuous-time dynamics represented by the state derivative functions in Eq. (1.2) are then discretized using the discretized state and control matrices and a specific method. The discretized dynamic equation is called the defect constraint. There are different ways to construct the defect constraints using various local or global collocation schemes. In local collocation schemes, the state and control trajectories are approximated using low-degree polynomials. Global collocation schemes, also called pseudospectral methods, use global trial functions with orthogonal collocations to approximate the trajectories [7, 8, 10, 19, 21, 22].

In this work, we concentrate mainly on local collocation or single-step methods to form defect constraints. Consider the time interval $t \in [t_k, t_{k+1}]$. In this interval, the dynamics can be

approximated using the trapezoidal rule as:

$$\boldsymbol{\zeta}(\boldsymbol{t}_k, \boldsymbol{t}_{k+1}) = \boldsymbol{\Xi}_{k+1} - \boldsymbol{\Xi}_k - \frac{h}{2}\left[\boldsymbol{f}_{k+1} + \boldsymbol{f}_k\right] \quad \forall k = 1, \ldots, n_t - 1 \tag{2.9}$$

where $\boldsymbol{f}_k = \boldsymbol{f}(t_k, \boldsymbol{\Xi}_k, \boldsymbol{U}_k, \boldsymbol{x}_p)$.

Combining $\boldsymbol{\zeta}$ for all time points we have:

$$\boldsymbol{\zeta}_t = \begin{bmatrix} \boldsymbol{\zeta}(t_1) \\ \boldsymbol{\zeta}(t_2) \\ \vdots \\ \boldsymbol{\zeta}(t_{n_t-1}) \end{bmatrix} \tag{2.10}$$

For a given interval, the derivative function is evaluated at only two time points. As the number of mesh points increases, the final defect constraint matrix will be large and sparse. This sparsity facilitates efficient solution methods [19].

**Additional constraints.**

The algebraic equality and inequality constraints in Eqs. (1.5,1.3) are evaluated at each mesh point $t_k$ to ensure that the constraints are satisfied at each point:

$$\boldsymbol{h}_k = \boldsymbol{h}(t_k, \boldsymbol{\Xi}_k, \boldsymbol{U}_k, \boldsymbol{x}_p, \boldsymbol{\Xi}_1, \boldsymbol{\Xi}_{n_t}) = \boldsymbol{0} \quad \forall k = 1, \ldots, n_t \tag{2.11a}$$

$$\boldsymbol{g}_k = \boldsymbol{g}(t_k, \boldsymbol{\Xi}_k, \boldsymbol{U}_k, \boldsymbol{x}_p, \boldsymbol{\Xi}_1, \boldsymbol{\Xi}_{n_t}) \leq \boldsymbol{0} \quad \forall k = 1, \ldots, n_t \tag{2.11b}$$

These are combined as:

$$\boldsymbol{h}_t = \begin{bmatrix} \boldsymbol{h}_1 & \boldsymbol{h}_2 & \cdots & \boldsymbol{h}_{n_t} \end{bmatrix}^T \tag{2.12a}$$

$$\boldsymbol{g}_t = \begin{bmatrix} \boldsymbol{g}_1 & \boldsymbol{g}_2 & \cdots & \boldsymbol{g}_{n_t} \end{bmatrix}^T \tag{2.12b}$$

### 2.2.3 Objective Function

Similar to the constraints, the Lagrange and Mayer terms in Eq. (1.7) are also reevaluated in terms of the discretized optimization variables $(\boldsymbol{\Xi}, \boldsymbol{U})$ from Eq. (2.7). The Mayer term can be evaluated exactly as it is dependent on the discretized values of the variables [22]. The Lagrange term, however, needs to be evaluated using numerical quadrature. In this thesis, we use the composite

trapezoidal rule to evaluate the Lagrange term:

$$o = \int_{t_0}^{t_f} \mathcal{L}(t, \boldsymbol{\xi}, \boldsymbol{u}, \boldsymbol{x}_p) dt \approx \frac{t_f - t_0}{n_t} \left( \frac{\mathcal{L}(t_0) + \mathcal{L}(t_f)}{2} + \sum_{k=1}^{n_t - 1} \mathcal{L}(t_k) \right) \tag{2.13}$$

### 2.2.4 Fully Discretized Form

Using the transformations discussed in the previous sections, the DO problem is given by Eq. (1.8) can be converted to the following nonlinear programming problem:

$$\min_{\boldsymbol{\Xi}, \boldsymbol{U}, \boldsymbol{x}_p} \quad o(\boldsymbol{t}, \boldsymbol{\Xi}, \boldsymbol{U}, \boldsymbol{x}_p) \tag{2.14a}$$

$$\text{subject to:} \quad \boldsymbol{\zeta}_t(\boldsymbol{t}, \boldsymbol{\Xi}, \boldsymbol{U}, \boldsymbol{x}_p) = \boldsymbol{0} \tag{2.14b}$$

$$\boldsymbol{h}_t(\boldsymbol{t}, \boldsymbol{\Xi}, \boldsymbol{U}, \boldsymbol{x}_p, \boldsymbol{\Xi}_1, \boldsymbol{\Xi}_{n_t}) = \boldsymbol{0} \tag{2.14c}$$

$$\boldsymbol{g}_t(\boldsymbol{t}, \boldsymbol{\Xi}, \boldsymbol{U}, \boldsymbol{x}_p, \boldsymbol{\Xi}_1, \boldsymbol{\Xi}_{n_t}) \leq \boldsymbol{0} \tag{2.14d}$$

To solve the nonlinear programming problem, various algorithms have been developed. Commercial solvers like *fmincon*, *SNOPT*, and *IPOPT* have all been used to solve the nonlinear programming problem generated by transcribing a DO problem [22, 45]. In this thesis, we use *fmincon* [50], the inbuilt constrained nonlinear optimization algorithm in *MATLAB* to solve the NLP problem. The insights and solutions obtained are generally independent of the solver.

## 2.3 Linear Quadratic Dynamic Optimization

This section discusses a particular subclass of NLDOs, namely linear-quadratic dynamic optimization (LQDO) problems. LQDO problems have a quadratic objective function subject to linear constraints.

### 2.3.1 Problem Formulation

The LQDO problem is represented as:

$$\min_{\boldsymbol{x}=[\boldsymbol{\xi},\boldsymbol{u},\boldsymbol{x}_p]} \quad \int_{t_0}^{t_f} (\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{F}^T\boldsymbol{x} + c)dt + \boldsymbol{R}(\boldsymbol{\xi}(t_0), \boldsymbol{\xi}(t_f), \boldsymbol{x}_p) \tag{2.15a}$$

$$\text{subject to:} \quad \dot{\boldsymbol{\xi}}(t) - [\boldsymbol{A}(t)\boldsymbol{\xi}(t) + \boldsymbol{B}(t)\boldsymbol{u}(t) + \boldsymbol{G}(t)\boldsymbol{x}_p + \boldsymbol{d}(t)] = \boldsymbol{0} \tag{2.15b}$$

$$\boldsymbol{h}^L(\boldsymbol{t}, \boldsymbol{\xi}, \boldsymbol{u}, \boldsymbol{x_p}, \boldsymbol{\xi}(t_0), \boldsymbol{\xi}(t_f)) = \boldsymbol{0} \tag{2.15c}$$

$$\boldsymbol{g}^L(\boldsymbol{t}, \boldsymbol{\xi}, \boldsymbol{u}, \boldsymbol{x_p}, \boldsymbol{\xi}(t_0), \boldsymbol{\xi}(t_f)) \leq \boldsymbol{0} \tag{2.15d}$$

where $(\boldsymbol{h}^L(\cdot), \boldsymbol{g}^L(\cdot))$ are linear in terms of the optimization variables and $\boldsymbol{R}(\cdot)$ is the linear-quadratic Mayer term. The elements of the LQDO problem only need to be linear/quadratic with respect to the optimization problem variables $(\boldsymbol{\xi}, \boldsymbol{u}, \boldsymbol{x}_p)$.

### 2.3.2 Quadratic Programming

Applying DT to the LQDO problem Eq. (2.15) yields a quadratic programming (QP) problem [8]. A QP problem formulation follows as:

$$\min_{\boldsymbol{x}} \quad \frac{1}{2}\boldsymbol{x}^T\boldsymbol{H}\boldsymbol{x} + \boldsymbol{f}^T\boldsymbol{x} + c \tag{2.16a}$$

$$\text{subject to:} \quad \boldsymbol{A}_{eq}\boldsymbol{x} = \boldsymbol{b}_{eq} \tag{2.16b}$$

$$\boldsymbol{A}_{in}\boldsymbol{x} \leq \boldsymbol{b}_{in} \tag{2.16c}$$

LQDO and QP problems have a predictable structure that facilitates the development of automated and efficient solvers [8, 22]. For QP/LQDO problems without inequality constraints, the optimal solution can be obtained with matrix inversion [9]. The QP that is encountered after the problem is discretized has problem elements that are large and sparse for the reasons explained in the previous section. There are many efficient algorithms for solving QPs including interior-point, active-set, augmented Lagrangian, conjugate gradient, and alternating direction method of multipliers [9, 20, 51, 52]. For large-scale sparse QPs, certain algorithms can utilize sparse linear algebra to efficiently find optimal solutions [21, 52, 53].

In this work, we will be using *DTQP*, which is a *MATLAB*-based tool for efficiently constructing the matrices/vectors for all LQDO and NLDO problem elements using user-friendly, structure-based definition [54]. To solve the formulated QP, *DTQP* uses *quadprog*, *MATLAB*'s inbuilt quadratic programming solver. While there are a variety of other general DT-based tools [55–57], the initial focus only on LQDO problems in the *DTQP* has led to a number of focused advancements. The sparse matrices are constructed using vectorized computations and direct creation of the row, column, and value sequences [58] that define the required matrices resulting in good scalability for both small and large meshes. A variety of defect constraint methods, including several single-step and pseudospectral schemes, as well as the common quadrature schemes, discussed in Sec. 2.2.3 are available. The main algorithms are described in Ref. [8].

# Chapter 3

# Linear-Quadratic Methods in Solving Nonlinear Dynamic Optimization Problems with Direct Transcription

In this chapter, we discuss the use of LQDO elements in solving NLDO problems. These three strategies are discussed and used to solve different dynamic optimization problems as case studies. Using these case studies, the advantages and drawbacks of these strategies are discussed[2].

## 3.1 Introduction

As is the case with the advancement of many numerical methods, exploiting specific mathematical problem structure can lead to developments that both increase the efficiency and robustness of the underlying techniques. This has undoubtedly been the case in DT research, such as with problem sparsity and derivatives [20, 37]. Here we will be exploring methods for exploiting a particular class of DO problems where the objective function is quadratic, and the constraints are linear, i.e., *linear-quadratic dynamic optimization* (LQDO). Because of the considerable degree of problem structure, there are many advantages to developing numerical methods that specifically exploit these patterns. This has historically included explicit matrix construction [8, 54], two-level [24, 26, 34, 59], quasilinearization [38, 44, 46, 53, 60–64], and sequential quadratic programming methods [20, 37, 53, 64, 65], which will be discussed in Sec. 3.3. Furthermore, many relevant DO problems contain linear or quadratic problem elements [42, 51, 60–62], such as is the case in model-predictive control [51, 64, 66]. Therefore, generalizing the uses of LQDO-based methods will allow for this structure to be exploited when needed and is the primary purpose of this article.

[2]This chapter is based on the following publication [22].

Many dynamic optimization problems contain both nonlinear and linear elements. It is difficult to process/evaluate nonlinear problem elements as compared to linear elements. However, many DT solvers do not treat linear elements differently than nonlinear elements. Thus, they do not leverage the unique properties of linear-quadratic elements that make them easier to process. Additionally, it is possible to create robust and efficient solvers for LQDO problems, and we can pinpoint the factors that contribute to their efficiency, and leverage them for developing efficient tools.

With all these points in favor of LQDO tools, we seek to answer the following questions:

1. How can we leverage the factors that contribute to the efficiency of LQDO problems to solve NLDO problems?
2. If an NLDO has both linear-quadratic and nonlinear elements, can we separate them to solve each problem efficiently?
3. Even if there are no linear quadratic elements in a problem, is there a way to use linear quadratic elements still and solve the original NLDO problem?

## 3.2 Reference Linear-Quadratic Dynamic Optimization Problems

As discussed previously, linear-quadratic dynamic optimization (LQDO) problem is a subclass of NLDO problem given in Eq. (1.8) where the objective function is limited to quadratic terms and the constraints have only linear terms. To completely characterize the reference LQDO problem class, we first need to define the Jacobian matrix of a vector-valued function $\boldsymbol{e}$ of size $m$ in variables $\boldsymbol{x}$ of size $n$:

$$\boldsymbol{J}^{\boldsymbol{e}}_{\boldsymbol{x}}(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial e_1(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial e_1(\boldsymbol{x})}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial e_m(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial e_m(\boldsymbol{x})}{\partial x_n} \end{bmatrix} \tag{3.1}$$

which is a matrix of all its first-order partial derivatives, and the Hessian matrix of a scalar-valued function $e$ in variables $\boldsymbol{x}$ of size $n$:

$$\boldsymbol{H}_{\boldsymbol{x}}^{e}(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial^2 e(\boldsymbol{x})}{\partial x_1^2} & \cdots & \frac{\partial^2 e(\boldsymbol{x})}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 e(\boldsymbol{x})}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 e(\boldsymbol{x})}{\partial x_n^2} \end{bmatrix} \tag{3.2}$$

which is a square matrix of second-order partial derivatives.

Now consider some reference solution $\bar{\boldsymbol{x}}$ comprised of reference control trajectories $\bar{\boldsymbol{u}}$, state trajectories $\bar{\boldsymbol{\xi}}$, and parameter values $\bar{\boldsymbol{x}}_p$. Using the derivative matrices, we can construct the best local approximations of the NLDO problem elements about $\bar{\boldsymbol{x}}$. The best linear approximation of $\boldsymbol{e}$ around point $\bar{\boldsymbol{x}}$ is:

$$\boldsymbol{e}(\boldsymbol{x}) \approx \boldsymbol{e}^{L}(\boldsymbol{x}, \bar{\boldsymbol{x}}) = \boldsymbol{e}\left(\bar{\boldsymbol{x}}\right) + \boldsymbol{J}_{\boldsymbol{x}}^{\boldsymbol{e}}\left(\bar{\boldsymbol{x}}\right)\left[\boldsymbol{x} - \bar{\boldsymbol{x}}\right] \tag{3.3}$$

which is the first-order multivariate Taylor expansion of $\boldsymbol{e}$. The best quadratic approximation of $e$ around $\bar{\boldsymbol{x}}$ is:

$$e(\boldsymbol{x}) \approx e^{Q}\left(\boldsymbol{x}, \bar{\boldsymbol{x}}\right) = e\left(\bar{\boldsymbol{x}}\right) + \left[\boldsymbol{J}_{\boldsymbol{x}}^{e}\left(\bar{\boldsymbol{x}}\right)\right]^{\mathsf{T}}\left[\boldsymbol{x} - \bar{\boldsymbol{x}}\right] + \frac{1}{2}\left[\boldsymbol{x} - \bar{\boldsymbol{x}}\right]^{\mathsf{T}} \boldsymbol{H}_{\boldsymbol{x}}^{e}\left(\bar{\boldsymbol{x}}\right)\left[\boldsymbol{x} - \bar{\boldsymbol{x}}\right] \tag{3.4a}$$

which is the second-order Taylor polynomial.

Now, we apply these approximations on the appropriate problem elements of Eq. (1.8), resulting in the following DO problem:

$$\min_{\boldsymbol{x}=[\boldsymbol{u},\boldsymbol{\xi},\boldsymbol{x}_p]} \quad o^{Q}\left(t, \boldsymbol{u}, \boldsymbol{\xi}, \boldsymbol{x}_p, \boldsymbol{\xi}_0, \boldsymbol{\xi}_f, \bar{\boldsymbol{x}}\right) \tag{3.5a}$$

$$\text{subject to:} \quad \dot{\boldsymbol{\xi}}(t) - \boldsymbol{f}^{L}\left(t, \boldsymbol{u}, \boldsymbol{\xi}, \boldsymbol{x}_p, \bar{\boldsymbol{x}}\right) = \boldsymbol{0} \tag{3.5b}$$

$$\boldsymbol{h}^{L}\left(t, \boldsymbol{u}, \boldsymbol{\xi}, \boldsymbol{x}_p, \boldsymbol{\xi}_0, \boldsymbol{\xi}_f, \bar{\boldsymbol{x}}\right) = \boldsymbol{0} \tag{3.5c}$$

$$\boldsymbol{g}^{L}\left(t, \boldsymbol{u}, \boldsymbol{\xi}, \boldsymbol{x}_p, \boldsymbol{\xi}_0, \boldsymbol{\xi}_f, \bar{\boldsymbol{x}}\right) \leq \boldsymbol{0} \tag{3.5d}$$

$$\text{where:} \quad \boldsymbol{\xi}_0 = \boldsymbol{\xi}(t_0),\ \boldsymbol{\xi}_f = \boldsymbol{\xi}(t_f) \tag{3.5e}$$

which is an LQDO problem by construction because the highest-order terms in the objective function are quadratic and the highest-order terms in the constraints are linear[3].

[3]The dynamic constraint could be written in the same form as $\boldsymbol{h}^{L}$ and $\boldsymbol{g}^{L}$ but this form is equivalent because $[\dot{\boldsymbol{\xi}}(t)]_{\bar{\boldsymbol{x}}} = \dot{\boldsymbol{\xi}}(t)$ and $\boldsymbol{J}_{\bar{\boldsymbol{x}}}^{\dot{\boldsymbol{\xi}}}(\bar{\boldsymbol{x}}) = \boldsymbol{0}$.

For the linearized constraints, it is common to express the resulting expression stratified by optimization variable type. For example, $\boldsymbol{f}^L$ is a linear time-varying (LTV) dynamic system about the reference trajectory $\bar{\boldsymbol{x}}$:

$$\boldsymbol{f}^L\left(t, \boldsymbol{u}, \boldsymbol{\xi}, \boldsymbol{x}_p, \bar{\boldsymbol{x}}\right) = \boldsymbol{A}\left(t, \bar{\boldsymbol{x}}\right)\boldsymbol{\xi} + \boldsymbol{B}\left(t, \bar{\boldsymbol{x}}\right)\boldsymbol{u} + \boldsymbol{G}\left(t, \bar{\boldsymbol{x}}\right)\boldsymbol{x}_p + \boldsymbol{d}\left(t, \bar{\boldsymbol{x}}\right) \tag{3.6a}$$

$$\text{where:} \quad \boldsymbol{A}\left(t, \bar{\boldsymbol{x}}\right) = \boldsymbol{J}^{\boldsymbol{f}}_{\boldsymbol{\xi}}\left(\bar{\boldsymbol{x}}\right) \tag{3.6b}$$

$$\boldsymbol{B}\left(t, \bar{\boldsymbol{x}}\right) = \boldsymbol{J}^{\boldsymbol{f}}_{\boldsymbol{u}}\left(\bar{\boldsymbol{x}}\right) \tag{3.6c}$$

$$\boldsymbol{G}\left(t, \bar{\boldsymbol{x}}\right) = \boldsymbol{J}^{\boldsymbol{f}}_{\boldsymbol{x}_p}\left(\bar{\boldsymbol{x}}\right) \tag{3.6d}$$

$$\boldsymbol{d}\left(t, \bar{\boldsymbol{x}}\right) = \boldsymbol{f}\left(t, \bar{\boldsymbol{u}}, \bar{\boldsymbol{\xi}}, \bar{\boldsymbol{x}_p}\right) - \boldsymbol{J}^{\boldsymbol{f}}_{\boldsymbol{x}}\left(\bar{\boldsymbol{x}}\right)\bar{\boldsymbol{x}} \tag{3.6e}$$

Similar representations can be made for the additional path and boundary constraints, as well as the objective function.

As an example, consider one of the dynamic equations for the Van der Pol Oscillator problem (cf. Eq. (3.15b)):

$$e(\boldsymbol{x}) = -p_1\xi_1 + \xi_2\left[1 - p_2\xi_1^2\right] + u_1 \tag{3.7}$$

Now we have the LTV dynamic system defined in Eq. (3.6):

$$e^L(\boldsymbol{x}, \bar{\boldsymbol{x}}) = \begin{bmatrix} -\bar{p}_1 - 2\bar{p}_2\bar{\xi}_1\bar{\xi}_2 \\ 1 + \bar{p}_2\bar{\xi}_1^2 \end{bmatrix}^\mathsf{T} \begin{bmatrix} \xi_1 \\ \xi_2 \end{bmatrix} + u_1 - \begin{bmatrix} \bar{\xi}_1 \\ \bar{\xi}_1^2\bar{\xi}_2 \end{bmatrix}^\mathsf{T} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} + \bar{p}_1\bar{\xi}_1 + 3\bar{p}_2\bar{\xi}_1^2\bar{\xi}_2 \tag{3.8}$$

### 3.2.1 Ordinary vs. Reference LQDO Problems

The current definition of an LQDO problem depends on the reference trajectory $\bar{\boldsymbol{x}}$, now termed a reference LQDO (RLQDO) problem.

It is quite common to have quadratic and linear problem elements in the NLDO problem; thus, the derivative matrices for those elements are constant and do not depend on $\bar{\boldsymbol{x}}$. These are termed ordinary linear-quadratic (OLQ) elements. Common problem elements such as linear dynamic systems, simple state bounds, initial and final conditions, quadratic Lagrange terms, and linear Mayer terms are all OLQ elements. Here we define an OLQ problem as one where there are only OLQ elements, and Eq. (1.8) and Eq. (3.5) are equivalent.

For example, these four constraints are all OLQ elements:

$$\dot{\xi}_1 + 4\xi_1 - 2u_1 + \sin(t) = 0, \quad \xi_1(0) = 0\xi_1 - 1 \leq 0, \quad tu_1 - 2p_1 + 1 \leq 0$$

noting that linear time-varying quantities are admissible. However, this objective function term is not an OLQ element even though it contains a quadratic term (however, it can be partitioned as is discussed in Sec. 3.3.1):

$$\int_{t_0}^{t_f} \left[\xi_2^2 + \sin(u_1)\right] dt$$

There are some standard LQDO problems such as the finite-horizon linear-quadratic regulator [43].

## 3.3 Methods for Solving NLDO Problems Using LQDO Problem Elements

In this section, we will discuss several methods for effectively utilizing LQDO problems and OLQ elements to solve NLDO problems.

### 3.3.1 Direct Incorporation of OLQ Elements

The first method is perhaps the most self-evident. For a given NLDO problem, we can treat the OLQ elements differently than the general nonlinear elements. In many DT-based software implementations, most of the problem elements are assumed to be nonlinear, and there is no way to specify if a particular constraint or objective term is an OLQ element (with the primary exception being linear constraints for the initial and final values of the trajectories and simple box constraints).

Consider the following partitioning of the NLDO problem in Eq. (1.8):

$$\min_{\boldsymbol{x}=[\boldsymbol{u},\boldsymbol{\xi},\boldsymbol{x}_p]} \quad \int_{t_0}^{t_f} \left[\ell^{OL}(\cdot) + \ell^{NL}(\cdot)\right] dt + m^{OL}(\cdot) + m^{NL}(\cdot) \tag{3.9a}$$

$$\text{subject to:} \quad \dot{\boldsymbol{\xi}}(t) - \boldsymbol{f}(\cdot) = \begin{bmatrix} \dot{\boldsymbol{\xi}}^{OL}(t) - \boldsymbol{f}^{OL}(\cdot) \\ \dot{\boldsymbol{\xi}}^{NL}(t) - \boldsymbol{f}^{NL}(\cdot) \end{bmatrix} = \boldsymbol{0} \tag{3.9b}$$

$$\boldsymbol{h}(\cdot) = \begin{bmatrix} \boldsymbol{h}^{OL}(\cdot) \\ \boldsymbol{h}^{NL}(\cdot) \end{bmatrix} = \boldsymbol{0} \tag{3.9c}$$

$$\boldsymbol{g}(\cdot) = \begin{bmatrix} \boldsymbol{g}^{OL}(\cdot) \\ \boldsymbol{g}^{NL}(\cdot) \end{bmatrix} \leq \boldsymbol{0} \tag{3.9d}$$

where the superscript $OL$ indicates an OLQ element and superscript $NL$ indicates a non-OLQ element. There are a few methods for determining the correct partitioning. First is a manual specification by the user, assuming it is supported by the selected tool. Second are the automated methods using either exact methods, such as symbolic or automatic differentiation, or a careful numerical procedure that checks the finite-difference values at several diverse points (because the result for OLQ elements would be the same for every point).

There are several potential advantages to this approach. First, most gradient-based NLP solvers require the computation of the problem's derivatives, which can be expensive. Running these computations for the OLQ elements within the NLP solver is an additional computational burden because results are independent of the current iteration. Therefore, computing the matrices once before starting the NLP solver will reduce the overall computational cost of computing the derivatives. If analytic derivatives are provided, the difference might be relatively small. If finite-difference methods are being utilized for computation, the gap can be much more pronounced, even if a sparsity pattern is provided. Additionally, because of the numerical errors introduced through finite-difference schemes, the derivatives for the OLQ elements may be more accurate.

Second, many NLP algorithms have special methods for handling linear constraints that may be more numerically accurate and computationally efficient. For example, *MATLAB*'s `fmincon` directly allows for the inclusion of the linear constraints and simple upper and lower bounds [50].

Derivatives are then known for these problem elements, and certain options can be enabled such as "honor bounds". More specifically, many NLP solvers utilize the Lagrangian function:

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\sigma}) = v(\mathbf{x}) + \boldsymbol{\lambda} \cdot \mathbf{h}(\mathbf{x}) + \boldsymbol{\sigma} \cdot \mathbf{g}(\mathbf{x}) \tag{3.10}$$

where $\boldsymbol{\lambda}$ and $\boldsymbol{\sigma}$ are the multipliers for the equality and inequality constraints, respectively [9]. Second derivative computations for $\boldsymbol{H}^{\mathcal{L}}_{\mathbf{x}}(\mathbf{x})$ may be required. Direct incorporation of the $\boldsymbol{h}^{OL}$ constraints will result in a linear system $\mathbf{h}^{OL}(\mathbf{x}) = \mathbf{A}_{\mathbf{h}^{OL}}\mathbf{x} - \mathbf{b}_{\mathbf{h}^{OL}}$. Now, observing only the contribution of equality constraint terms:

$$\boldsymbol{H}^{\mathcal{L}}_{\mathbf{x}}(\mathbf{x}) = \sum_j \boldsymbol{\lambda}^{OL}_j \cdot \cancelto{\mathbf{0}}{\left[\boldsymbol{H}^{\mathbf{h}^{OL}_j}_{\mathbf{x}}(\mathbf{x})\right]} + \sum_j \boldsymbol{\lambda}^{NL}_j \cdot \left[\boldsymbol{H}^{\mathbf{h}^{NL}_j}_{\mathbf{x}}(\mathbf{x})\right] + \cdots \tag{3.11}$$

Therefore, we see that no Hessian computations are required for linear equality constraints. This is also true for the other linear OLQ constraints.

Finally, both linear variable scaling and constraint row scaling can be efficiently implemented for the OLQ matrices. The extent of the advantages of direct incorporation of the OLQ elements can depend on the number and type of OLQ elements and mesh size, but generally, there are few disadvantages.

### 3.3.2 Two-Level Methods

Two-level optimization problems consider some partitioning of the optimization variables into upper $\boldsymbol{x}^u$ and lower $\boldsymbol{x}^l$ variables (sometimes termed outer and inner variables) [67]. These problems are represented as:

$$\min_{\boldsymbol{x}^u} \quad o\left(\boldsymbol{x}^u, \boldsymbol{x}^l\right) \tag{3.12a}$$

$$\text{subject to:} \quad \boldsymbol{x}^u \in \Omega \tag{3.12b}$$

$$\boldsymbol{x}^l \in L(\boldsymbol{x}^u) \tag{3.12c}$$

where $\boldsymbol{\Omega}$ is the feasibility region of $\boldsymbol{x}^u$, and $L(\boldsymbol{x}^u)$ is a set defined by the solution to an appropriate optimization problem. Determining a feasible $\boldsymbol{x}^l$ for a particular candidate $\boldsymbol{x}^u$ is solving the *lower-level problem* (LLP), while Prob. (3.12) is termed the *upper-level problem* (ULP). The two-level method is only one type of decomposition-based optimization strategy. Alternative decomposition-

based formulations and solvers may also utilize the properties and methods for LQDO problems and OLQ elements [51,68,69]. Here we will discuss two common partitioning schemes for NLDO problems with DT that incorporate LQDO LLPs.

**Shooting.**

The first partitioning scheme considers only $\boldsymbol{\xi}$ in the LLP, while $\boldsymbol{u}$ and $\boldsymbol{x}_p$ are optimized in the ULP. This is a version of the direct shooting approach [20] where feasible dynamics are determined in the LLP. Reducing the state dynamic analyses to a sparse system of linear equations can be computationally attractive, permit computation of ULP derivatives, and still leverage the global stability and robustness properties of DT methods that are lacking in forward simulation-based methods [21]. Additionally, it is straightforward to include a low-dimensional parameterization of $\boldsymbol{u}$. In many cases, the LLP is effectively a feasibility problem. However, care must be taken to ensure that there exists a solution to LLP or some coordination/penalty method is utilized.

**Fixed parameters.**

The second scheme is applicable to the class of NLDO problems where for fixed values of $\boldsymbol{x}_p$, the resulting subproblem is a convex LQDO problem. The two-level architecture is constructed with only the parameters and associated problem elements in ULP, while the LLP with respect to $\boldsymbol{u}$ and $\boldsymbol{\xi}$ is an LQDO problem. This partitioning allows for the LLP to utilize LQDO-based tools and QP solvers. Typically the set of parameters and associated ULP constraints are relatively small compared to the size of the mesh-dependent control and state DT variables, as well as the number of defect and path constraints. Furthermore, while general NLDO problems are non-convex, many common LQDO formulations are convex. Therefore, we can implement efficient solution strategies in the LLP and focus global search procedures in the ULP which has a much smaller design space to consider. However, this approach may still have issues with feasibility of the LLP, derivative computations in the ULP, and may be less efficient computationally depending of the particular problem [59].

**ALGORITHM 1:** Quasilinearization.

**Input :** $\mathbf{x}^0$ – initial reference solution
$k_{\max}$ – maximum number of iterations
$\epsilon$ – convergence tolerance
**Output:** $\mathbf{x}^*$ – final (optimal) solution

$k \leftarrow 0$ // initialize iteration counter
$v\left(\mathbf{x}^{k-1}\right) \leftarrow \infty$ // initialize objective value
**while** $\left|v\left(\mathbf{x}^k\right) - v\left(\mathbf{x}^{k-1}\right)\right| \leq \epsilon$ or $k \leq k_{\max}$ **do**
  $k \leftarrow k + 1$ // update iteration counter
  $\{\mathbf{f}, \mathbf{H}, \mathbf{A_h}, \mathbf{b_h}, \mathbf{A_g}, \mathbf{b_g}\} \leftarrow$ Construct QP matrices using the RLQDO problem and selected DT method with $\mathbf{x}^{k-1}$
  $\mathbf{x}^k \leftarrow$ Solve QP defined by $\{\mathbf{f}, \mathbf{H}, \mathbf{A_h}, \mathbf{b_h}, \mathbf{A_g}, \mathbf{b_g}\}$
**end**
$\mathbf{x}^* \leftarrow \mathbf{x}^k$ // final solution (optimal if $k < k_{\max}$)

This approach has been studied previously under the term nested (control) co-design [24, 26, 34, 59], and the aforementioned class of problems have been termed *LQDO-amendable co-design problems* [34] and discussed in Eqs. (1.11,1.12). References [59] include comparisons showing that the nested method as described here had lower computational expense than a simultaneous approach, i.e., Eq. (1.8). There are several co-design studies that have effectively utilized the nested approach including attitude control of a distributed actuation system on a satellite [70], active vehicle suspensions [34], and horizontal-axis wind turbines [40].

### 3.3.3 Quasilinearization Methods

The methods described in the previous two sections did not utilize the RLQDO problem discussed in Sec. 3.2. Here we will discuss a class of methods, frequently termed *quasilinearization* [44, 60–62, 71], that utilizes reference trajectories to solve the original NLDO problem. Generally, these methods construct RLQDO subproblems that are successively solved where the previous iteration's solution is the reference trajectory for the next iteration. Under certain conditions, convergence of the solutions of the RLQDO subproblems implies that the result is good approximation of a feasible minimizer in both RLQDO and original NLDO problems [38, 44]. A basic quasilinearization algorithm for NLDO is shown in Alg. 1.

In general, a good initial reference solution is preferred, otherwise, there may be slow convergence, convergence to an undesirable local minima, or divergence [38, 44, 61, 64]. Because of the LQDO-nature of the subproblems, sparse Hessians and Jacobians are readily determined [61]. Quasilinearization can be implemented for a variety of different defect constraint and quadrature schemes [38, 62]. Typically, parameters $\boldsymbol{x}_p$ are not directly studied, but their inclusion is fairly straightforward with the primary difference being a reference value rather than a reference trajectory. Now four different variations of quasilinearization will be discussed.

**Dynamics only.**

Perhaps the most commonly used form of quasilinearization is when only the dynamics are linearized as in Eq. (3.6). In many NLDO problems, the dynamics are the most challenging problem element because of their non-convex structure [38]. Frequently, assumptions are made so that other problem elements are OLQ [60]. In these cases, the OLQ elements can be directly incorporated in the same manner as Sec. 3.3.1, while LQDO-based methods for constructing the defect constraints can be used at each iteration to efficiently update the matrices based on the previous iteration's solution. This successive linear approximation of the dynamics results in a particularly advantageous constraint form that can be effectively utilized in specialized solvers, such a sequential convex programming [38] and Riccati difference equations [46].

**All constraints.**

In this case, all constraints are linearized with Eqs. (3.5b)–(3.5d), including terminal state constraints [61] and state-control path constraints [62]. Generally, the same methods can be applied as the dynamics-only case, but an issue of inconsistencies of the RLQDO constraints can be exacerbated. Constraint relaxation and trust-region methods could be utilized to prevent this failure condition (and many require the solution of additional QPs) [37, 52].

**Feasible point.**

Here we ignore the objective function in order to find a point $\mathbf{x}$ that satisfies all constraints in the NLDO problem [53]. In this form, we are attempting to solve a sequence of linear systems potentially with inequalities. However, these are frequently under-determined equations so additional measures should be taken to ensure convergence and uniqueness [37]. The result can be used as the initial point for the basic quasilinearization algorithm or any general NLP solver.

**Entire NLDO problem.**

The RLQDO in Prob. (3.5) constructed an LQDO problem with respect to the reference solution $\bar{\boldsymbol{x}}$. Therefore, a solution strategy for the entire NLDO problem is Alg. 1. However, in addition the potential inconsistencies in the constraints, the RLQDO problem might result in a non-convex QP. Quasi-Newton methods can be alternative strategy, and defines a different QP problem than the one created from Prob. (3.5) [20, 37]. Alternative linear approximations other than a Taylor series can be used [63]. If only linear approximations are used in the objective function, the problem is a linear program.

All the previously discussed implementations of quasilinearization have ignored an important fact that because we are solving a constrained finite-dimensional optimization problem when using DT methods, the optimality conditions are the Karush-Kuhn-Tucker conditions [9], which include the Lagrange multipliers as shown in Eq. (3.10). Utilizing the Lagrangian instead of $v(\cdot)$ provides curvature information from both objective and constraints [52]. Sequential quadratic programming (SQP) is a popular family of methods that directly incorporate the Lagrangian and previous solution estimates for $\boldsymbol{\lambda}$ and $\boldsymbol{\sigma}$ [20, 37, 52, 72]. The SQP subproblem for Prob. (2.14) is:

$$\min_{\mathbf{s}} \quad v\left(\bar{\mathbf{x}}\right) + \left[\boldsymbol{J}_{\mathbf{x}}^{v}\left(\bar{\mathbf{x}}\right)\right]^{\mathsf{T}} \mathbf{s} + \frac{1}{2}\mathbf{s}^{\mathsf{T}} \boldsymbol{H}_{\mathbf{x}}^{\mathcal{L}}\left(\bar{\mathbf{x}}, \bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\sigma}}\right) \mathbf{s} \tag{3.13a}$$

$$\text{subject to:} \quad \mathbf{h}\left(\bar{\mathbf{x}}\right) + \left[\boldsymbol{J}_{\mathbf{x}}^{\mathbf{h}}\left(\bar{\mathbf{x}}\right)\right]^{\mathsf{T}} \mathbf{s} = \mathbf{0} \tag{3.13b}$$

$$\mathbf{g}\left(\bar{\mathbf{x}}\right) + \left[\boldsymbol{J}_{\mathbf{x}}^{\mathbf{g}}\left(\bar{\mathbf{x}}\right)\right]^{\mathsf{T}} \mathbf{s} \leq \mathbf{0} \tag{3.13c}$$

$$\text{where:} \quad \mathbf{s} = \mathbf{x} - \bar{\mathbf{x}} \tag{3.13d}$$

SQP methods have been long studied in the context of DT methods [20, 37, 53, 64, 65].

Now an interesting connection can be made between the QP generated for a RLQDO problem and the SQP subproblem for the original NLDO problem in Eq. (2.14). Consider the case when the discretization matrices in Eq. (2.7) are the same for $\bar{\mathbf{x}}$ and an appropriately interpolated $\bar{\boldsymbol{x}}$. If we introduce a linear transformation $\boldsymbol{s} = \boldsymbol{x} - \bar{\boldsymbol{x}}$ to the RLQO problem, then the resulting QP will have the same constraints as the SQP subproblem. This is due to the linear summation structure of the defect constraints. This implies that SQP methods on DT-based NLPs are really quasilinearization methods based on a RLQDO problem that utilize additional Hessian information from the constraints (as well as the other techniques that make SQP methods so effective). Because of this, LQDO-based methods for defining the problem elements for the SQP subproblems can be directly used. Exact computations for the additional Hessian information can also be incorporated with modifications to the LQDO discretization methods (considering the additional terms have the same sparsity structure as the original Hessian) [64, 73]. Full implementation of SQP methods using `DTQP` is left as future work.

## 3.4 Method Options

In this section, we summarize the methods that we implemented and compared. The four option fragments are now described.

1. *NLDO algorithm*. IP denotes use of the interior-point method in `fmincon` to solve the NLDO problem. QLIN denotes the quasilinearization method presented in Alg. 1. The RLQDO problem is automatically created using symbolic differentiation. TLFP denotes a two-level fixed parameter method in Sec. 3.3.2 was used to solve the NLDO problem (i.e., the nested control co-design solution strategy). The LQDO LLP is created and solved using `DTQP`, while the ULP is solved using `fmincon` with the interior-point algorithm. Finite-difference methods are used to obtain derivatives in ULP.
2. *OLQ elements*. OLQ denotes the OLQ elements are directly incorporated as discussed in Sec. 3.3.1. NL denotes all problem elements were treated as nonlinear except the simple bounds.

3. *Derivatives*. SD denotes symbolic derivatives are provided. Problem element derivatives are automatically determined using symbolic differentiation and used to construct the required differentiation matrices for the select DT method. FD denotes a forward finite-difference scheme was used to compute any required derivatives.
4. *DT methods*. TR denotes defect constraints constructed using the trapezoidal rule (a SS DT method), and the composite trapezoidal rule was used for quadrature. PS denotes defect constraints constructed using the single-interval Legendre pseudospectral method on a Lagrange-Gauss-Lobatto mesh and Gaussian quadrature weights [74]. For both methods, the catenated number indicates the number of points in the mesh.

These four options are combined to create a complete NLDO solution strategy. For example, IP-SD-OLQ-TR200 indicates that the NLP problem is constructed using the trapezoidal rule with 200 time points and symbolic derivatives with direct incorporation of OLQ elements and solved using an interior-point method. The computational cost is measured in several parts. $T_{sym}$ indicates the time spent determining the symbolic derivatives. The total time is denoted $T$, which the combination of the initialization time $T_{int}$ (which includes the time to create the OLQ element matrices) and optimizer time $T_{opt}$. The symbolic computations are not included in the total time so a fair comparison can be made between the methods. All the method and case study details are available in the free and open-source `DTQP` software tool [54]. The computer architecture used for all case study results was a desktop workstation with an AMD 3970X CPU at 3.7 GHz, 128 GB DDR4 2666 MHz RAM, `MATLAB` 2020a update 4, and `Windows` 10 build 17763.1397.

## 3.5 Case Studies

In this section, four different case studies are presented to demonstrate how various LQDO-based methods can be incorporated to solve NLDO problems.

### 3.5.1 Container Crane

**Problem description.**

In this problem, we want to optimally transfer the states of a container crane subject to linear path constraints. Please see Ref. [75] and is also known as `cran` in Ref. [42]. The NLDO problem is:

$$\min_{\boldsymbol{u},\boldsymbol{\xi}} \quad \frac{1}{2}\int_0^{t_f} \left[\xi_3^2 + \xi_6^2 + \rho\left[u_1^2 + u_2^2\right]\right] dt \tag{3.14a}$$

$$\text{subject to:} \quad \dot{\boldsymbol{\xi}} = \begin{bmatrix} \xi_4 \\ \xi_5 \\ \xi_6 \\ u_1 + c_4\xi_3 \\ u_2 \\ -\left[u_1 + c_5\xi_3 + 2\xi_5\xi_6\right]/\xi_2 \end{bmatrix} \tag{3.14b}$$

$$\boldsymbol{\xi}(0) = (0, 22, 0, 0, -1, 0) \tag{3.14c}$$

$$\boldsymbol{\xi}(t_f) = (10, 14, 0, 2.5, 0, 0) \tag{3.14d}$$

$$-c_6 \le \xi_4 \le c_6, \quad c_7 \le \xi_5 \le c_7 \tag{3.14e}$$

$$-c_1 \le u_1 \le c_1, \quad c_2 \le u_2 \le c_3 \tag{3.14f}$$

Note that all parts except for $\dot{\xi}_6$ are OLQ elements. The constants are $c_1 = 2.83374, c_2 = -0.80865, c_3 = 0.71265, c_4 = 17.2656, c_5 = 27.0756, c_6 = 2.5, c_7 = 1, t_f = 9,$ and $\rho = 0.01$. The initial guess used for all methods was a linear solution between the specified initial and final states values and maximum and minimum control values (see Fig. 3.1).

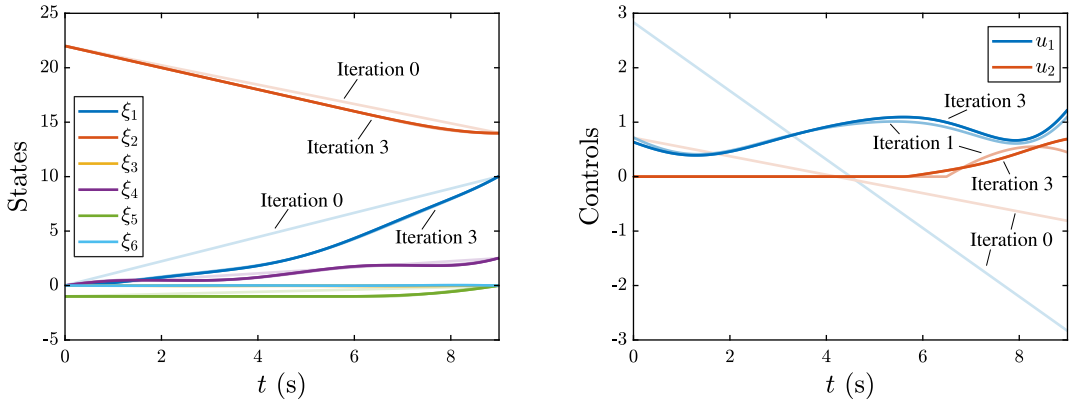
**Results.**

The results for this case study are summarized in Table 3.1.

We will first focus on the effectiveness of the quasilinearization method. In most of the cases, QLIN had a lower computational cost than the best IP method while converging to nearly the same solution. For example, QLIN was 2.6× faster for the TR20 and 1.9× faster for the PS40 variant.

**Table 3.1:** Results for the Container Crane case study.

| Method | $v$ | Iter. | $T_{sym}$ | $T_{int}$ | $T_{opt}$ | $T$ |
|---|---|---|---|---|---|---|
| QLIN-TR20 | 0.0380 | 3 | 0.57 | 0.007 | 0.03 | 0.03 |
| IP-SD-OLQ-TR20 | 0.0380 | 20 | 1.42 | 0.002 | 0.08 | 0.08 |
| IP-SD-NL-TR20 | 0.0380 | 20 | 2.17 | 0.003 | 0.16 | 0.16 |
| IP-FD-OLQ-TR20 | 0.0380 | 22 | 0.00 | 0.003 | 0.22 | 0.22 |
| IP-FD-NL-TR20 | 0.0380 | 22 | 0.00 | 0.001 | 0.24 | 0.24 |
| QLIN-TR200 | 0.0375 | 3 | 0.56 | 0.010 | 0.31 | 0.32 |
| IP-SD-OLQ-TR200 | 0.0375 | 29 | 1.41 | 0.002 | 0.41 | 0.41 |
| IP-SD-NL-TR200 | 0.0375 | 29 | 2.17 | 0.003 | 0.58 | 0.59 |
| IP-FD-OLQ-TR200 | 0.0375 | 30 | 0.00 | 0.003 | 0.88 | 0.89 |
| IP-FD-NL-TR200 | 0.0375 | 29 | 0.00 | 0.002 | 0.95 | 0.95 |
| QLIN-TR2000 | 0.0375 | 3 | 0.56 | 0.033 | 8.23 | 8.26 |
| IP-SD-OLQ-TR2000 | 0.0375 | 35 | 1.42 | 0.008 | 7.21 | 7.21 |
| IP-SD-NL-TR2000 | 0.0375 | 35 | 2.18 | 0.004 | 8.25 | 8.26 |
| IP-FD-OLQ-TR2000 | 0.0375 | 39 | 0.00 | 0.003 | 13.92 | 13.92 |
| IP-FD-NL-TR2000 | 0.0375 | 35 | 0.00 | 0.003 | 13.95 | 13.95 |
| QLIN-PS10 | 0.0376 | 4 | 0.55 | 0.009 | 0.03 | 0.04 |
| IP-SD-OLQ-PS10 | 0.0376 | 24 | 1.42 | 0.002 | 0.08 | 0.08 |
| IP-SD-NL-PS10 | 0.0376 | 28 | 2.17 | 0.003 | 0.21 | 0.21 |
| IP-FD-OLQ-PS10 | 0.0376 | 29 | 0.00 | 0.003 | 0.25 | 0.26 |
| IP-FD-NL-PS10 | 0.0376 | 31 | 0.00 | 0.002 | 0.31 | 0.31 |
| QLIN-PS40 | 0.0375 | 3 | 0.56 | 0.008 | 0.18 | 0.18 |
| IP-SD-OLQ-PS40 | 0.0375 | 29 | 1.42 | 0.002 | 0.34 | 0.34 |
| IP-SD-NL-PS40 | 0.0375 | 29 | 2.18 | 0.002 | 0.46 | 0.46 |
| IP-FD-OLQ-PS40 | 0.0375 | 29 | 0.00 | 0.002 | 0.53 | 0.53 |
| IP-FD-NL-PS40 | 0.0375 | 31 | 0.00 | 0.002 | 0.62 | 0.62 |




**(a)** States. **(b)** Controls.

**Figure 3.1:** Quasilinearization results for the Container Crane problem (QLIN-TR2000).

There was some additional cost to create the OLQ matrices, but the overall solver time was faster. However, when the number of time points was large in the TR2000 variant, the QLIN method was 1.2× slower (but still competitive with all other variants). This indicates that further studies are needed to determine what the best uses are, especially since this problem was a prime candidate for the QLIN method with many OLQ elements. The different QLIN-TR2000 iterations are shown in Fig. 3.1 where the darker lines indicate later iterations. We note that the initial guess was quite poor for the controls, but reasonable for the states. However, only after a single iteration, all trajectories were nominally similar to their final convergence counterparts.

Now let us turn our focus to the impact of the direct incorporation of OLQ elements. As was anticipated, in all cases, the computational cost was lower when the OLQ matrices were included for the appropriate problem elements, and each variant converged to the same solution with the same DT method specification. In some cases, the iterations were different, potentially due to the differences in the calculated derivatives. The benefit of the OLQ option with the symbolic derivatives provided was between 15–163%, while for the tested finite-difference variants, it was between 0–19% faster. The wide range with SD can be attributed to the fact that the optimization algorithm computational cost starts to dominate when the problem size grows; thus, the marginal gains recorded with TR2000. The minor benefit for the FD variants is because there is generally a high fixed cost with computing the finite differences for this relatively small problem. This illustrates the point that there are clear advantageous to allowing the user of a DT-based tool to define the OLQ elements directly and create the underlying methods that leverage these elements.

A final point of emphasis is the fact that computational costs of direct incorporation of the OLQ elements were minimal. All values of $T_{int}$ were quite small when compared to the other computational costs. Even the QLIN methods, which had to create large matrices with time-varying elements, had a much lower cost than the actual optimization algorithms and symbolic operations. Therefore, there are limited disadvantages in OLQ when properly implemented.

### 3.5.2 Van der Pol Oscillator

**Problem description.**

The Van der Pol oscillator is a common test problem for many numerical optimal control methods (see `vpol` in Ref. [42]), including quasilinearization [60, 61]. Here we consider two variations: V1 $= \{t_f = 5, c_1 = -\infty, c_2 = -0.3, c_3 = 1, c_4 = 1, c_5 = 1\}$ which has fixed parameter values [76], and V2 $= \{t_f = 5, c_1 = -0.4, c_2 = -0.5, c_3 = 1, c_4 = 0.1, c_5 = 5\}$ which must determine two parameter values [5]. The NLDO problem is:

$$\min_{\boldsymbol{u},\boldsymbol{\xi},\boldsymbol{x}_p} \quad \int_0^{t_f} \left[\xi_1^2 + \xi_2^2 + u_1^2\right] dt \tag{3.15a}$$

$$\text{subject to:} \quad \dot{\boldsymbol{\xi}} = \begin{bmatrix} \xi_2 \\ -p_1\xi_1 + \xi_2\left[1 - p_2\xi_1^2\right] + u_1 \end{bmatrix} \tag{3.15b}$$

$$\boldsymbol{\xi}(0) = (1, 0) \tag{3.15c}$$

$$c_1 \le \xi_2, \quad c_2 \le u(t) \le c_3 \tag{3.15d}$$

$$c_4 \le p_1 \le c_5, \quad c_4 \le p_2 \le c_5 \tag{3.15e}$$

Note that all problem elements except for $\dot{\xi}_2$ are OLQ elements. The linearized form for $\dot{\xi}_2$ was shown in Eq. (3.8). The initial guess for both variations and all methods can be seen in Fig. 3.3.

**Results.**

For V1, similar results to Container Crane are observed. This problem has been well studied as a convergent quasilinearization example [60, 61]. More interesting observations can be made for V2, and the results are summarized in Table 3.2. A very accurate solution is obtained using IP-SD-OLQ-PS40 in only 0.07 s, and is shown in Fig. 3.2 with optimal parameter values of 0.15206 and 1.5374, respectively.

Here, the basic quasilinearization method *fails to converge* (and was tested with many different initial guesses). The iteration behavior is shown in Fig. 3.3 where after a few iterations, the solution oscillates between two trajectories. Both state and path constraints enter and exit activity as well as the parameters' simple bound constraints. The inclusion of more advanced techniques such as

**Table 3.2:** Results for the Van der Pol Oscillator V2 case study.

| Method | $v$ | Iter. | $T_{sym}$ | $T_{int}$ | $T_{opt}$ | $T$ |
|---|---|---|---|---|---|---|
| IP-SD-OLQ-TR20 | 1.9809 | 15 | 0.85 | 0.001 | 0.04 | 0.04 |
| IP-SD-NL-TR20 | 1.9809 | 15 | 1.14 | 0.001 | 0.06 | 0.06 |
| IP-FD-OLQ-TR20 | 1.9809 | 21 | 0.00 | 0.001 | 0.10 | 0.10 |
| IP-FD-NL-TR20 | 1.9809 | 18 | 0.00 | 0.001 | 0.09 | 0.09 |
| IP-SD-OLQ-TR200 | 1.9652 | 23 | 0.86 | 0.002 | 0.12 | 0.12 |
| IP-SD-NL-TR200 | 1.9652 | 23 | 1.22 | 0.001 | 0.16 | 0.16 |
| IP-FD-OLQ-TR200 | 1.9652 | 26 | 0.00 | 0.001 | 0.18 | 0.18 |
| IP-FD-NL-TR200 | 1.9652 | 27 | 0.00 | 0.001 | 0.25 | 0.25 |
| IP-SD-OLQ-TR2000 | 1.9650 | 26 | 0.86 | 0.003 | 1.10 | 1.10 |
| IP-SD-NL-TR2000 | 1.9650 | 26 | 1.17 | 0.002 | 1.21 | 1.21 |
| IP-FD-OLQ-TR2000 | 1.9650 | 27 | 0.00 | 0.002 | 1.52 | 1.52 |
| IP-FD-NL-TR2000 | 1.9650 | 28 | 0.00 | 0.002 | 2.22 | 2.22 |
| IP-SD-OLQ-PS10 | 1.9649 | 17 | 0.85 | 0.002 | 0.04 | 0.04 |
| IP-SD-NL-PS10 | 1.9649 | 17 | 1.15 | 0.001 | 0.06 | 0.06 |
| IP-FD-OLQ-PS10 | 1.9649 | 22 | 0.00 | 0.001 | 0.07 | 0.07 |
| IP-FD-NL-PS10 | 1.9649 | 29 | 0.00 | 0.001 | 0.12 | 0.12 |
| IP-SD-OLQ-PS40 | 1.9650 | 16 | 0.86 | 0.002 | 0.07 | 0.07 |
| IP-SD-NL-PS40 | 1.9650 | 16 | 1.15 | 0.001 | 0.09 | 0.09 |
| IP-FD-OLQ-PS40 | 1.9650 | 17 | 0.00 | 0.001 | 0.09 | 0.09 |
| IP-FD-NL-PS40 | 1.9650 | 19 | 0.00 | 0.001 | 0.13 | 0.13 |
| QLIN-TRX | — | did not converge | | | | |
| QLIN-PSX | — | did not converge | | | | |

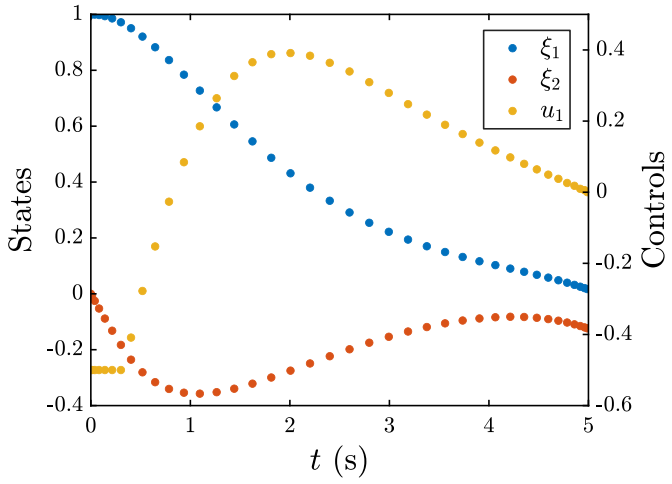
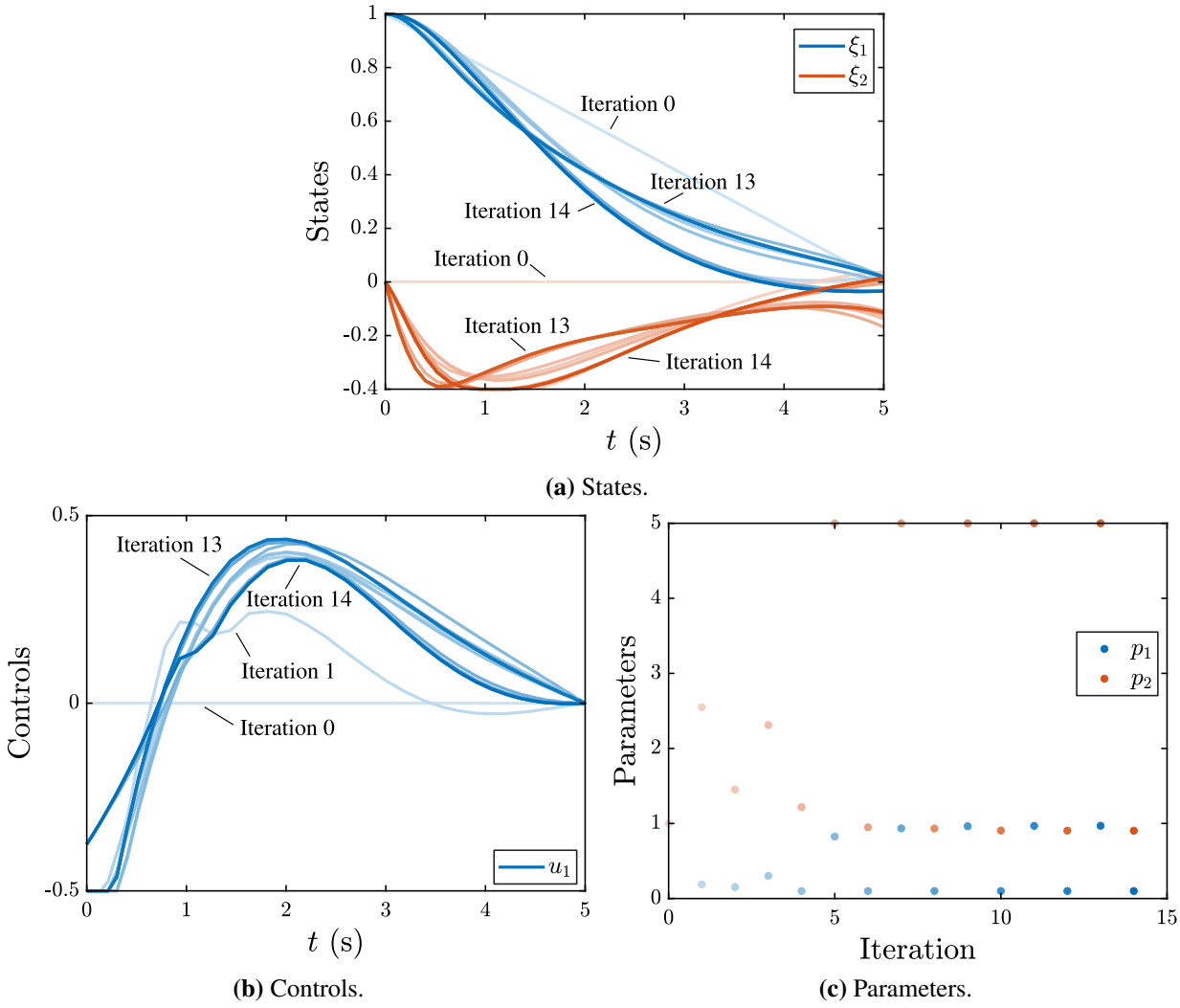



**Figure 3.2:** Results for the Van der Pol Oscillator problem V2 (IP-SD-OLQ-PS40).




**(a)** States.

**(b)** Controls.

**(c)** Parameters.

**Figure 3.3:** Quasilinearization results for the Van der Pol Oscillator problem V2 QLIN-PS40).

line searches and trust regions could improve convergence, but, as was discussed in Sec. 8, the modified approach would be similar to existing methods for solving NLPs.

Again, the use of the OLQ option resulted in faster overall solving times in all cases. Computational costs were decreased between 10–50% with SD and between $-10\%$ and 71% with FD. The small negative performance hit was due to the increased iterations needed by the OLQ option vs. NL, again indicating that the derivatives are different between the two.

**Table 3.3:** Results for the Co-Design Transfer case study.

| Method | $v$ | Iter. | $T_{sym}$ | $T_{int}$ | $T_{opt}$ | $T$ |
|---|---|---|---|---|---|---|
| TLFP-TR2000 | 23.6228 | 6 | — | — | — | 0.31 |
| QLIN-TR2000 | 23.6228 | 8 | 0.25 | 0.031 | 0.17 | 0.20 |
| IP-SD-OLQ-TR2000 | 23.6228 | 6 | 0.31 | 0.002 | 0.14 | 0.14 |
| IP-SD-NL-TR2000 | 23.6228 | 6 | 0.47 | 0.001 | 0.16 | 0.16 |
| IP-FD-OLQ-TR2000 | 23.6228 | 8 | 0.00 | 0.003 | 0.58 | 0.58 |
| IP-FD-NL-TR2000 | 23.6228 | 8 | 0.00 | 0.003 | 0.61 | 0.61 |

### 3.5.3 Co-Design Transfer

**Problem description.**

This is a control co-design test problem with a known solution [59]. It has been previously used to study the TLFP method. The NLDO problem is:

$$\min_{\boldsymbol{u},\boldsymbol{\xi},\boldsymbol{x}_p} \quad \int_0^{t_f} u_1^2 dt \tag{3.16a}$$

$$\text{subject to:} \quad \dot{\boldsymbol{\xi}} = \begin{bmatrix} \xi_2 \\ -p_1\xi_1 + u_1 \end{bmatrix} \tag{3.16b}$$

$$\boldsymbol{\xi}(0) = (x_0, v_0) \tag{3.16c}$$

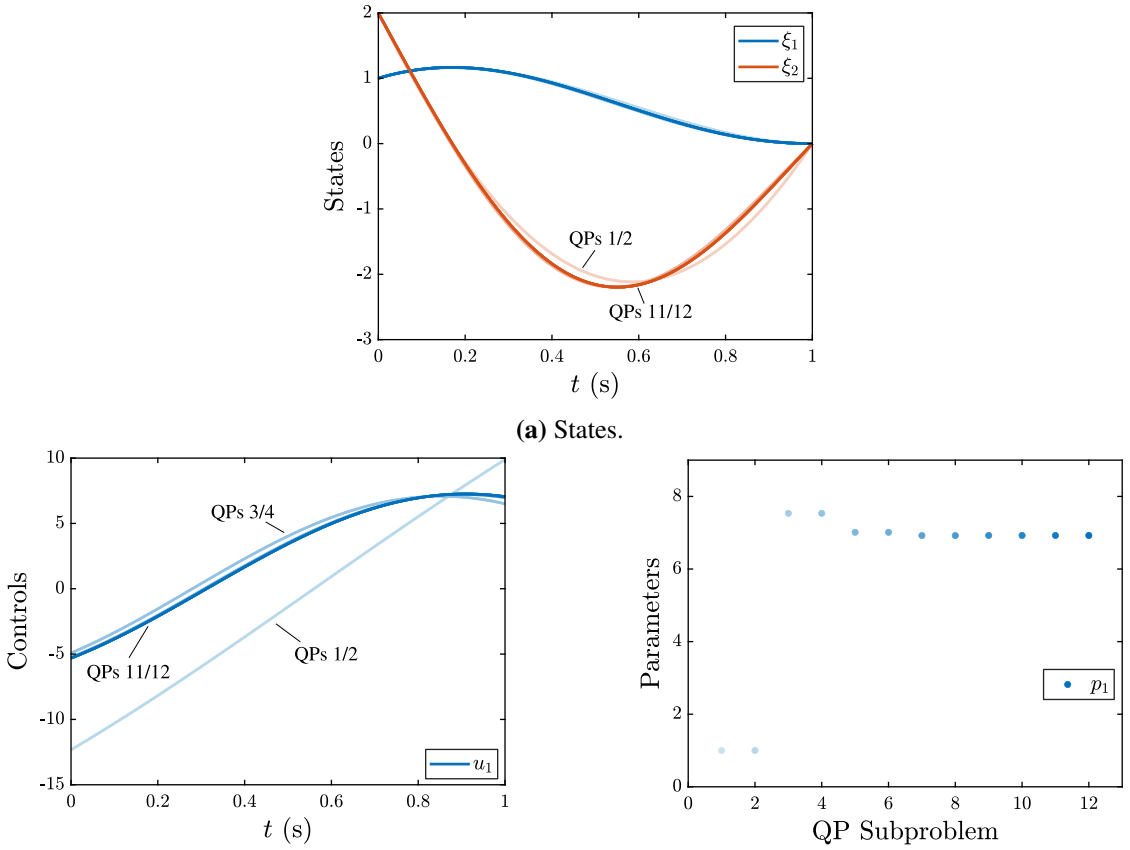
$$\boldsymbol{\xi}(t_f) = (0, 0) \tag{3.16d}$$

$$0 \leq p_1 \tag{3.16e}$$

Note that all parts except for $\dot{\xi}_2$ are OLQ elements. The constants are $t_f = 1, x_0 = 1$, and $v_0 = 2$. A poor initial guess of one was used for all optimization variables.

**Results.**

The results for this case study are summarized in Table 3.3. This is the first of the case study problems that is an LQDO-amenable co-design problem; thus, TLFP is applicable. In Table 3.3, this strategy is slower than QLIN and IP-SD-OLQ. However, it can be argued that comparisons to FD are more appropriate because TLFP does not require any symbolic operations. Therefore, comparing TLFP and IP-FD-OLQ, we see that the two-level method is faster or only slightly slower. Even if all OLQ elements are being directly incorporated, there are still some problem




**(a)** States.

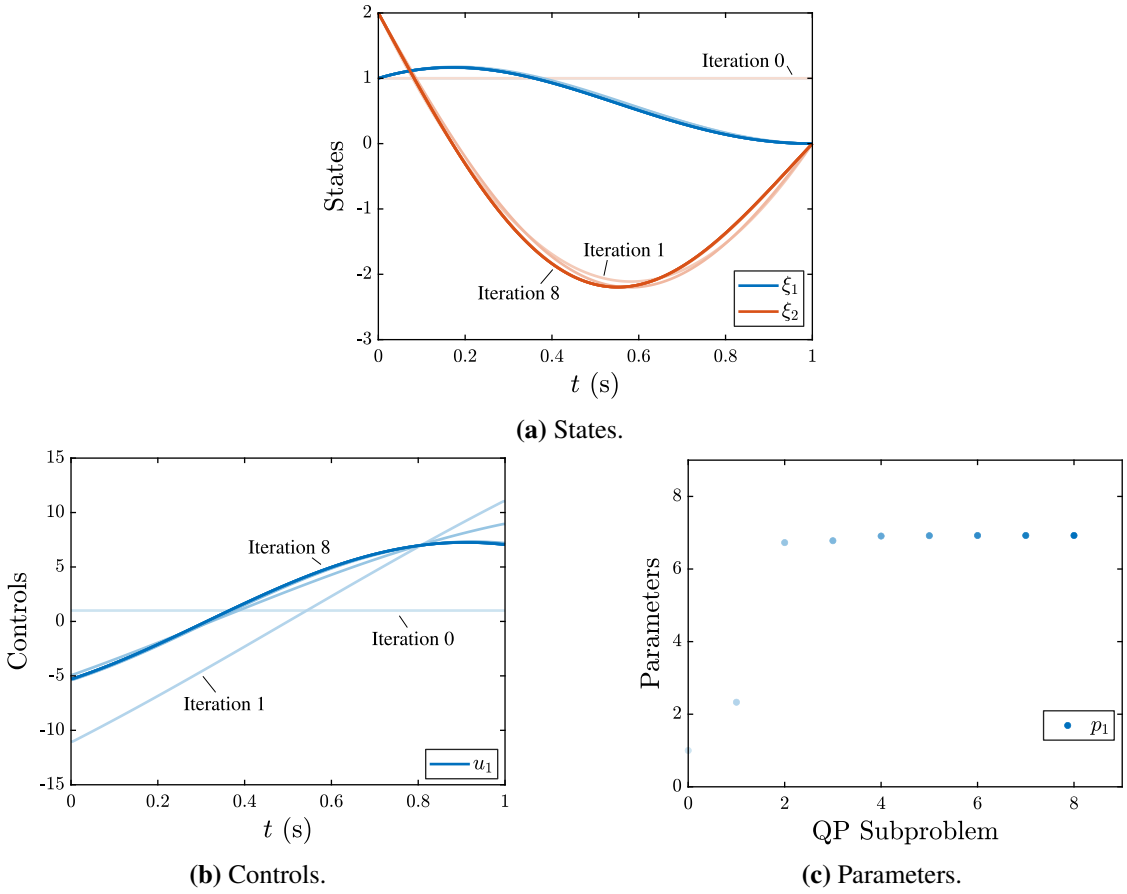
**(b)** Controls.

**(c)** Parameters.

**Figure 3.4:** Two-level fixed parameter results for the Co-Design Transfer problem (TLFP-TR2000).

elements with nonlinear behavior. If a user identifies that their problem is appropriate for the TLFP method, then there still might be some place for the nested co-design method when analytic derivatives are unavailable. This will be further studied in the next example.

The iteration sequences of TLFP-TR2000 and QLIN-TR2000 are shown in Figs. 3.4 and 3.5, respectively. Each method requires the solution of a different sequence of QP subproblems (LQDO problems). Note that each iteration of TLFP has feasible dynamics with respect to the original problem, while this is not necessarily true for QLIN. The finite-difference nature of the TLFP method can also be observed.




**(a)** States.

**(b)** Controls.

**(c)** Parameters.

**Figure 3.5:** Quasilinearization results for the Co-Design Transfer problem (QLIN-TR2000).

### 3.5.4 Vehicle Suspension Co-design

**Problem description.**

This is a simplified quarter-car vehicle suspension problem, illustrated in Fig. 3.6. Similar versions of this problem have been studied in Refs. [7, 34, 77]. The four states represent: 1) the difference in the unsprung mass and road elevation $z_0$, 2) velocity of the unsprung mass, 3) difference in the mass positions, and 4) velocity of the spring mass. The single open-loop control variable represents the active force actuator in the suspension, and the two parameters are for the suspension damper and spring constants, respectively.

**Table 3.4:** Results for the Active Suspension Problem case study.

| Method | $v$ | Iter. | $T_{sym}$ | $T_{int}$ | $T_{opt}$ | $T$ |
|---|---|---|---|---|---|---|
| TLFP-TR200 | 1.977 | 17 | — | — | — | 1.10 |
| IP-SD-OLQ-TR200 | 1.977 | 12 | 6.46 | 0.003 | 0.3 | 0.31 |
| IP-SD-NL-TR200 | 1.977 | 12 | 6.47 | 0.003 | 0.3 | 0.33 |
| IP-FD-OLQ-TR200 | 2.048 | 598 | 0.00 | 0.003 | 21.8 | 21.80 |
| IP-FD-NL-TR200 | 2.048 | 598 | 0.00 | 0.003 | 21.9 | 21.92 |
| TLFP-TR2000 | 1.996 | 22 | — | — | — | 9.18 |
| IP-SD-OLQ-TR2000 | 1.996 | 13 | 6.49 | 0.005 | 1.6 | 1.63 |
| IP-SD-NL-TR2000 | 1.996 | 13 | 6.51 | 0.004 | 1.7 | 1.74 |
| IP-FD-OLQ-TR2000 | 1.997 | 599 | 0.00 | 0.006 | 216.7 | 216.68 |
| IP-FD-NL-TR2000 | 1.997 | 599 | 0.00 | 0.004 | 216.4 | 216.43 |
| QLIN-TRX | — | did not converge | | | | |

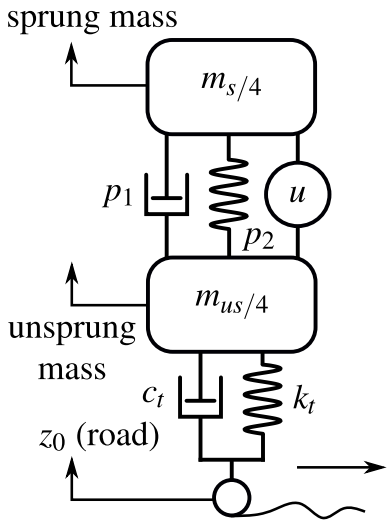



**Figure 3.6:** Quarter car suspension

.

Briefly:

$$\min_{u,\boldsymbol{\xi},\boldsymbol{x}_p} \quad \int_0^{t_f} \left[ w_1 \xi_1^2 + w_2 \dot{\xi}_4^2 + w_3 u^2 \right] dt \tag{3.17a}$$

$$\text{subject to:} \quad \dot{\boldsymbol{\xi}}(t) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{-k_t}{m_{us/4}} & \frac{-[p_1+c_t]}{m_{us/4}} & \frac{p_2}{m_{us/4}} & \frac{p_1}{m_{us/4}} \\ 0 & -1 & 0 & 1 \\ 0 & \frac{p_1}{m_{s/4}} & \frac{-p_2}{m_{s/4}} & \frac{-p_1}{m_{s/4}} \end{bmatrix} \boldsymbol{\xi}(t) + \begin{bmatrix} 0 \\ \frac{-1}{m_{us/4}} \\ 0 \\ \frac{1}{m_{s/4}} \end{bmatrix} u(t) + \begin{bmatrix} -1 \\ \frac{c_t}{m_{us/4}} \\ 0 \\ 0 \end{bmatrix} \dot{z}_0(t) \tag{3.17b}$$

$$\boldsymbol{\xi}(0) = \mathbf{0} \tag{3.17c}$$

$$|\xi_3(t)| \leq r_{\max} \tag{3.17d}$$

$$\boldsymbol{x}_{p\,\min} \leq \boldsymbol{x}_p \leq \boldsymbol{x}_{p\,\max} \tag{3.17e}$$

where objective in Eq. (3.17a) is a weighted sum of handling, comfort, and control effort metrics, and Eq. (3.17d) represents a rattle space path constraint. The constants are:

$$w_1 = 10^5, w_2 = 0.5, w_3 = 10^{-5}, k_t = 232500, c_t = 0, m_{us/4} = 65, m_{s/4} = 325, r_{\max} = 0.04$$

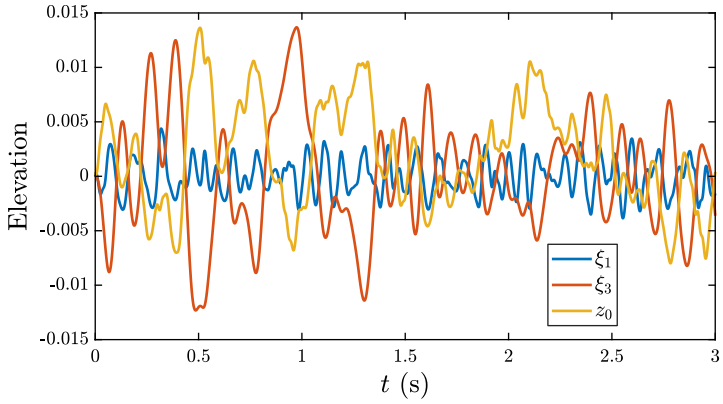
$$\boldsymbol{x}_{p\,\min} = \left[10^2, 10^2\right], \boldsymbol{x}_{p\,\max} = \left[10^5, 10^6\right]$$

The rough road profile used is from Refs. [7, 34] and is shown in Fig. 3.7a. The TLFP method is applicable [34].

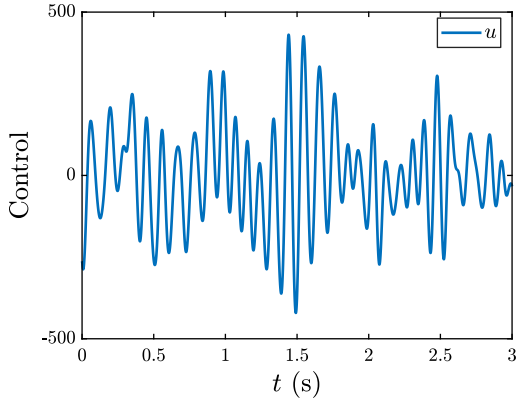
**Results.**

The results for this case study are summarized in Table 3.4. The optimal solution using TLFP-TR2000 for select states and the control are shown in Figs. 3.7a and 3.7b, respectively. Optimal parameter values were found to be 101.12 and 21558. Because a large mesh is needed to accurately solve this problem, the single-interval PS methods were not tested. Similar to Van der Pol Oscillator V2, this problem did not converge when using QLIN.

The main takeaway from this example is the relative effectiveness of the TLFP method. The cases using the symbolic derivatives were significantly faster (3.5–5.6×) than TLFP, but as was stated in Sec. 3.5.3, a fairer comparison is to the FD methods. Now the two-level approach is 20–




**(a)** Select states and road elevation profile.




**(b)** Control.

**Figure 3.7:** Results for the Active Suspension Problem (TLFP-TR2000).

24× faster! Furthermore, the FD methods results have noticeably poorer final solutions, especially for TR200, as well as much slower convergence rates. Therefore, for certain LQDO-amendable co-design problems where the symbolic derivatives are not feasible, the TLFP approach might perform better than the simultaneous approach [34, 59].

Finally, the difference between OLQ and NL was relatively small. In one case it was 7% faster, while in other cases the effect within the margin of error. Only two of the state derivative functions were OLQ elements, and the objective function was nonlinear, so there was little difference between the two options.

---

The main takeaway for solving DO problems from this chapter is that the NL method with high-quality derivative information produces the fastest results, but the bilevel approach is preferred

when this derivative information is not available. In the next chapter, we apply these insights to a more comprehensive case of the suspension problem presented above.

# Chapter 4

# Fair Comparisons between the Nested and Simultaneous Control Co-Design Methods

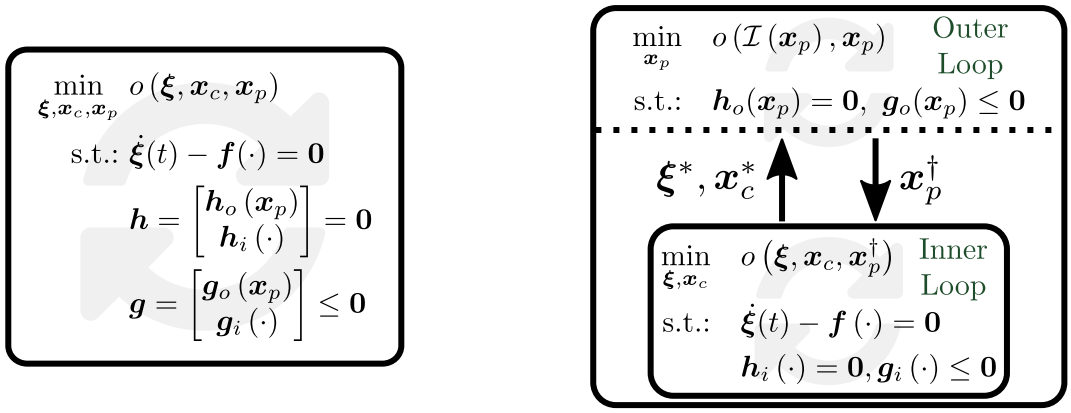
This chapter explores Study 2 proposed in Sec. (1.4.2), and applies the insights developed in Chapter. 3: Using LQDO Elements in solving NLDO Problems from Chap. 3, to carry out a fair comparison between the nested CCD and simultaneous CCD methods[4].

## 4.1 Introduction

Control co-design (CCD) is the term for a class of integrated design methods that concurrently treat the dynamic system's physical and control aspects, overcoming some of the limitations of traditional sequential and siloed approaches [11, 24]. Because of the broad applicability and promise of improved integrated system performance [28], CCD has been adopted by researchers in several areas including automotive [5, 7], thermal management systems [4], spacecraft [70], wave energy [14], and wind energy [31, 36].

There are various solution strategies for CCD that have been studied, but only relatively few studies have made detailed comparisons. Without thorough investigations utilizing state-of-the-art methods and modern design problems, the existing domain-specific suggestions might not apply to today's problems and methods. Such broad and up-to-date investigations are critical as CCD is being used to solve increasingly complex and large-scale system design problems. Such studies could elucidate needed early insights into the appropriate choice of strategy and implementation techniques. The goal of this article is to make a contribution towards this end for a subset of popular CCD coordination and solution strategies on a complex CCD problem. While guidelines applicable to broad classes of CCD problems will not be made, the existing literature-based comparisons and

[4]This chapter is based on the following publication [78].




**(a)** Simultaneous.

**(b)** Nested.

**Figure 4.1:** Two common CCD coordination strategies.

a detailed case study will better demonstrate the state-of-the-art understanding and how a thorough investigation can be conducted to yield the desired implementation insights.

## 4.2 Control Co-Design Strategies and Solution Methods

Here we are considering coordination strategies for CCD problems that can be represented as a deterministic nonlinear dynamic optimization problem (NLDO) [7, 14]. Of late there have been studies that look into coordination strategies for CCD problems with stochastic problem elements as well. Please refer Ref. [79] for more details regarding stochastic CCD formulations. Because of the multidisciplinary nature of these problems, different multidisciplinary design optimization (MDO) architectures can be considered [7, 11, 80]. However, suppose we are limited to single-system problems and architectures that do not partition the system across trajectories. In that case, there are a limited number of appropriate MDO methods suitable for CCD [11, 23]. Because of these reasons and more, the two leading coordination strategies for CCD are 1) simultaneous analysis and design or *simultaneous*, and 2) the nested control problem formulation or simply *nested* [26].

### 4.2.1 Simultaneous Formulation

In the simultaneous CCD formulation, a single optimization problem is put forth, and the optimizer simultaneously analyzes and designs the plant, state, and control variables, as shown in Fig. 4.1a [7, 30]. The general NLDO problem form is shown in Eq. (1.8), but is shown here again in a form more conducive to discussions in this chapter:

$$\min_{\boldsymbol{x}=[\boldsymbol{\xi},\boldsymbol{x}_c,\boldsymbol{x}_p]} o = \int_{t_0}^{t_f} \ell\left(t, \boldsymbol{\xi}, \boldsymbol{x}_c, \boldsymbol{x}_p\right) dt + m\left(\boldsymbol{\xi}_0, \boldsymbol{\xi}_f, \boldsymbol{x}_c, \boldsymbol{x}_p\right) \tag{4.1a}$$

$$\text{subject to: } \dot{\boldsymbol{\xi}}(t) - \boldsymbol{f}(t, \boldsymbol{\xi}, \boldsymbol{x}_c, \boldsymbol{x}_p) = \boldsymbol{0} \tag{4.1b}$$

$$\boldsymbol{h} = \begin{bmatrix} \boldsymbol{h}_o\left(\boldsymbol{x}_p\right) \\ \boldsymbol{h}_i\left(t, \boldsymbol{\xi}, \boldsymbol{x}_c, \boldsymbol{x}_p, \boldsymbol{\xi}_0, \boldsymbol{\xi}_f\right) \end{bmatrix} = \boldsymbol{0} \tag{4.1c}$$

$$\boldsymbol{g} = \begin{bmatrix} \boldsymbol{g}_o\left(\boldsymbol{x}_p\right) \\ \boldsymbol{g}_i\left(t, \boldsymbol{\xi}, \boldsymbol{x}_c, \boldsymbol{x}_p, \boldsymbol{\xi}_0, \boldsymbol{\xi}_f\right) \end{bmatrix} \leq \boldsymbol{0} \tag{4.1d}$$

$$\text{where: } \boldsymbol{\xi}_0 = \boldsymbol{\xi}(t_0),\ \boldsymbol{\xi}_f = \boldsymbol{\xi}(t_f) \tag{4.1e}$$

where $t \in [t_0, t_f]$ is the fixed time horizon, $\{\boldsymbol{\xi}, \boldsymbol{x}_c, \boldsymbol{x}_p\}$ are the collections of the selected states, control design variables, and plant design variables, respectively. The objective function $o(\cdot)$ is composed of the Lagrange term $\ell(\cdot)$ and Mayer term $m(\cdot)$ [19]. The first-order differential equation $\dot{\boldsymbol{\xi}}(t) - \boldsymbol{f}(\cdot) = \boldsymbol{0}$ represents the system dynamics. The equality constraints $\boldsymbol{h}(\cdot)$ are partitioned into two sets $\{\boldsymbol{h}_o(\cdot), \boldsymbol{h}_i(\cdot)\}$ where $\boldsymbol{h}_o(\cdot)$ depends only on $\boldsymbol{x}_p$. A similar partitioned form is used for the inequality constraints $\boldsymbol{g}(\cdot)$.

### 4.2.2 Nested Formulation

The nested CCD formulation is an intentional reorganization of the simultaneous CCD problem in Prob (4.1) as a two-level optimization problem with an outer and inner problem hierarchy, as shown in Fig. 4.1b. This has been discussed in the previous chapters in Eqs. (1.11,1.12). For the

outer-loop problem, we solve only with respect to the plant design variables:

$$\min_{\boldsymbol{x}_p} \quad o\left(\mathcal{I}\left(\boldsymbol{x}_p\right), \boldsymbol{x}_p\right) \tag{4.2a}$$

$$\text{subject to:} \quad \boldsymbol{h}_o(\boldsymbol{x}_p) = \boldsymbol{0} \tag{4.2b}$$

$$\boldsymbol{g}_o(\boldsymbol{x}_p) \leq \boldsymbol{0} \tag{4.2c}$$

$$\text{where:} \quad \mathcal{I}\left(\boldsymbol{x}_p\right) = \underset{\boldsymbol{\xi}, \boldsymbol{x}_c}{\arg\min} \left(\text{Prob. (4.3) with } \boldsymbol{x}_p\right) \tag{4.2d}$$

where $\mathcal{I}\left(\boldsymbol{x}_p\right)$ is the solution to the following inner-loop problem:

$$\min_{\boldsymbol{\xi}, \boldsymbol{x}_c} \quad o\left(\boldsymbol{\xi}, \boldsymbol{x}_c, \boldsymbol{x}_p^{\dagger}\right) \tag{4.3a}$$

$$\text{subject to:} \quad \dot{\boldsymbol{\xi}}(t) - \boldsymbol{f}\left(t, \boldsymbol{\xi}, \boldsymbol{x}_c, \boldsymbol{x}_p^{\dagger}\right) = \boldsymbol{0} \tag{4.3b}$$

$$\boldsymbol{h}_i\left(t, \boldsymbol{\xi}, \boldsymbol{x}_c, \boldsymbol{x}_p^{\dagger}, \boldsymbol{\xi}_0, \boldsymbol{\xi}_f\right) = \boldsymbol{0} \tag{4.3c}$$

$$\boldsymbol{g}_i\left(t, \boldsymbol{\xi}, \boldsymbol{x}_c, \boldsymbol{x}_p^{\dagger}, \boldsymbol{\xi}_0, \boldsymbol{\xi}_f\right) \leq \boldsymbol{0} \tag{4.3d}$$

where $\boldsymbol{x}_p^{\dagger}$ is the candidate plant design from the outer loop. This nested strategy was termed nested control problem formulation because Prob. (4.3) is now in the standard optimal control form where the optimal states and controls are desired. In Refs. [23, 26], it is shown that the solution to Prob. (4.2) is mathematically equivalent to the simultaneous form in Prob. (4.1) under the condition that a solution exists for the inner-loop problem for every considered $\boldsymbol{x}_p$ by the outer loop.

## 4.3 Comparing the Strategies

With both the considered CCD coordination strategies and solution methods described, we can now start comparing the nested and simultaneous strategies.

### 4.3.1 Literature-Based Discussions

In many studies, there is a brief statement on the coordination selection (likely supported by work not directly shown). In Refs. [31, 70, 81, 82], the authors state that the nested approach is better suited for their problem, while Refs. [5, 7, 83] assert the simultaneous approach. For the

studies that selected the nested approach, the main motivating reasons can be summarized as the impractical size and complexity of the simultaneous formulation [27], ability to use tailored inner-loop methods (e.g., LQR and LQDO) [34, 70, 81, 82], and reduction of calls to computationally-expensive plant models [31, 70]. In Ref. [7], the simultaneous approach was selected because the nested implementation used was very computationally inefficient. However, the CCD problem in that study (and the one considered in this article) is an LQDO-amenable CCD problem but was not treated as such in Ref. [7], among other implementation improvements that would reduce computational expense.

Several studies have shown quantitative comparisons between a simultaneous and nested implementation of the same CCD problem. In Ref. [31], the two strategies converged to the same solution (as expected if all equivalency assumptions are met). The main computational complexity comparison was in the form of function evaluations of $\boldsymbol{f}$, which was a costly black-box function. The nested approach had about half of the number of function calls. However, without also comparing computation times, it may have been the case that the simultaneous approach had a lower CPU time because there is more involved with the two coordination strategies than function calls (although in this study, it may indeed have been the dominating factor).

In Ref. [77], the simultaneous approach produced a lower objective function value, even using a QP for the nested inner-loop problem. However, the implementation details and computation times are lacking, so it is challenging to generalize the outcomes. Similar statements can be made for Ref. [30] (and the simultaneous and nested approaches are much better than the other considered strategies). In Ref. [23], comparisons were made on a very simple CCD problem, and results showed that the nested approach was much better for large $n_t$. However, the simultaneous implementation was extremely inefficient, not considering sparsity or accurate derivative computations. In Ref. [22], a moderately-complex LQDO-amendable CCD problem was considered using both approaches. It was shown that there are several factors that impact which method is superior. However, the conclusions were relatively limited scope because the sensitivity to the convergence tolerances and other means for reducing runtime were not explored.

### 4.3.2 Potential Trade-offs

We now summarize the potential trade-offs between the coordination strategies as found in the CCD and MDO literature.

**Advantages (+) of the simultaneous strategy**

- Can potentially find the solution quickly by letting the optimizer explore regions that are infeasible [80].
- Naturally handles bidirectional coupling between the plant and control design variables [7, 23].
- Supports fine-grained parallelization [7].
- Better supports advanced derivative methods (e.g., complete problem analytic derivatives and the complex-step method).
- Only a single problem to construct and manage.

**Advantages (+) of the nested strategy**

- Each subproblem's structure is simplified and size reduced from the original simultaneous formulation (e.g., dynamics now considered with fixed $\boldsymbol{x}_p$).
- Tailored optimization algorithms (and tolerances) can be used in the different subproblems that can leverage the simplified subproblem structure (e.g., QP or LQR) or outer-loop global search [23, 34, 70, 81, 82].
- If the inner loop is always feasible, this approach naturally handles bidirectional coupling between the plant and control design variables [23].
- Results from intermediate iterations are feasible.
- In some cases, potentially fewer function calls for certain elements (e.g., if $\boldsymbol{f}(\cdot)$ is a linear dynamic system) [31, 34, 70].

**Disadvantages (−) of the simultaneous strategy**

- Large problem size [80].

- Requires many function calls to potentially expensive problem elements such as $\boldsymbol{f}(\cdot)$ [31, 34, 70].
- Results from intermediate iterations are not guaranteed to be feasible [80].

**Disadvantages ($-$) of the nested strategy**

- Issues when the inner-loop problem is not defined at a particular $\boldsymbol{x}_p$ [23].
- Challenging to compute accurate outer-loop derivatives (e.g., accuracy of the subproblem impacts accuracy of outer-loop derivative information [80], and analytical derivatives sometimes impossible to construct).
- Only supports coarse-grained parallelization [7].

### 4.3.3 Towards a Fair Comparison

From the discussion above and existing CCD literature, there does not seem to be a consensus or even good, purposeful guidelines on when to use either strategy. This shortcoming may be due to CCD problems and design goals diversity, but clear case studies could be quite insightful.

One of the main challenges in making a fair comparison is the fact that different optimization and analysis architectures are typically used when implementing the two coordination strategies on the same problem. Furthermore, if ineffective implementations of some parts of a coordination/solution method are used, then the comparison may be biased. For example, in Ref. [23], no fair comparisons were made because the simultaneous did not leverage the problem's sparsity. While complete optimization of the implementation strategy may not be useful in every case, a good case study should put forth the effort to compare multiple implementations. More universal metrics such as the number of function calls can be useful [11, 31], but often, the function calls are significantly different between the coordination strategies, including the use of different models and code vectorization, obscuring totals (and a similar case can be made for iterations).

In Ref. [23], runtime benchmarking was used to compare the strategies, but this approach may have issues with generalizability due to its dependence on computer architecture and the environment [84]. However, this approach does try to normalize the utility of the two strategies where

lowered measured runtime cost (for the same outcome) is desirable. Yet, solution quality is another dimension that could be argued. Generally, better solution quality implies lower objective function value but also can include closer satisfaction of the constraints and more alternative feasible solutions. Additionally, it can be argued that in (early-stage) CCD problems, the optimal plant design variables may only need to be determined to a few decimal places. In contrast, state variables need many more to model the system dynamics accurately and not have accumulated errors. Therefore, the commonplace solution quality vs. computational expense trade-off should be considered.

The results presented in Sec. 4.5 will be towards making better comparisons considering the many factors enumerated in this section.

## 4.4 Active Suspension Problem

Active vehicle suspension CCD problems have been used to show the efficacy of the CCD methodology and compare the trade-offs between different strategies [11, 26, 34, 77]. The considered quarter-car suspension system and components are illustrated in Fig. 3.6 [11, 77]. The system consists of two masses (sprung mass $m_{s/4}$ and unsprung mass $m_{us/4}$), and the suspension between them is consists of a force actuator $u(t)$, a linear spring $k_s(\boldsymbol{x}_p)$, and a linear damper $c_s(\boldsymbol{x}_p)$. The remainder of the system consists of a linear tire spring $k_t$ and damper $c_t$ and road input $z_0(t)$. A simplified version of this problem has been solved in the previous chapter in Sec. 3.5.4.

### 4.4.1 System Dynamics

There are four states in the system:

$$\boldsymbol{\xi}(t) = \begin{bmatrix} z_{us} - z_0 & \dot{z}_{us} & z_s - z_{us} & \dot{z}_s \end{bmatrix}^T \tag{4.4}$$

where $z_{us} - z_0$ is the displacement between $m_{us/4}$ and $z_0$, $\dot{z}_{us}$ is the velocity of the $m_{us/4}$, $z_s - z_{us}$ is the relative displacement between the masses, and $\dot{z}_s$ is the velocity of $m_{s/4}$. The initial conditions

for the states are assumed to be zero, i.e., $\boldsymbol{\xi}(0) = \mathbf{0}$. The differential equation is:

$$\dot{\boldsymbol{\xi}}(t) = \boldsymbol{A}(\boldsymbol{x}_p)\boldsymbol{\xi}(t) + \boldsymbol{B}u(t) + \boldsymbol{E}\dot{z}_0(t) \tag{4.5a}$$

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{-k_t(\boldsymbol{x}_p)}{m_{us/4}} & \frac{-[c_s(\boldsymbol{x}_p)+c_t]}{m_{us/4}} & \frac{k_s(\boldsymbol{x}_p)}{m_{us/4}} & \frac{c_s(\boldsymbol{x}_p)}{m_{us/4}} \\ 0 & -1 & 0 & 1 \\ 0 & \frac{c_s(\boldsymbol{x}_p)}{m_{s/4}} & \frac{-k_s(\boldsymbol{x}_p)}{m_{s/4}} & \frac{-c_s(\boldsymbol{x}_p)}{m_{s/4}} \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} 0 \\ \frac{-1}{m_{us/4}} \\ 0 \\ \frac{1}{m_{s/4}} \end{bmatrix}, \quad \boldsymbol{E} = \begin{bmatrix} -1 \\ \frac{c_t}{m_{us/4}} \\ 0 \\ 0 \end{bmatrix} \tag{4.5b}$$

### 4.4.2 System Objective

The performance measure for a design load case is a combination of quadratic penalties on handling $(z_{us} - z_0)$, passenger comfort $\ddot{z}_s$, and control effort $u$:

$$o = \int_{t_0}^{t_f} \left[ w_1 \xi_1^2 + w_2 [\dot{\xi}_4(t, \boldsymbol{\xi}, u, \boldsymbol{x}_p)]^2 + w_3 u^2 \right] dt \tag{4.6}$$

with $w_1 = 10^5$, $w_2 = 0.5$, and $w_3 = 10^{-5}$ from Ref. [7].

Here we consider two load cases: 1) a ramp input conditions (road grade at 25% speed of 10 m/s), 2) rough road profile using IRI737 data [7]. The two are combined using a weighted sum as follows:

$$\min_{\boldsymbol{\xi}, u, \boldsymbol{x}_p} \; 10^{-2} o(\boldsymbol{\xi}_{\text{ramp}}, u_{\text{ramp}}, \boldsymbol{x}_p) + o(\boldsymbol{\xi}_{\text{rough}}, u_{\text{rough}}, \boldsymbol{x}_p) \tag{4.7}$$

where $\boldsymbol{x}_p$ is naturally shared between the two cases.

### 4.4.3 Spring Design

The spring physical design variables are the wire diameter $d \in [0.005, 0.02]$ [m], helix diameter $D \in [0.05, 0.4]$ [m], pitch $p \in [0.02, 0.5]$ [m], and number of active coils $N_a \in [3, 16]$. The spring constant $k_s$ is computed by:

$$k_s(\boldsymbol{x}_p) = \frac{d^4 G}{8 D^3 N_a \left[1 + \frac{d^2}{2D^2}\right]} \tag{4.8}$$

where $G$ is the shear modulus. The quantity $L_0 = pN_a + 2d$ is the free length of the spring. $L_s = d(N_a + Q - 1)$ represents the solid height of the spring with $Q = 1.75$. $C = D/d$ is the spring index. The following constraints are included for manufacturability, antitangling, buckling,

minimum pocket length, and minimum pocket width:

$$g_{o,1}(\boldsymbol{x}_p) = 4 - C \leq 0 \tag{4.9}$$

$$g_{o,2}(\boldsymbol{x}_p) = C - 12 \leq 0 \tag{4.10}$$

$$g_{o,3}(\boldsymbol{x}_p) = L_0 - 5.26D \leq 0 \tag{4.11}$$

$$g_{o,4}(\boldsymbol{x}_p) = L_0 - 0.40 \leq 0 \tag{4.12}$$

$$g_{o,5}(\boldsymbol{x}_p) = d + D - 0.25 \leq 0 \tag{4.13}$$

The axial force is $F_s = k_s(L_0 - L_s)$, and the shear yield stress is related to the ultimate shear stress by $S_{sy} = 0.65S_{ut}$, where $S_{ut} = 1974d^{-0.108} \times 10^6$. The shear stress is:

$$\tau(F) = \left(\frac{4C+2}{4C-3}\right)\frac{8FD}{\pi d^3} \tag{4.14}$$

Now, we enforce that the maximum shear stress must not be higher than $S_{sy}$:

$$g_{o,6}(\boldsymbol{x}_p) = 1.2\tau(F_s) - S_{sy} \leq 0 \tag{4.15}$$

There are also constraints that depend on the dynamic nature of the states. A rattlespace constraint depends on the maximum displacement of the spring:

$$g_{i,1}(\boldsymbol{x}_p, \boldsymbol{\xi}) = \max_t |\xi_3(t)| - L_0 + L_s + 0.02 + \delta_g \leq 0 \tag{4.16}$$

where $\delta_g = m_{s/4}g/k_s$ is the static suspension deflection with $g = 9.81$ m/s$^2$. To ensure the spring linearity assumption, we include:

$$g_{i,2}(\boldsymbol{x}_p, \boldsymbol{\xi}) = 0.15 + 1 - \frac{L_0 - L_s}{\delta_g + 1.1\xi_3(t)} \leq 0 \tag{4.17}$$

The maximum and minimum axial forces are $F_{\max} = k_s(\max_t|\xi_3(t)| + \delta_g)$ and $F_{\min} = k_s(\delta_g - \max_t|\xi_3(t)|)$, respectively. Then, the mean axial force and force amplitude are defined by $F_m = (F_{\max}+F_{\min})/2$ and $F_a = (F_{\max}-F_{\min})/2$, respectively. Soderberg fatigue criterion and Zimmerli limit are considered with:

$$g_{i,3}(\boldsymbol{x}_p, \boldsymbol{\xi}) = \frac{1.2\tau(F_a)}{0.24S_{ut}} + \frac{\tau(F_m)}{S_{sy}} - 1 \leq 0 \tag{4.18}$$

$$g_{i,4}(\boldsymbol{x}_p, \boldsymbol{\xi}) = \frac{1.2\tau(F_a)}{241 \times 10^6} - 1 \leq 0 \tag{4.19}$$

### 4.4.4 Damper Design

There are three parameters for damper design including the valve diameter $D_o \in [0.003, 0.012]$ [m], working piston diameter $D_p \in [0.03, 0.08]$ [m], and damper stroke $D_s \in [0.1, 0.3]$ [m]. The damper constant $c_s$ is:

$$c_s(\boldsymbol{x}_p) = \frac{D_p^4}{8C_dC_2(D_o)D_o^2}\sqrt{\frac{\pi k_v \rho_1}{2}} \tag{4.20}$$

where $C_d$, $C_2(D_o)$, $\rho_1$, and $k_v$ are parameters in Ref. [7]. The damper must fit inside the spring, thus constraints on its size and range of motion are included:

$$g_{o,7}(\boldsymbol{x}_p) = d - D + D_p + 0.022 \leq 0 \tag{4.21}$$

$$g_{o,8}(\boldsymbol{x}_p) = 2D_s - 0.394 \leq 0 \tag{4.22}$$

$$g_{o,9}(\boldsymbol{x}_p) = L_0 - L_s - D_s \leq 0 \tag{4.23}$$

The dynamic constraints depend on the suspension's relative velocity $\dot{\xi}_3 = \xi_4 - \xi_2$ and are meant to limit the heat generated by the working fluid. These constraints restrict the maximum pressure on the damper fluid, the maximum allowable velocity, and the spool valve lift, respectively:

$$g_{i,5}(\boldsymbol{x}_p, \boldsymbol{\xi}) = \frac{4c_s(D_o)\max_t|\dot{\xi}_3(t)|}{\pi D_p^2} - 4.75 \times 10^6 \leq 0 \tag{4.24}$$

$$g_{i,6}(\boldsymbol{x}_p, \boldsymbol{\xi}) = \max_t|\dot{\xi}_3(t)| - 5 \leq 0 \tag{4.25}$$

$$g_{i,7}(\boldsymbol{x}_p, \boldsymbol{\xi}) = \frac{4\pi D_o^2 c_s(D_o)\max_t|\dot{\xi}_3(t)|}{4k_v\pi D_p^2} - 0.03 \leq 0 \tag{4.26}$$

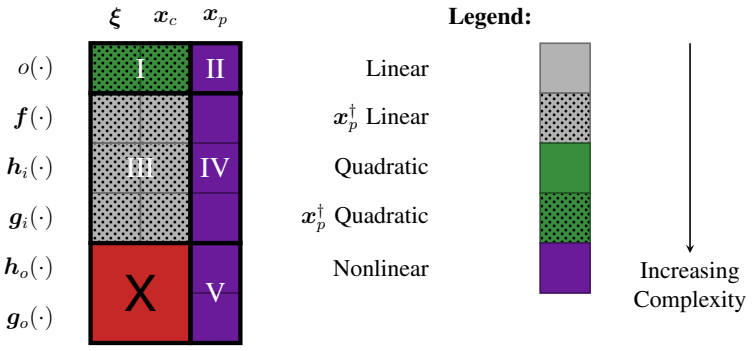



**Figure 4.2:** Allowable dependency matrix form for an LQDO-amendable CCD problem.

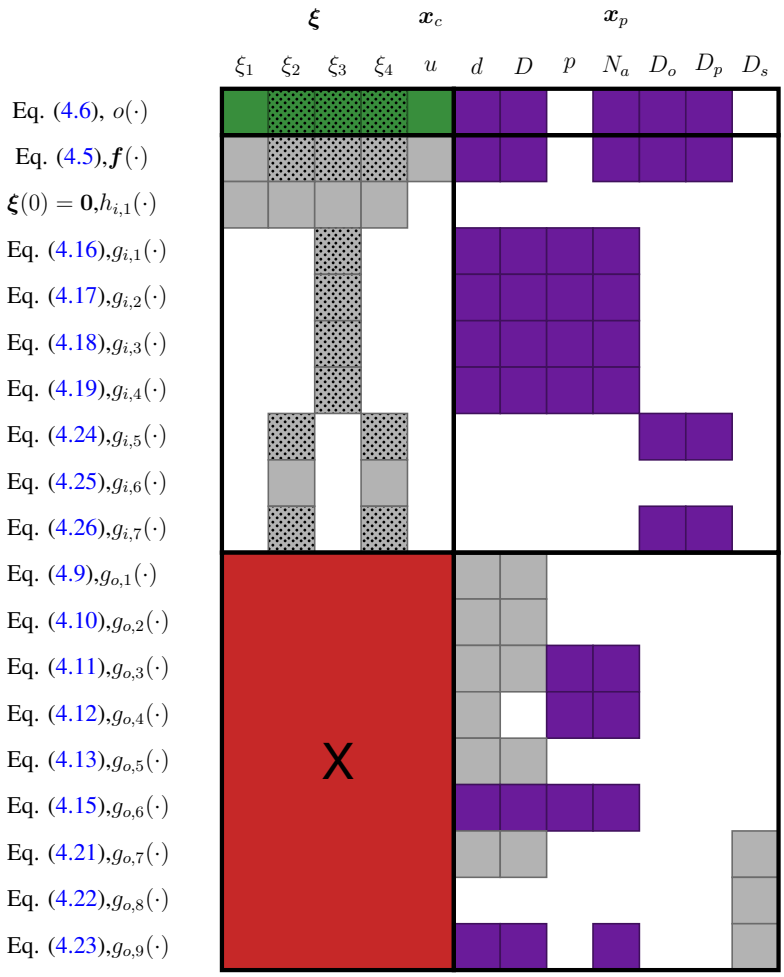



**Figure 4.3:** Dependency matrix for suspension CCD problem.

### 4.4.5 Analysis of the CCD Problem

With the entire problem defined, it can be helpful to analyze the CCD problem structure. To this end, a dependency matrix visualization template is provided in Fig. 4.2 where rows are problem elements from Prob. (4.1), and the columns are for design variable dependence. A similar analysis was performed in Refs. [70] without the detail provided here. Additionally, this is similar to the Jacobian matrix shown in Ref. [7] with a few key differences. First, the dependencies are categorized as linear, quadratic, and nonlinear, and a specialized qualifier is added for the case when the plant design is fixed (denoted $\boldsymbol{x}_p^\dagger$). For example, the optimization variable dependence for $\boldsymbol{f}(\cdot)$ in Eq. (4.5) is only linear if the plant design is fixed. By filling in all the appropriate entries, one can determine if their CCD problem is an LQDO-amendable CCD problem by merely

verifying the complexity is lower than the level shown in Fig. 4.2. It also facilitates the partitioning of the inner- and outer-loop constraints. Because of the definitionally disallowed X region, any constraints that do not only depend on $\boldsymbol{x}_p$ must be in the inner-loop.

For this case study, the dependency matrix is shown in Fig. 4.3. All $\boldsymbol{g}_i(\cdot)$ can be transformed into a linear state form, potentially needing two constraints. Comparing with Fig. 4.2, this is an LQDO-amendable CCD problem. There are additional uses for this representation. Regions II and IV provide insights into when the inner-loop problem needs to be solved. Here $D_s$ doesn't change the inner-loop problem. Furthermore, dominated inner-loop constraints are observed in Eqs. (4.16)–(4.19) because they have the same row properties with only a single linear state variable. Therefore, only one can be active at a time, and this constraint can be easily determined. The same can be said for Eqs. (4.24)–(4.26), again reducing three path constraints to only one. Finally, it helps determine in Region V which plant constraints are linear (if the entire row is linear).

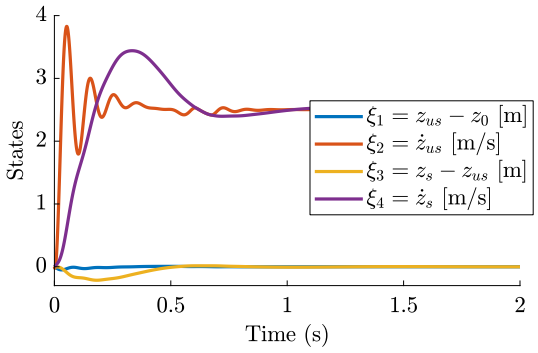
This discussion highlights the potential advantage of the nested strategy to structure and simplify the subproblem. Effective nested and simultaneous implementations should take advantage of all of this information.

## 4.5 Results

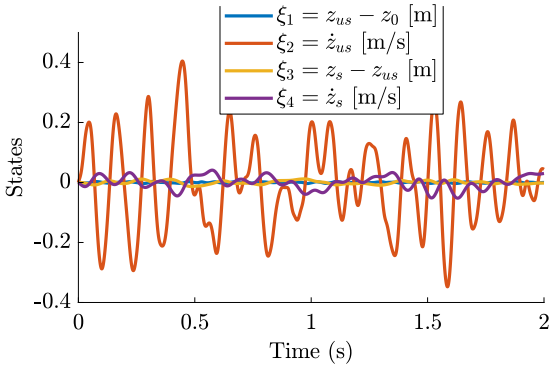
In this section, we solve the LQDO-amenable CCD problem defined in Sec. 4.4 using a variety of coordination strategies, computational methods, and tunable parameters. The case study is available at and solved using the free and open-source `DTQP` software tool (based on commit `33c2f24`) [54]. The computer architecture used for the results was a desktop workstation with an AMD 3970X CPU at 3.7 GHz, 128 GB 3200 MHz RAM, `Matlab` 2020a update 5, and `Windows` 10 build 17763.1432.

### 4.5.1 Complete Solution

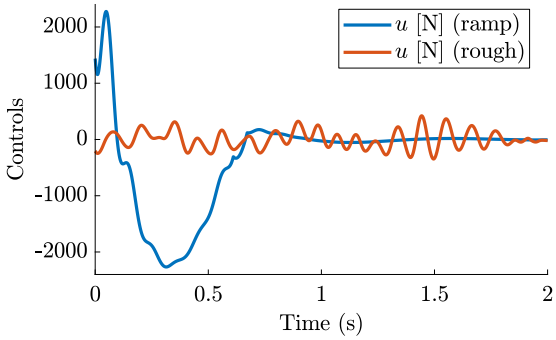
Before we begin comparing different instances of the CCD solution strategies, we first want to present a single complete and accurate solution to the CCD problem. The optimal trajectories




**(a)** Optimal states for ramp case.



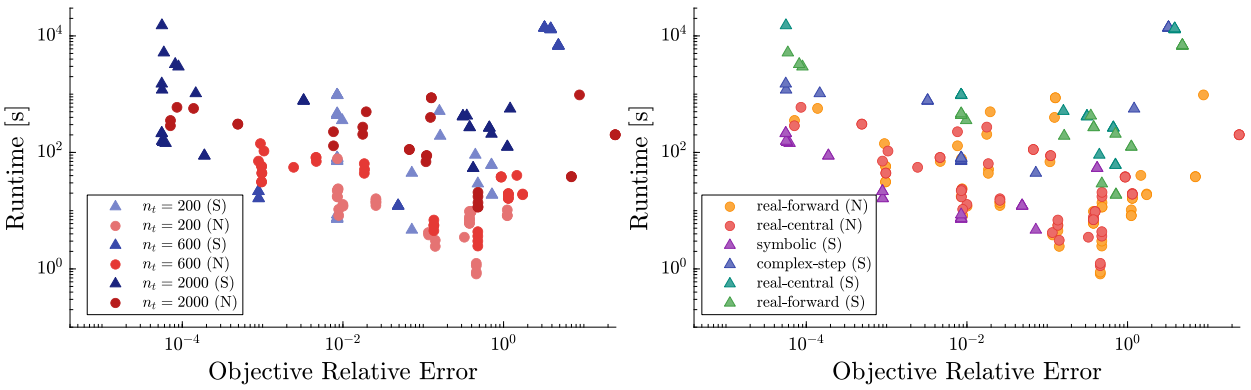

**(b)** Optimal states for rough road case.




**(c)** Optimal controls.

**Figure 4.4:** Optimal trajectories with $n_t = 5000$.

for select states and controls under the ramp and rough road load cases are shown in Fig. 4.4. The optimal objective function value $o^*$ was $2.0677$, and the ramp and rough road constituents were $56.340$ and $1.504$, respectively. These performance values were obtain using both the simultaneous ($\mathsf{S}$) and nested ($\mathsf{N}$) CCD strategies using $n_t = 5000$ and small tolerances to ensure high quality solutions. Interestingly, the plant designs from the two implementations were slightly different as noted in Fig. 4.4. However, the key intermediate variable values found using $\mathsf{S}$ of $k_s^* = 2.366 \times 10^4$ N/m and $c_s^* = 839.8$ Ns/m were within $2\%$ of $\mathsf{N}$ so the optimal state and control trajectories were nearly indistinguishable. In Ref. [7], a solution was demonstrated with $o^* = 2.12$ using a much coarser mesh, but the overall the values of $k_s^*$ and $c_s^*$ and the optimal trajectories were generally similar.




**(a)** Different $n_t$.

**(b)** Different derivative methods.

**Figure 4.5:** Run time vs. relative objective function error for various solution implementations.

### 4.5.2 Solution Implementation Variations

There are various key decisions to be made when attempting to find a solution to a CCD problem numerically. In this section, we will explore several implementation variations to gain insights into best practices.

The *S* architecture used `fmincon` with the built-in interior point algorithm where all derivatives are provided utilizing the sparsity pattern for DT problems. The variations tested were composed of permutations of the following:

- Three different values of $n_t = [200, 600, 2000]$
- Three optimality tolerance values $[10^{-3}, 10^{-5}, 10^{-7}]$
- Three feasibility tolerance values $[10^{-4}, 10^{-8}, 10^{-12}]$
- Four derivative methods (symbolic, complex-step differentiation, real-central finite difference (FD), and real-forward FD)

The *N* architecture used `fmincon` with the built-in interior point algorithm for the outer loop and `quadprog` for the inner-loop QP. Parallel computing was utilized in the outer loop for FD. The variations tested were composed of permutations of the following:

- Three different values of $n_t = [200, 600, 2000]$
- Three outer-loop optimality tolerances values $[10^{-1}, 10^{-3}, 10^{-5}]$
- Three inner-loop optimality and feasibility tolerance values $[10^{-4}, 10^{-8}, 10^{-12}]$

- Two outer-loop derivative methods (real-central and real-forward FD)
- If a hybrid approach was used where a genetic algorithm was first run for one iteration (for global search and to find a point with a feasible inner loop). Otherwise a starting point with a feasible inner loop was used.

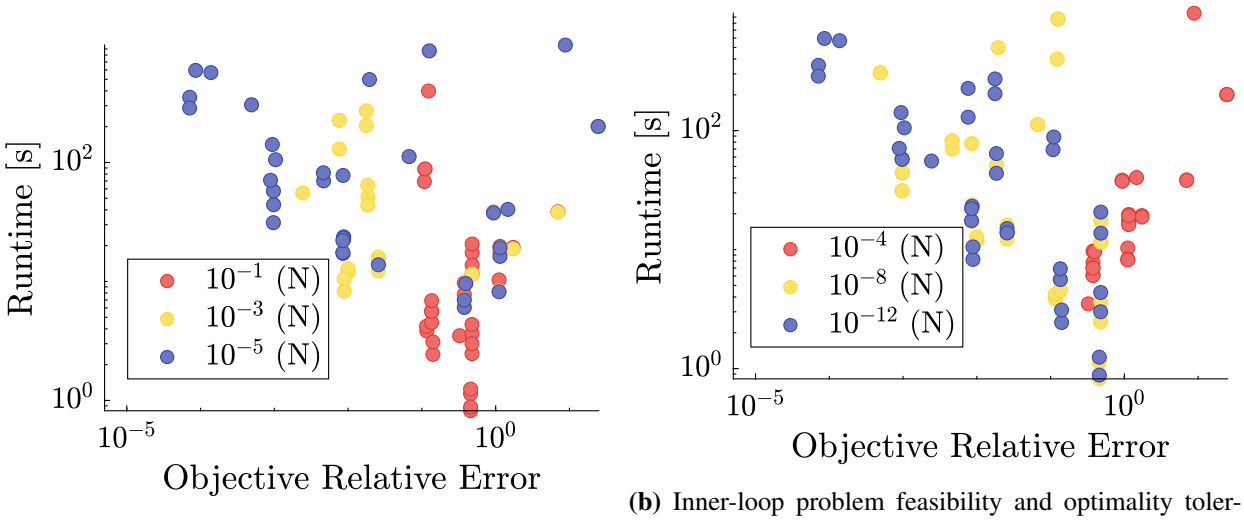
**Simultaneous vs. Nested**

All the variations are shown in Fig. 4.5 comparing the relative error of the objective value found above and the optimization runtime on the prescribed computer architecture. In Fig. 4.5a, the results indicate $S$ is the superior approach when using enough points for an accurate time discretization, i.e., $n_t \geq 600$.

However, it is important to look at the specific implementation that achieved this result. Figure 4.5b labels the points by the derivative method used. Now we can see why $S$ was superior; it was using symbolic derivatives! Accurate derivative information is shown to be a critical factor for efficient $S$ implementations. The implementations using the complex-step method, which is nearly as accurate as symbolic derivatives (second-order derivatives are less accurate), was on average $10\times$ slower. Therefore, these results indicate that $N$ is superior when symbolic derivatives are unavailable and potentially by an order of magnitude or more in runtime.

**Additional Simultaneous Insights**

The real-central and real-forward FD methods were about 5 to $10\times$ slower than the complex-step implementations. This result also explains why the reported runtimes in Ref. [11] where so much higher than the values reported here, which used the low-order derivative method with a large step size. Not shown in the figure, several instances of $S$ using the low-order derivative methods did not converge within a large iteration limit. Many models do not readily support symbolic derivatives or complex numbers, so one would need to use another derivative strategy shown to be quite inefficient, even when requesting low accuracy solutions (recalling that many variations on the tolerances were tested). Solutions with $O(10^{-2}, 10^{-3}, 10^{-4})$ accuracy were obtained in (7, 16, 144) s, respectively.




**(a)** Outer-loop problem optimality tolerance.

**(b)** Inner-loop problem feasibility and optimality tolerance.

**Figure 4.6:** Tolerance study results for the nested strategy.

### Additional Nested Insights

Figure 4.6 shows only the $N$ implementations highlighting the different inner and outer loop tolerances. In Fig. 4.6a, the outer-loop tolerance shows clear trade-off between accuracy and runtime. Loss in outer-loop accuracy is commonplace when the inner-loop tolerance is not small enough (cf. Fig. 4.6b). Balancing these tolerances can permit efficient CCD solution implementations that achieve the desired accuracy with extra computational cost. Solutions with tolerances set for $O(10^{-2}, 10^{-3}, 10^{-4})$ accuracy were obtained in (8, 31, 287) s, respectively.

A fundamental assumption for equivalence between the nested and simultaneous formulations is consistent inner-loop feasibility. However, uniform random sampling of plant designs within the simple upper and lower bounds indicate 44% have infeasible inner loops, and approximately 0.033% are infeasible with respect to all constraints! The starting point used in Ref. [7] is one of those infeasible points, so a simple nested implementation would not converge. In this case, infeasible inner loops are when some of the additional inequalities are not well-posed. In fact, the inner-loop QP is immediately declared infeasible by `quadprog`.

A possible deterrent against this issue is a hybrid optimization scheme. In this study, the hybrid scheme consisted of a genetic algorithm and then a derivative-based optimizer. Additionally, im-

plementing the five linear $\boldsymbol{g}_o(\cdot)$ constraints appropriately (so not as nonlinear constraints) was quite helpful in finding better populations and search directions because many optimization algorithms can directly avoid linear constraint infeasibility regions. Also, including appropriate outer-loop feasibility constraints, as suggested in Ref. [23], could help guide the outer-loop towards a feasible inner-loop region and would take the form of positive bound coefficients in $g_{i,1}(\cdot)$ through $g_{i,4}(\cdot)$ in this case.

These additions help in curbing infeasible results and help solve the problem faster as well. If a candidate is infeasible for one scenario, then the inner-loop terminates and the outer-loop solver is forced to find a better candidate.

---

This chapter provides a fair comparison between the nested and simultaneous CCD of an active vehicle suspension design problem. As presented above, the simultaneous strategy using symbolic derivatives was generally superior, while the nested strategy was preferable when any other derivative method was used. However, special care is needed to handle potentially infeasible inner loops in the nested method. In the next chapter, we will investigate the use of nested CCD on a considerably larger FOWT design problem.

# Chapter 5

# Control Co-Design of a Floating Offshore Wind Turbine

This chapter applies the heuristics developed from the previous studies on efficient CCD implementations and considerations to CCD of a floating offshore wind turbine[5].

## 5.1 Introduction

As mentioned previously, the design of floating offshore wind turbines (FOWT) has followed a sequential pattern, where the physical plant parameters are designed first, and a controller is then optimized for this particular plant [12, 13, 36, 86]. However, in FOWTs, there are strong interactions between the dynamic response of the system to environmental excitations and the controller. Unfortunately, a sequential design process can produce unstable systems as it does not account for this coupling [35, 87]. As a consequence of this, the turbine will not be able to generate enough power to make it economically feasible. Optimizing both the physical plant and the controller simultaneously enables identification of stable, system-level optimal results. This integrated design approach has been studied extensively under the term control co-design (CCD) [8,12,26,29,88,89]. Recently the importance of these integrated design approaches for energy system design has been recognized by domain experts. References [3, 35, 90] have explored the application of integrated design to offshore wind turbines, and Refs. [14, 91] discuss the application of CCD to the design of wave energy converters. Integrated design approaches have also found applications in design of mixed renewable/nonrenewable power generation systems [92].

The primary design goal of any wind-based energy system is to capture as much power from the incoming wind while minimizing the structure's dynamic loads. In the case of FOWT systems, the system dynamic loads and its power production are heavily dependent on both the physical

[5]This chapter is based on the following publication [85].

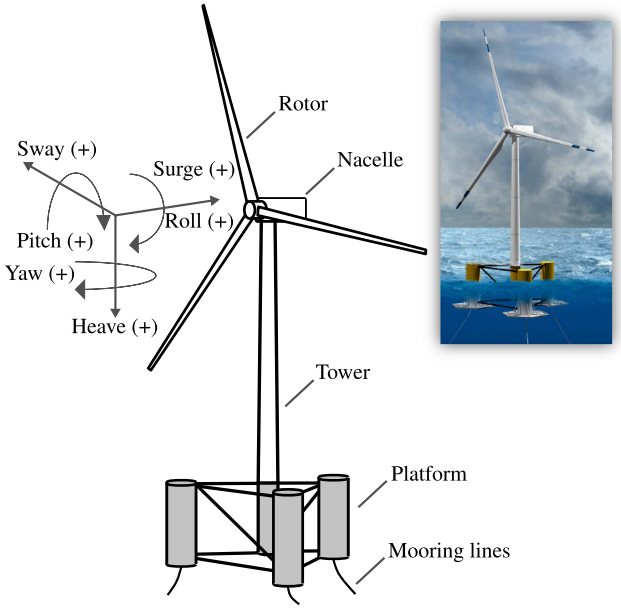



**Figure 5.1:** Floating offshore wind turbine (illustration courtesy of NREL).

plant specification and the control strategy used. In order to design a system, the objective must capture to capture the economic impact of the plant and control decisions, along with physical response. For energy production systems, these goals are captured by the levelized cost of energy (LCOE) [93]:

$$\text{LCOE} = \frac{\text{Total Lifetime Cost}}{\text{Total Lifetime Energy Output}} \tag{5.1}$$

The total lifetime costs of the FOWT system are a combination of the initial cost needed to build the system (capital cost) and the maintenance costs over its lifetime. The capital costs are often directly linked to the plant design decisions [1, 94]. The maintenance costs and the total lifetime energy output are dependent on how the system operates and, consequently, depend on the environment and how it is controlled [95]. Recent studies have shown that advanced control strategies for offshore wind applications can increase the power extracted from the turbine, and minimize the levelized cost [96]. Traditional LCOE estimates have not used such novel control methods, nor have they considered their physical and economic impact. In the case of highly coupled systems like FOWTs, such considerations are imperative, considering the amount of effort that goes into

making these systems economically viable [2]. Additionally, overlooking the impact of control decisions on the physical design is a pitfall of sequential design approaches.

## 5.2 Design of FOWT

### 5.2.1 Plant Design of Floating Offshore Wind Turbines

The plant design of a FOWT involves optimizing the physical response of the system in order to reduce fatigue and dynamic loads while keeping the cost of the system competitive. The primary elements of a FOWT are the rotor, drive train, nacelle, tower, and support structure, and are labeled in Fig. 5.1.

Stability of the FOWT about its natural equilibrium is required in all manner of wind, wave, and current excitations that the system might experience [97]. Reference [98] provides information about the standard industry requirements of a FOWT. Generally, increasing the mass of the support structure will make the FOWT more stable, but this would also raise the capital and other costs. Therefore, it is essential to optimize the system for cost while ensuring stability [99]. As the development cycle progresses, additional practical considerations like assembly costs and procedures, maintenance costs, and ease of transportation may also be incorporated into the plant design.

### 5.2.2 Wind Turbine Control of Floating Offshore Wind Turbines

The control system for a FOWT is instrumental in achieving the design goals stated in the previous sections. The power generated by a FOWT and the physical loads on its structure is heavily dependent on the loading conditions induced by the wind, waves, and currents. Operating the system in such a way so that it can remain stable while producing maximal power is the primary goal of the FOWT control system. Similar to the control of land-based wind turbines, the control strategy selected depends heavily on the system's input excitations, as these inputs produce the dynamical responses we seek to optimize.

The primary mode of control for any wind turbine depends heavily on the wind, so specific operating regions are often defined based on the wind speed [100, 101]. Typically, there are four

wind speed-based regions of interest, visualized in Fig. 5.2. At the lower, below-rated wind speeds, the system produces limited power. Above the rated wind speed, the turbine is designed to operate at its maximum power level. In between these regions, there is a transition behavior, and at extremely fast, above-rated wind speeds, the system is shut down as there can be permanent structural damage.

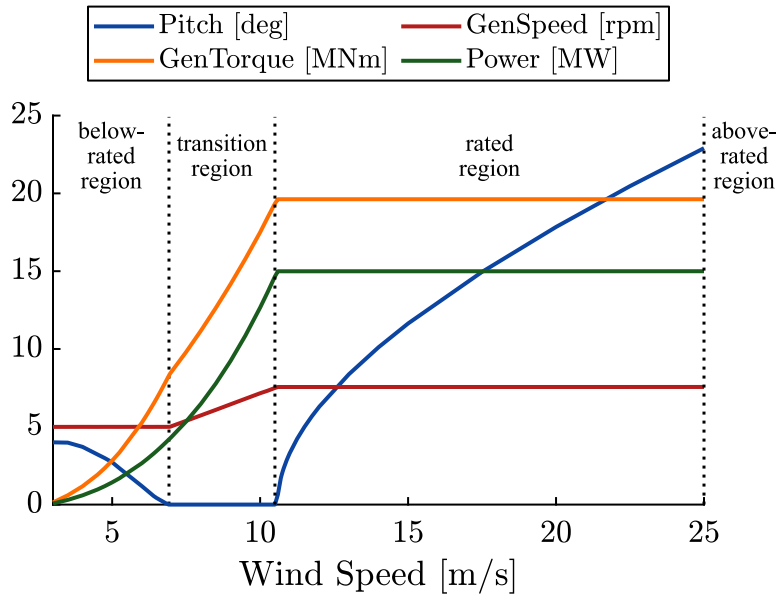



**Figure 5.2:** Controller regulation trajectory from Ref. [3].

The two primary control inputs for wind turbines are the pitch angle of the turbine blades (commonly called blade pitch) and the torque produced by the generator. In below-rated wind speeds, varying the generator torque is the primary mode of control of the turbine [3]. At rated wind speeds, the generator torque is held constant, and the blade pitch is varied in a process called maximum power point tracking where the rotor's angular velocity is continuously adjusted to extract the maximum possible power from the incoming wind.

### 5.2.3 Modeling Considerations

It is often necessary to conduct early-stage design studies to understand the desired fundamental system properties and behaviors that inform critical decisions that need to be made as the

system-of-interest is realized. The use of high-fidelity modeling tools in early-stage design studies is not always needed to achieve the desired design insights and can be prohibitive due to their complexity and computational expense.

To facilitate these design and control (both closed- and open-loop) studies, it is common to develop reduced or lower-order models that capture the system's essential physics. Results from these reduced-order models are validated against the simulations from high-fidelity tools to understand their veracity in studying the system's behavior. After validation, these models are then linearized around predetermined set-point values in distinct operating regions. These linearized models are then used to understand the system dynamics and design controllers in these operating regions.

However, there are some drawbacks in developing these lower-order models. The development of these lower-order models is complicated as they require extensive subject knowledge of FOWTs and the associated physics/engineering disciplines. Additionally, the lower-order models are developed to study a specific aspect of the system's behavior, e.g., the floating structure response, controller response, aerodynamic wake, etc. As such, the results from these models cannot be easily generalized. The highly-coupled nature of a FOWT can create further complications in modeling the system accurately [35, 102–105].

One way to mitigate these difficulties is by using linearized models obtained directly from high-fidelity (e.g., computational fluid dynamics) modeling tools [106, 107]. These models are obtained by linearizing the nonlinear system around specific operating points, often stationary points where the system exhibits steady state dynamics. A linear time-invariant state-space dynamic model about the static operating point $(\boldsymbol{\xi}_o, \boldsymbol{u}_o)$ typically has the following form:

$$\frac{d\boldsymbol{\xi}_\Delta(t)}{dt} = \boldsymbol{A}_o\boldsymbol{\xi}_\Delta(t) + \boldsymbol{B}_o\boldsymbol{u}_\Delta(t) \tag{5.2a}$$

$$\boldsymbol{y}(t) = \boldsymbol{C}_o\boldsymbol{\xi}_\Delta(t) + \boldsymbol{D}_o\boldsymbol{u}_\Delta(t) + \boldsymbol{g}_o \tag{5.2b}$$

where $t$ is time, $\boldsymbol{\xi}_\Delta(t)$ are the relative states related to the original states $\boldsymbol{\xi}$ with $\boldsymbol{\xi}(t) = \boldsymbol{\xi}_\Delta(t) + \boldsymbol{\xi}_o$, $\boldsymbol{u}_\Delta(t)$ are the relative inputs related to the original inputs $\boldsymbol{u}$ with $\boldsymbol{u}(t) = \boldsymbol{u}_\Delta(t) + \boldsymbol{u}_o$, $\boldsymbol{y}(t)$ are the outputs, and the matrices $(\boldsymbol{A}_o, \boldsymbol{B}_o, \boldsymbol{C}_o, \boldsymbol{D}_o, \boldsymbol{g}_o)$ are associated with the linearization process.

A significant drawback with any kind of linearized model is their accuracy in capturing the system's dynamic response diminishes quickly as the system's behavior moves away from the initial operating point. Thus, it becomes difficult to work with many diverse design load cases, where the wind speed continuously changes. Some studies that have used linearized models leveraged them in gain scheduling approaches to account for nonlinearities. However, this approach does not guarantee stability and performance for all possible values of the wind speed [108].

In this chapter, we will discuss the use of linear parameter-varying (LPV) models to help overcome the drawbacks of distinct linear models [108, 109]. These LPV models show good accuracy when capturing the original nonlinear dynamics, and can be used to generate open-loop optimal control trajectories [110]. Additionally, they have been used to develop closed-loop controllers for wind turbines [108, 110].

### 5.2.4 Integrated Design with Control Co-Design

The CCD approach provides a rigorous framework that can naturally handle the coupling between the plant and control drivers present in FOWTs.

There are certain advantages for nested CCD approach, as discussed in the previous chapters, for problems where the inner loop is a linear-quadratic dynamic optimization (LQDO) problem. The use of open-loop control during early-stage design studies can show the maximum achievable performance of the system and provide the desired insights into the optimal system dynamics and controller behavior [31, 78, 91]. Additionally, nested CCD is often necessary when black-box models of the dynamics are used (as will be the case in this work) [29, 32].

### 5.2.5 Use of OpenFAST and WEIS Models

The wind energy with integrated servo-control (WEIS) is an open-source project that is currently being developed by the National Renewable Energy Laboratory (NREL) and partners, that will allow users to perform CCD of FOWT systems [6, 111]. The WEIS toolbox is built on OpenFAST, another open-source toolbox developed by NREL, that generates a full-system dynamic response of FOWTs under wind, wave, and current excitations. The OpenFAST tool is built on

independent modules that capture the important physical phenomena of the different FOWT subsystems and couplings between them. There are different modules to capture the aerodynamic, hydrodynamic, servodynamic, and mooring dynamics response of the system to these excitations. A variety of plant design decisions can be explored within these tools as well [6].

In this chapter, the dynamic models of FOWTs will be generated using the linearization capabilities of the WEIS/OpenFAST tools, and the original nonlinear dynamics simulation capabilities being used for validation of the results. A detailed discussion regarding the linearization capabilities of OpenFAST and the entire tool can be found in Refs. [106, 107, 111, 112]. Wind speed is used to select the state and control operating points for this linearized model.

## 5.3 Linear Parameter-Varying Models

As mentioned previously in Sec. 5.2.3, linearized models, like the one defined in Eq. (5.2), can accurately describe the system's behavior for small perturbations about the operating point from which they were derived. For the design and optimization activities of a FOWT system, it is essential to understand the system behavior over multiple input excitations, and consequently over multiple operating points. While there are additional drivers for modeling variations, the primary one in wind energy systems, including FOWTs, is the wind speed in the direction of the turbine-blade system. Under different wind conditions, the stationary operation points for the FOWT system greatly varies as do the matrices defining the dynamic model in Eq. (5.2). Therefore, here we will consider models that are dependent on this important parameter andcan be useful in generating open-loop optimal control trajectories for the CCD approach.

### 5.3.1 Linear Parameter-Varying Model Derivation

LPV models are a special case of linear time-varying (LTV) systems where the system matrices are continuous and are a function of a set of parameters [109]. Here we will consider the single parameter case where the parameter is denoted by $w$, and indicates the current wind speed value.

Now, consider the following nonlinear parameter-dependent model:

$$\Sigma = \begin{cases} \dfrac{d\boldsymbol{\xi}}{dt} = \boldsymbol{f}(\boldsymbol{\xi}, \boldsymbol{u}, w) \\ \boldsymbol{y} = \boldsymbol{g}(\boldsymbol{\xi}, \boldsymbol{u}, w) \end{cases} \tag{5.3}$$

Our goal is to linearize this model about the $w$-varying operating point functions $(\boldsymbol{\xi}_o(w), \boldsymbol{u}_o(w))$ where stationary or steady-state models characterize their values:

$$\boldsymbol{f}(\boldsymbol{\xi}_o(w), \boldsymbol{u}_o(w), w) = \boldsymbol{0}, \ \forall w \in [w_{\min}, w_{\max}] \tag{5.4}$$

where $w_{\min}$ is the minimum parameter value considered and $w_{\max}$ the maximum.

Now the relationship between the linearization states and the original states depends on the parameter $w$:

$$\boldsymbol{\xi}(t) = \boldsymbol{\xi}_\Delta(t) + \boldsymbol{\xi}_o(w), \quad \boldsymbol{u}(t) = \boldsymbol{u}_\Delta(t) + \boldsymbol{u}_o(w) \tag{5.5}$$

Assuming that $w$ is time varying, the time derivative relationship of the states is:

$$\frac{d\boldsymbol{\xi}}{dt} = \frac{d\boldsymbol{\xi}_\Delta}{dt} + \frac{d}{dt}\boldsymbol{\xi}_o(w(t)) \tag{5.6a}$$

$$= \frac{d\boldsymbol{\xi}_\Delta}{dt} + \frac{\partial \boldsymbol{\xi}_o}{\partial w}\frac{dw}{dt} \tag{5.6b}$$

Now we use the following notation for the derivatives of the nonlinear model:

$$\boldsymbol{A}(w) \coloneqq J^{\boldsymbol{f}}_{\boldsymbol{\xi}}\left(\boldsymbol{\xi}_o(w), \boldsymbol{u}_o(w), w\right), \ \boldsymbol{B}(w) \coloneqq J^{\boldsymbol{f}}_{\boldsymbol{u}}\left(\boldsymbol{\xi}_o(w), \boldsymbol{u}_o(w), w\right)$$

$$\boldsymbol{C}(w) \coloneqq J^{\boldsymbol{g}}_{\boldsymbol{\xi}}\left(\boldsymbol{\xi}_o(w), \boldsymbol{u}_o(w), w\right), \ \boldsymbol{D}(w) \coloneqq J^{\boldsymbol{g}}_{\boldsymbol{u}}\left(\boldsymbol{\xi}_o(w), \boldsymbol{u}_o(w), w\right)$$

where $\boldsymbol{J}^{\boldsymbol{f}}_{\boldsymbol{x}}$ denotes the Jacobian of $\boldsymbol{f}$ with respect to $\boldsymbol{x}$, and the values of these functions are dependent on the operating points and are denoted as:

$$\boldsymbol{f}(w) \coloneqq \boldsymbol{f}\left(\boldsymbol{\xi}_o(w), \boldsymbol{u}_o(w), w\right), \ \boldsymbol{g}(w) \coloneqq \boldsymbol{g}\left(\boldsymbol{\xi}_o(w), \boldsymbol{u}_o(w), w\right)$$

With this derivative relationship in Eq. (5.6) and the notation above, the nonlinear system $\Sigma$ in Eq. (5.3) is linearized about $(\boldsymbol{\xi}_o(w), \boldsymbol{u}_o(w))$ yielding the following LPV system:

$$\Sigma_w = \begin{cases} \dfrac{d\boldsymbol{\xi}_\Delta}{dt} = \cancelto{0}{\boldsymbol{f}(w)} + \boldsymbol{A}(w)\boldsymbol{\xi}_\Delta + \boldsymbol{B}(w)\boldsymbol{u}_\Delta - \dfrac{\partial \boldsymbol{\xi}_o(w)}{\partial w}\dfrac{dw}{dt} \\ \boldsymbol{y} = \boldsymbol{g}(w) + \boldsymbol{C}(w)\boldsymbol{\xi}_\Delta + \boldsymbol{D}(w)\boldsymbol{u}_\Delta \end{cases} \tag{5.8}$$

Note, if only a single time-invariant value of the parameter denoted $w_o$ is considered, then we have the following system:

$$\frac{d\boldsymbol{\xi}_\Delta}{dt} = \boldsymbol{A}(w_o)\boldsymbol{\xi}_\Delta + \boldsymbol{B}(w_o)\boldsymbol{u}_\Delta - \frac{\partial \boldsymbol{\xi}_o(w_o)}{\partial w}\cancelto{0}{\frac{dw}{dt}} \tag{5.9a}$$

$$\boldsymbol{y} = \boldsymbol{g}(w_o) + \boldsymbol{C}(w_o)\boldsymbol{\xi}_\Delta + \boldsymbol{D}(w_o)\boldsymbol{u}_\Delta \tag{5.9b}$$

which gives us:

$$\Sigma_o = \begin{cases} \dfrac{d\boldsymbol{\xi}_\Delta}{dt} = \boldsymbol{A}(w_o)\boldsymbol{\xi}_\Delta + \boldsymbol{B}(w_o)\boldsymbol{u}_\Delta \\ \quad \boldsymbol{y} = \boldsymbol{g}(w_o) + \boldsymbol{C}(w_o)\boldsymbol{\xi}_\Delta + \boldsymbol{D}(w_o)\boldsymbol{u}_\Delta \end{cases} \tag{5.10}$$

which is the same LTI system defined in Eq. (5.2) for a single operating point characterized by the parameter $w_o$.

### 5.3.2 Construction using Multiple Linearized Models

The system $\Sigma_w$ with continuous dependence on the parameter $w$ generally will not be directly available because linearized models are often realized through numeric methods for specific operating points (i.e., $\Sigma_o$). Therefore, it may be necessary to construct $\Sigma_w$ from a strategic finite set of $\Sigma_o$ models. To accomplish this goal, the matrix entries of $\Sigma_w$ are determined by element-wise matrix interpolation from a set of given denoted $\boldsymbol{\Omega} = \{\Sigma_{o1}, \Sigma_{o2}, \cdots, \Sigma_{on}\}$ each created using the parameters values $\boldsymbol{W} = [w_1, w_2, \cdots, w_n]$. Derivatives of the polynomial interpolating function as computed directly when needed.

Now there are several properties to consider to ensure such an interpolation scheme has reasonable chance of meaningfully capturing the nonlinear dynamics including:

(P1) The states, inputs, and outputs are unchanging for all considered $\Sigma_o$.

(P2) The sparsity patterns (nonzero entries in the system matrices) are generally similar between analogous matrices.

(P3) The stationary condition in Eq. (5.4) holds for the given interpolation scheme and $\boldsymbol{W}$, i.e., $(\boldsymbol{\xi}_o(w), \boldsymbol{u}_o(w))$ can be found through interpolation such that the condition holds.

(P4) The element-wise relationships between different matrices can be reasonably interpolated using a selected $\boldsymbol{W}$ (note that this is hard to quantify because errors in these coefficients might not result in large errors in the key outputs).

(P5) At various validation points not in $\boldsymbol{W}$, the error between the actual linearized system at $w_o$ and the interpolated system $\Sigma_w$ quantified by the $H_\infty$ norm is below a tolerance $\epsilon$:

$$\|\boldsymbol{G}_o(s) - \boldsymbol{G}_w(s)\|_{H_\infty} \leq \epsilon \tag{5.11}$$

where $\boldsymbol{G}_o(s)$ and $\boldsymbol{G}_w(s)$ are the transfer function matrices for $\Sigma_o$ and $\Sigma_w$, respectively. Note that this error metric better captures the input/output error in the system.

(P6) Time-domain simulations between the nonlinear $\Sigma$ and LPV $\Sigma_w$ should be similar.

At this time, the selection of $\boldsymbol{W}$ was informed by expert intuition and figures like Fig. 5.2 that characterize the different regions of operation and their transition points. Future work will consider automated sampling strategies that try to optimally sample points for constructing an accurate LPV using the condition in Eq. (5.11).

## 5.4 LPV Model Validation for IEA-15 MW Turbine

The International Energy Agency (IEA) 15 MW offshore wind turbine is a reference turbine model jointly developed by NREL and Danish Technical University (DTU) [3, 113], visualized in Fig. 5.1. The turbine is supported by a floating semisubmersible platform, and a chain catenary mooring system, and the details of the support structure is available in Ref. [114]. It is meant to serve as an open benchmark for the offshore wind community. All the studies in this work are done on the linearized models derived from this reference turbine.

### 5.4.1 WEIS Toolbox

Wind Energy with Integrated Servo-control (WEIS) enables the co-design of the physical plant and control system of FOWT systems, under three different modeling fidelities. The Level 1 modules aim to perform CCD with frequency-domain models of the system. Level 2 used the linearized

time-domain models to perform CCD. Level 3 module aims to perform CCD with closed-loop control trajectories of the full nonlinear system [6]. This work concentrates solely on Level 2, CCD with linearized models.

The WEIS toolbox is built on OpenFAST, another open-source toolbox developed by NREL that generates a full-system dynamic response of FOWTs for wind, wave, and current excitation. The OpenFAST toolbox is built on independent modules that capture the important physical phenomena of the different FOWT subsystems and couplings between them. There are different modules to capture the effects of aerodynamic, hydrodynamic, servodynamic, and mooring dynamic response of the system to these excitations.

The linearization capabilities of WEIS come from linearization capabilities built within OpenFAST. Briefly, this process involves

1. Identifying a suitable operating point
2. Linearizing the nonlinear physics of each module around this point
3. Linearizing the module to module input-output coupling relationships about the operating point
4. Combining all the linearized matrices into the full-system linear state-space model

A detailed discussion regarding the linearization capabilities of OpenFAST and the code for it can be found in the following references [106, 107, 112].

### 5.4.2 Processing the Linearized Models

Wind speed is used to select the state and control operating points for this linearized model. The standard model is then linearized using the linearization capabilities of OpenFAST. With linearized models, it is common practice to time average them, before using the models for analysis. However, direct time averaging eliminates the periodic terms that contribute to system dynamics and leads to inaccurate results. Thus, after linearizing, it is crucial to perform an operation called

multi-blade coordinate (MBC) transformation. MBC captures the cumulative dynamics of the rotor blades and the interaction of the rotor with the tower-nacelle subsystem. It makes the system matrices well conditioned by eliminating nonessential periodicity and performing the filtering operation. More details about this process and the code are available at Ref. [115]. The final output is a state-space model, with state, control, output and disturbance matrices corresponding to the operating point, given by $\boldsymbol{A}_o, \boldsymbol{B}_o, \boldsymbol{C}_o, \boldsymbol{D}_o$ respectively.

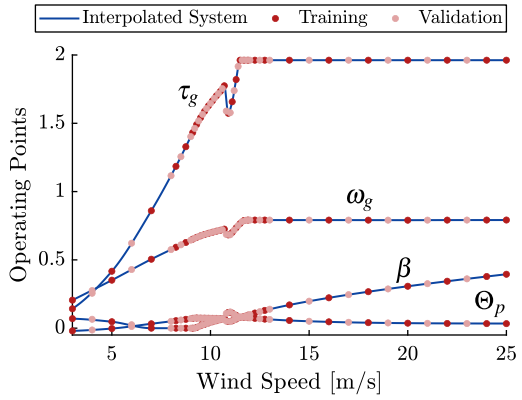
Two keys states in this system are the generator speed $\omega_g$ and platform pitch $\Theta_p$. In its current form, the model is excited by wind inputs only; wave and current disturbances are not considered. Correspondingly, the total inputs to the system are the wind speed $w$, the generator torque $\tau_g$, and the blade pitch $\beta$:

$$\boldsymbol{u}(t) = \begin{bmatrix} w & \tau_g & \beta \end{bmatrix}^T \tag{5.12}$$

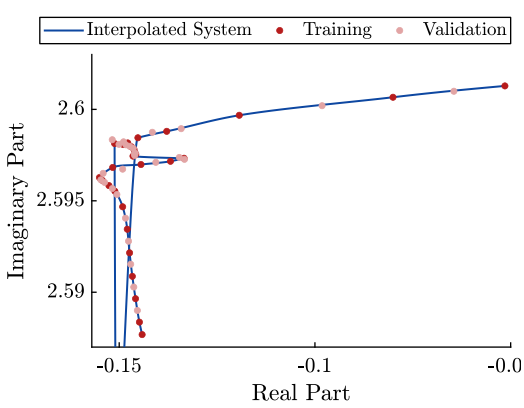
For the considered system, the OpenFAST tool can provide accurate simulations of the nonlinear dynamics of the system (i.e., the outputs of $\Sigma$). To alleviate the concerns expressed in the previous sections, a LPV model is a less computationally-expensive and structured alternative to these expensive simulations. The natural choice for the parameter used to construct the LPV model $\Sigma_w$ is the wind speed. The operating region of a wind turbine is between the cut-in wind speed ($w_{\min} = 3$ [m/s] in this study) and the cut-out wind speed ($w_{\max} = 25$ [m/s]). The system exhibits highly nonlinear behaviour in the transition region and to ensure accuracy while modeling, more points were sampled in the region between rated wind speed and below-rated wind speed. To understand the accuracy of the LPV modeling approach for this system and to test if the model satisfies the given properties, several comparison studies were made.

### 5.4.3 State-Space Model Comparisons

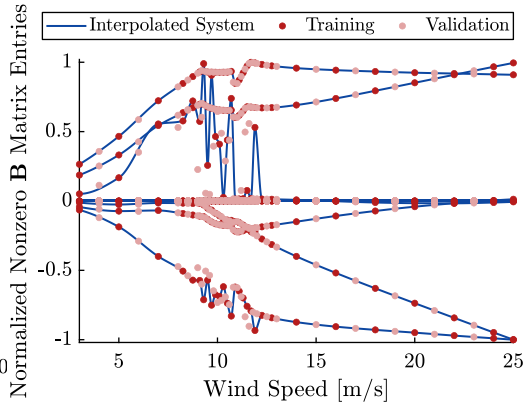
With a selected $\boldsymbol{W}$ (56 distinct wind speeds in this study), the set of linearized state-space models $\boldsymbol{\Omega}$ at each of the wind speed values are obtained. To construct the continuous $\Sigma_w$ using $\boldsymbol{W}$ and $\boldsymbol{\Omega}$, direct element-wise interpolation of the matrices $(\boldsymbol{A}_o, \boldsymbol{B}_o, \boldsymbol{C}_o, \boldsymbol{D}_o)$ was used. To reduce the




**(a)** Select stationary operating point values.




**(b)** One eigenvalue of $\boldsymbol{A}(w)$.

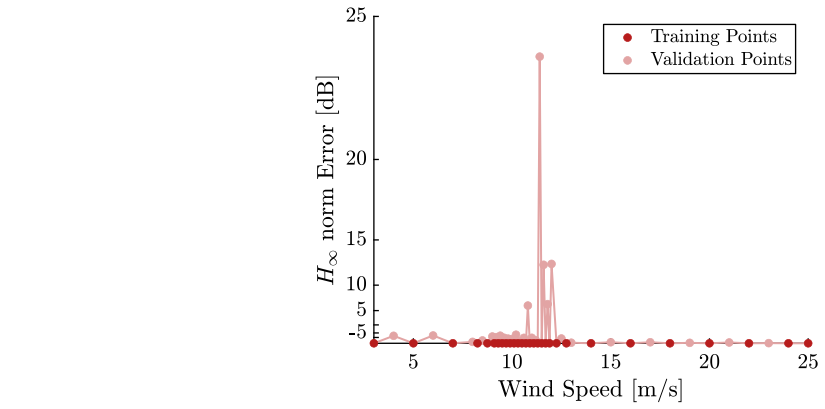



**(c)** Interpolated matrix entries of $\boldsymbol{B}(w)$.

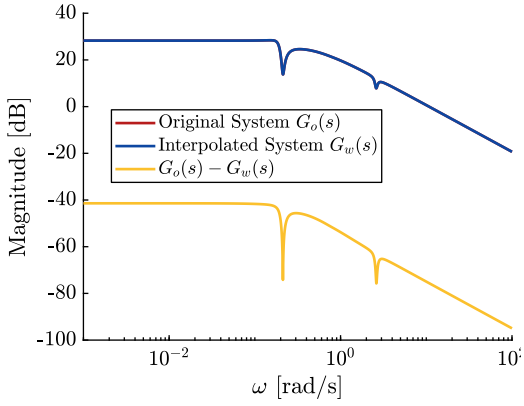
**Figure 5.3:** Select stationary points, eigenvalues, and input matrix for $\Sigma_w$ for the IEA-15.

interpolation costs, the sparsity pattern of the matrices were considered. Only entries with nonzero values were interpolated (and the sparsity pattern remained similar (P2)).

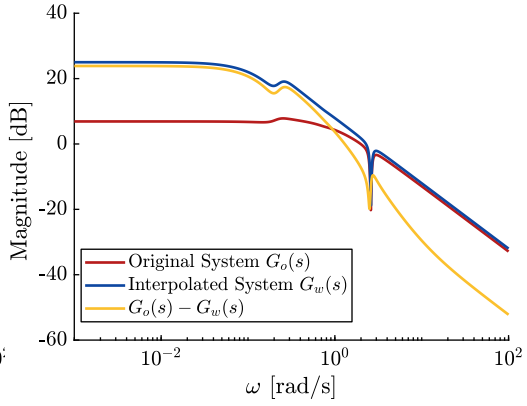
To understand the predictive accuracy of this approach, and check if these models satisfy (P4), the following test is carried out. Every alternate point in $\boldsymbol{W}$ was chosen as a training data for the interpolation procedure, and the values in-between are selected as validation points. This allows us to assess if the interpolation approach can predict matrix properties by comparing to the validation systems. In Fig. 5.3a, several key $\boldsymbol{\xi}_o(w)$ and $\boldsymbol{u}_o(w)$ are shown and there is good agreement between the interpolated LPV system and the validation points, even in the transition region. In Fig. 5.3b, one of the eigenvalues of $\boldsymbol{A}(w)$ that changes with the wind is shown. Again, the eigenvalues generally are well predicted with the largest errors in the transition region. Finally, the normalized




**(a)** $L_\infty$ error using validation points.




**(b)** Close prediction at $w = 23$ [m/s] between blade pitch and generator speed.




**(c)** Largest $H_\infty$ error at $w = 11.5$ [m/s] between blade pitch and generator speed.

**Figure 5.4:** Transfer function-based comparisons using the validation wind speed values for the IEA-15.

nonzero entries of $\boldsymbol{B}(w)$ are shown in Fig. 5.3c, and there are some validation points with high error in the transition region but good agreement in the other regions.

### 5.4.4 Frequency-Domain Verification

The transfer function matrix of the LPV models was studied to better understand if the input/output relationship is accurately predicted and to compute the error in Eq. (5.11) in (P5). Here, we consider the six relationships between the two key states $\omega_g$ and $\Theta_p$ and the inputs $\boldsymbol{u}$.

The $H_\infty$ norm error between the training and validation systems and the interpolated systems is shown in Fig. 5.4a. The errors at the training points is near zero, as expected using interpolation. However, the systems derived from the transition region (8–12 [m/s]) have the highest error when compared to the other regions. This figure shows how advanced sampling strategies could be used

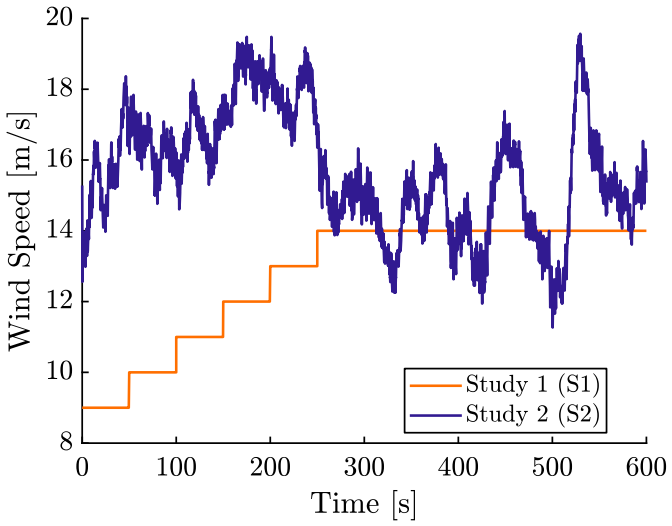


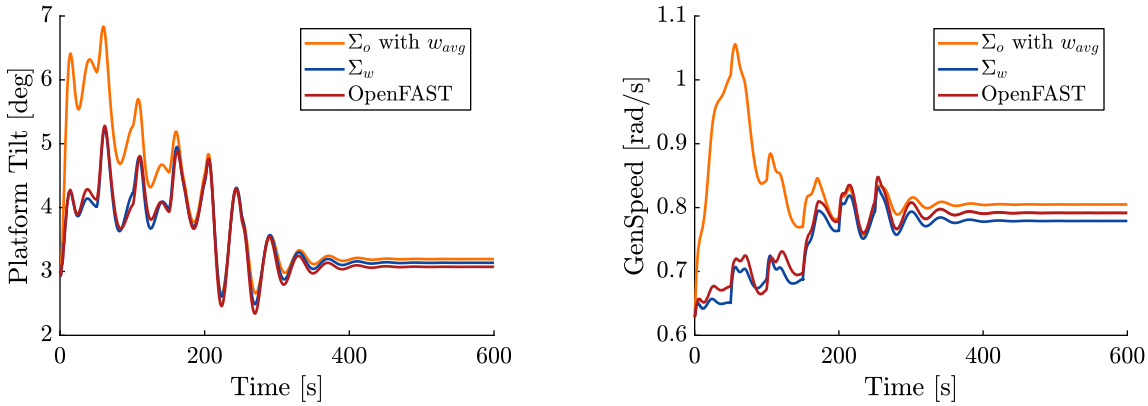

**Figure 5.5:** Two different wind inputs used in the Time-Domain Verification simulations.

to better sample from regions of high error. Additionally, the transfer functions between $\beta$ and $\omega_g$ are shown in Figs. 5.4b and 5.4c with a close prediction and largest $H_\infty$ error, respectively.

### 5.4.5 Time-Domain Verification

The final comparisons were based on (P6) using OpenFAST to determine the nonlinear response of $\Sigma$. Using the same input trajectories, three different models ($\Sigma$, $\Sigma_w$, and $\Sigma_o$ using the average wind speed $w_{avg}$) are simulated, and then the resulting state trajectories are compared. Two different input sets were simulated. The wind inputs (step-like and turbulent) are shown in Fig. 5.5 (and the nonzero trajectories for $\beta$ and $\tau_f$ are not shown).

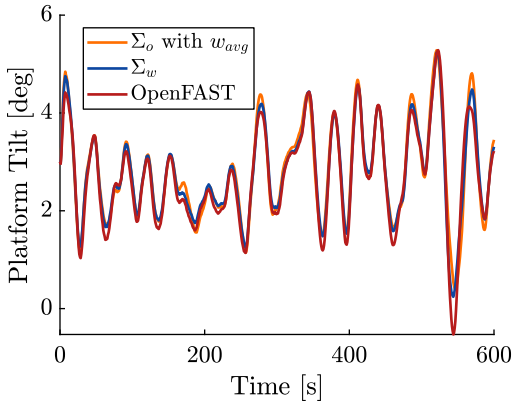
From the results, we see that $\Sigma_w$ captures the nonlinear response from OpenFAST more accurately that $\Sigma_o$ using $w_{avg}$. In the first study (S1), $w_{avg} = 12.8$ [m/s]. Early in the simulation, when the wind speed value is far away from $w_{avg}$, we see that the $\Sigma_o$ using $w_{avg}$ produces inaccurate results for $\Theta_p$ in Fig. 5.6a and $\omega_g$ in Fig. 5.6b. In the second study (S2), $w_{avg} = 15.7$ [m/s], and the wind profile is generally in a region where the dynamics are more predictable between wind speeds (cf. Figs. 5.3 and 5.4). Because of this property, all the model responses are similar, but again $\Sigma_w$ more closely follows $\Sigma$. The simulation results for $\Theta_p$ and $\omega_g$ are in Figs. 5.6c and 5.6d,



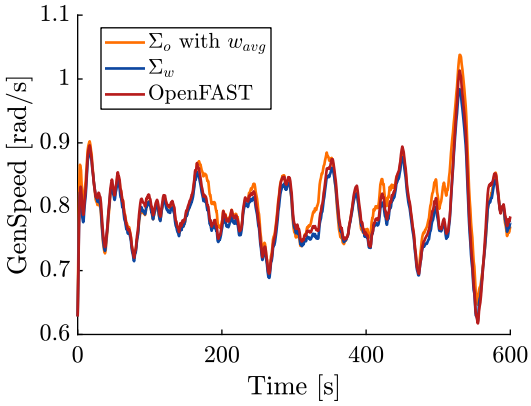

**(a)** Platform pitch (S1).




**(b)** Generator speed (S1).




**(c)** Platform pitch (S2).



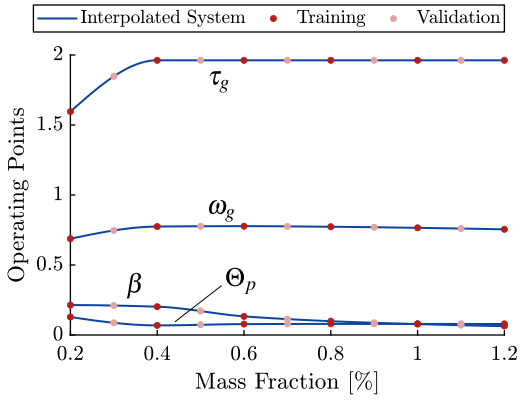

**(d)** Generator speed (S2).

**Figure 5.6:** Model validation simulations between nonlinear $\Sigma$, LPV $\Sigma_w$, and LTI $\Sigma_o$ using $w_{avg}$ models.

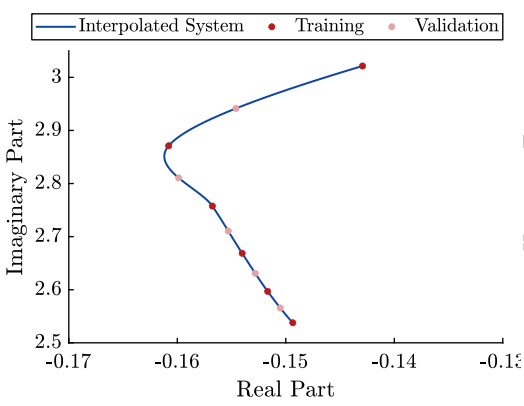
respectively. Using all the different comparisons, it was concluded that the LPV model $\Sigma_w$ can with reasonable accuracy, capture the dynamics of the considered FOWT.

### 5.4.6 Interpolation Based on Platform Mass

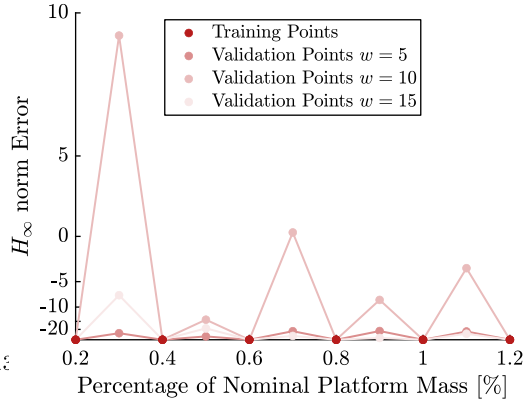
The model $\Sigma_w$ just presented was obtained using a particular instance of the system's plant design, denoted by $\boldsymbol{x}_p$ in Sec. 5.2.1. However, we also want to consider the design impacts of a single plant design parameter, namely the platform mass. For such an investigation, using a collection of various values of the platform mass $\boldsymbol{M}$, a full set of linear models $\boldsymbol{\Omega}$ corresponding to $\boldsymbol{W}$ are obtained. A similar interpolation scheme and analysis are carried out in this additional dimension.




**(a)** Select stationary operating point values .




**(b)** One eigenvalue of $\boldsymbol{A}(m_r)$ .




**(c)** $H_\infty$ error using validation points.

**Figure 5.7:** Select stationary points, eigenvalues , and $H_\infty$ error for all eleven $\boldsymbol{m}_r$ for the IEA-15.

Similar tests to the ones outlined in the previous sections were carried out to check the predictive accuracy of interpolation based on the platform's mass. Denoting the nominal platform mass as $m_r = 1$, a total of eleven mass value fractions between 0.2 and 1.2 were used in this study. The nominal platform specifications are available at Ref. [114]. Stationary operating points, eigenvalues of $\boldsymbol{A}(w)$ for $w = 11.4$ [m/s], and the $H_\infty$ norm error between the training and validation mass points for wind speeds in the three different regions are shown in Fig. 5.7. From these results, we see that interpolation based on $\boldsymbol{M}$ is generally well behaved, potentially more so than the wind speed dimension.

## 5.5 Control Co-Design Problem Formulation

This section describes the nested CCD problem constructed using the LPV models from Sec. 5.3 to study the impact of various stability constraints on the LCOE for the considered single device FOWT.

### 5.5.1 Outer-Loop Plant Design Problem Formulation

The outer-loop plant optimization problem in the nested CCD approach employed here is centered around the LCOE calculation in Eq. (5.1). In this calculation, the total lifetime cost is estimated as follows:

$$C_n = \frac{I(\boldsymbol{x}_p)}{(1+r)} + \sum_{k=1}^{n} \frac{O}{(1+r)^k} \tag{5.13}$$

where $I(\boldsymbol{x}_p)$ is the investment cost that depends on the plant design, $O$ are the annual operating costs, $r$ is the annual discount rate, and $n$ is the expected lifetime of the system in years.

Modelling the cost of different wind turbine elements along with the balance of system cost is a specialization in itself. Commonly, these estimates are made with inputs from different vendors and component manufacturers. For this study, we created a low-fidelity cost and scaling model for the blades, generator, nacelle, and tower from Refs. [116–119].

Here, we will be considering platform mass as the key plant design variable. Different platform studies use different cost models, and these cost models depend on the cost per tonne of the materials used to construct the platform. The IEA-15 turbine used in this study is a semisubmersible platform. The constituent materials used for constructing the platform are steel, fixed ballast, and outfitting, and their nominal masses and costs per tonne are from Ref. [114], The cost of the platform relative to the nominal platform design is:

$$C_{\text{platform}} = M_{\text{steel}} C_{\text{steel}} + M_{\text{FB}} C_{\text{FB}} + M_{\text{outfit}} C_{\text{outfit}} \tag{5.14a}$$

$$I(\boldsymbol{x}_p) = C_{\text{sys}} + m_r C_{\text{platform}}(\boldsymbol{x}_p) \tag{5.14b}$$

where $m_r$ is a unitless ratio between the platform's mass and the nominal platform mass.

The weights are obtained from Ref. [114].

**Table 5.1:** Nominal platform cost per unit weight and mass for the constituent materials [1, 2].

| Material | Cost [$/t] | Mass [t] |
|---|---|---|
| Steel | 3600 | 3914 |
| Fixed Ballast | 150 | 2540 |
| Outfitting | 7250 | 100 |

The total lifetime energy output is determined using a representative year long energy production calculation using $m$ operational scenarios:

$$E = \sum_{k=1}^{m} \tau_k \bar{P}_k^*(\boldsymbol{x}_p) \tag{5.15}$$

where $k$ is the operational scenario, $\bar{P}_k^*(\boldsymbol{x}_p)$ is the expected average power, and $\tau_k$ is the expected amount of time over one year for scenario $k$. The expected average power for scenario $k$ is determined by optimizing the dynamics and control for the given plant design. This control-focused subproblem of the nested CCD coordination strategy will be defined in the next section. The discount rate is accounted for in a similar manner as Eq. (5.13):

Finally, the annual energy production (AEP) is computed with:

$$\text{AEP} = \eta_u E \tag{5.16}$$

where $0 \leq \eta_u \leq 1$ is the expected uptime ratio.

Then the expected total energy output over $n$ years is:

$$E_n = \sum_{k=1}^{n} \frac{E}{(1+r)^k} \tag{5.17}$$

Therefore, LCOE = $C_n/E_n$, and the complete outer-loop optimization problem is:

$$\min_{\boldsymbol{x}_p} \quad \text{LCOE}(\boldsymbol{x}_p) \tag{5.18a}$$

$$\text{subject to:} \quad \boldsymbol{L}_p \leq \boldsymbol{x}_p \leq \boldsymbol{U}_p \tag{5.18b}$$

where only simply upper and lower bounds on the plant variables are considered at this time (although more complex plant-only constraints can be readily incorporated).

Note that for a fixed plant, the solution for each $\bar{P}_k^*(\boldsymbol{x}_p)$ can be determined through independent minimization problems. Therefore, the control subproblems can be solved in parallel, reducing computational costs.

### 5.5.2 Control Subproblem for a given Design Load Case

The main outcome from the control subproblem is to understand the impact of the control decisions on system response, power production, and ultimately the LCOE design objective. An open-loop optimal control problem is constructed to maximize the power produced for a given operational scenario or design load case (DLC). The optimization formulation is presented using the original $(\boldsymbol{\xi}, \boldsymbol{u})$, but the linear time-varying transformation in Eq. (5.5) based on the wind-dependent operating point is applied so that $(\boldsymbol{\xi}_\Delta, \boldsymbol{u}_\Delta)$ are the states and controls for this subproblem.

The energy produced by the turbine is:

$$\int_0^{t_f} P(t)\mathrm{d}t = \int_0^{t_f} \eta_g \tau_g(t)\omega_g(t)\mathrm{d}t \tag{5.19}$$

where $\eta_g$ is the generator efficiency. Note, the control term $\tau_g$ appears linearly in the objective term Eq. (5.19). The presence of linear control terms in the objective function with linear dynamics can give rise to singular arcs [19] as the control trajectory cannot be uniquely determined. To help mitigate this issue, a quadratic penalty term is introduced in the objective term:

$$\Pi(t) = \boldsymbol{u}^T \begin{bmatrix} 10^{-8} & 0 \\ 0 & 10^{8} \end{bmatrix} \boldsymbol{u} \tag{5.20}$$

where values in this penalty matrix were identified according to the method discussed in Ref. [14].

The linear dynamic constraints included using $\Sigma_w$ from Eq. (5.8) are:

$$\frac{d\boldsymbol{\xi}_\Delta}{dt} = \boldsymbol{A}(w)\boldsymbol{\xi}_\Delta + \boldsymbol{B}(w)\boldsymbol{u}_\Delta - \frac{\partial \boldsymbol{\xi}_o(w)}{\partial w}\frac{dw}{dt} \tag{5.21}$$

where the initial state values correspond to the state operating points for $w(0)$:

$$\boldsymbol{\xi}(0) = \boldsymbol{\xi}_o(w(0)), \text{ or equivalently } \boldsymbol{\xi}_\Delta(0) = \boldsymbol{0} \tag{5.22}$$

In order to protect the generator components from excess electrical loads and the nacelle from the dynamic loads, an upper bound for generator speed $\omega_g$ is set constraining the generator speed

to the rated speed of the turbine:

$$\omega_g(t) \leq \omega_{g,\max} \tag{5.23}$$

To account for the stability of the system, an upper bound on the platform pitch tilt $\Theta_p$ is included:

$$\Theta_p(t) \leq \Theta_{p,\max} \tag{5.24}$$

Maximum and minimum value constraints are placed on the controls blade pitch $\beta$ and the generator torque $\tau_g$, according to the values prescribed in Ref. [3] :

$$0 \leq \tau_g(t) \leq \tau_{g,\max} \tag{5.25a}$$

$$0 \leq \beta(t) \leq \beta_{\max} \tag{5.25b}$$

An additional constraint on the pitch rate is included to ensure that the rate of change of the blade pitch is within practical limits:

$$\frac{d\beta}{dt} \leq \dot{\beta}_{\max} \tag{5.26}$$

Another constraint is included to ensure the power generated by the turbine does not exceed the rated power. Here we approximate the nonlinear power constraint $\tau_g\omega_g \leq P_{\max}$ with the following linear path constraint:

$$\tau_g\omega_{g,\max} + \tau_{g,\max}\omega_g \leq P_{\max} + \tau_{g,\max}\omega_{g,\max} \tag{5.27}$$

Now, the complete control subproblem formulation is:

$$\min_{\boldsymbol{u}_\Delta, \boldsymbol{\xi}_\Delta} \quad \int_0^{t_f} \left(-P(t) + \Pi(t)\right) \mathrm{d}t \tag{5.28a}$$

$$\text{subject to:} \quad \text{Eqs. } (5.21) - (5.27) \tag{5.28b}$$

which will yield the average power $\bar{P}^* = \int_0^{t_f} P(t)\mathrm{d}t/t_f$ needed in Eq. (5.15). It can be observed that Problem (5.28) has only quadratic objective function terms and linear constraints; therefore, can be classified as a LQDO problem (see Sec. 5.2.4).

The linear-quadratic optimal control problem was solved using *DTQP* [54].

The values for the parameters in the constraints are given in the table Tab. 5.2:

**Table 5.2:** Problem parameters.

| Variable | Value | Units |
|---|---|---|
| $\omega_{g,\max}$ | 0.7913 | [rad/s] |
| $\Theta_{p,\max}$ | 6 | [deg] |
| $P_{\max}$ | 15 | [MW] |
| $\tau_{g,\max}$ | 19.62 | [MNm] |
| $\beta_{\max}$ | 0.3948 | [rad] |
| $\eta_g$ | 96.55 | % |

## 5.6 Results

In this section, we describe the results of an LCOE-focused CCD study using the IEA-15 turbine [3] supported by a floating semisubmersible platform [114]. The values for the CCD problem parameters defined in Sec. 5.5.2 are given in Table 5.2. Here, we consider six design load cases (DLCs) based on the input wind speed trajectories shown in Fig. 5.8. The average power values are combined using Eq. (5.15), with the weights (in days):

$$\tau = \begin{bmatrix} 23.0 & 92.2 & 115.3 & 54.0 & 46.1 & 34.6 \end{bmatrix} \tag{5.29}$$

which correspond to an average wind speed of $13.4$ [m/s], and the wind distribution is approximately a Weibull distribution.

The LQDO optimal control problems based on Problem (5.28) are solved using `DTQP`, an open-source `Matlab`-based toolbox using the direct transcription (DT) method and quadratic programming [8, 54]. Each problem was discretized using 5000 mesh points.

A single critical plant design variable was considered as this time, namely the platform's mass. A sensitivity approach was used to explore how the plant design decisions impact the system cost and performance. More interpolated mass values were added near the lower bound of $\boldsymbol{m}_r$. In order to understand the impact of platform mass on the system stability, power production, and subsequently the LCOE, several constraint bounds on the platform pitch tilt $\Theta_p$ were explored. More specifically, $\Theta_p$ was constrained to four different values between $3-6^\circ$. Although no wave/current

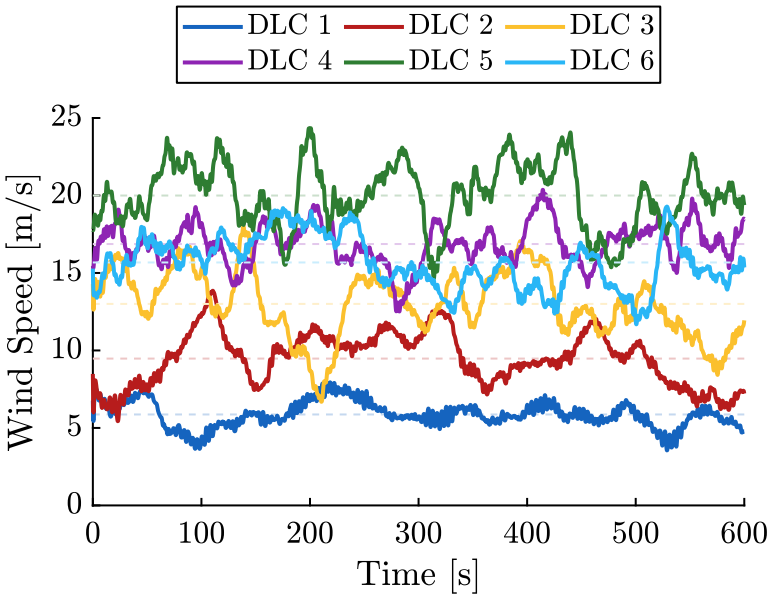


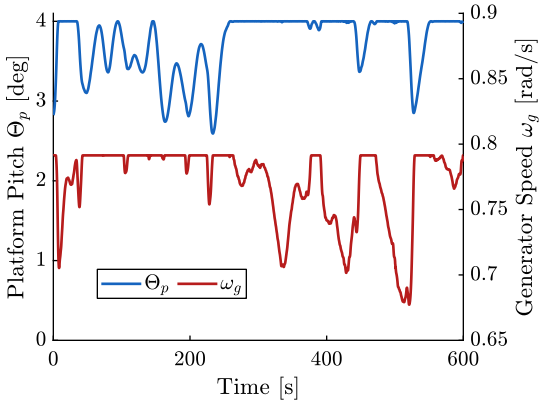

**Figure 5.8:** Design load cases considered based on an input wind speed trajectory.

forces are included as disturbances at this time, these different constraint values on $\Theta_p$ will roughly indicate performance in more dynamic wave and current conditions.

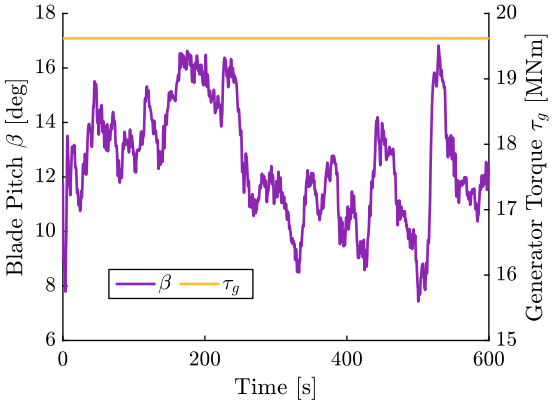
Overall, there were $32 \times 6 \times 4 = 768$ inner-loop control subproblems solved for each combination of mass, DLC, and $\Theta_{p,\max}$. The computational cost was 37 minutes on a desktop workstation with an AMD 3970X CPU, 128 GB DDR4 2666 MHz RAM, `Matlab` 2021a update 1, and `Windows` 10 build 17763.1790. The code for inner loop studies mentioned in the previous sections is available at Ref. [54].

### 5.6.1 Results for a Single Control Subproblem

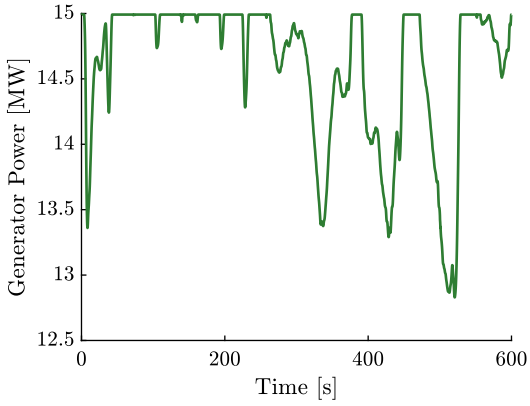
Figure 5.9 summarizes the optimal control results for 1 of the 264 problems with $m_r = 0.7$, DLC 6, and $\Theta_p \leq 4°$. The optimal trajectories for the generator speed and platform pitch are in Fig. 5.9a. We see that the constraint $\Theta_p \leq 4°$ and others in Table 5.2 are satisfied. DLC 6 is in the rated-power region, so we might expect pitch control to be active and the generator torque to be held roughly constant in Fig. 5.9b [100]. However, to satisfy the platform pitch constraints, we see that the generator speed does need to decrease when the pitch constraint becomes active.




**(a)** Select states $\omega_g$ and $\Theta_p$.



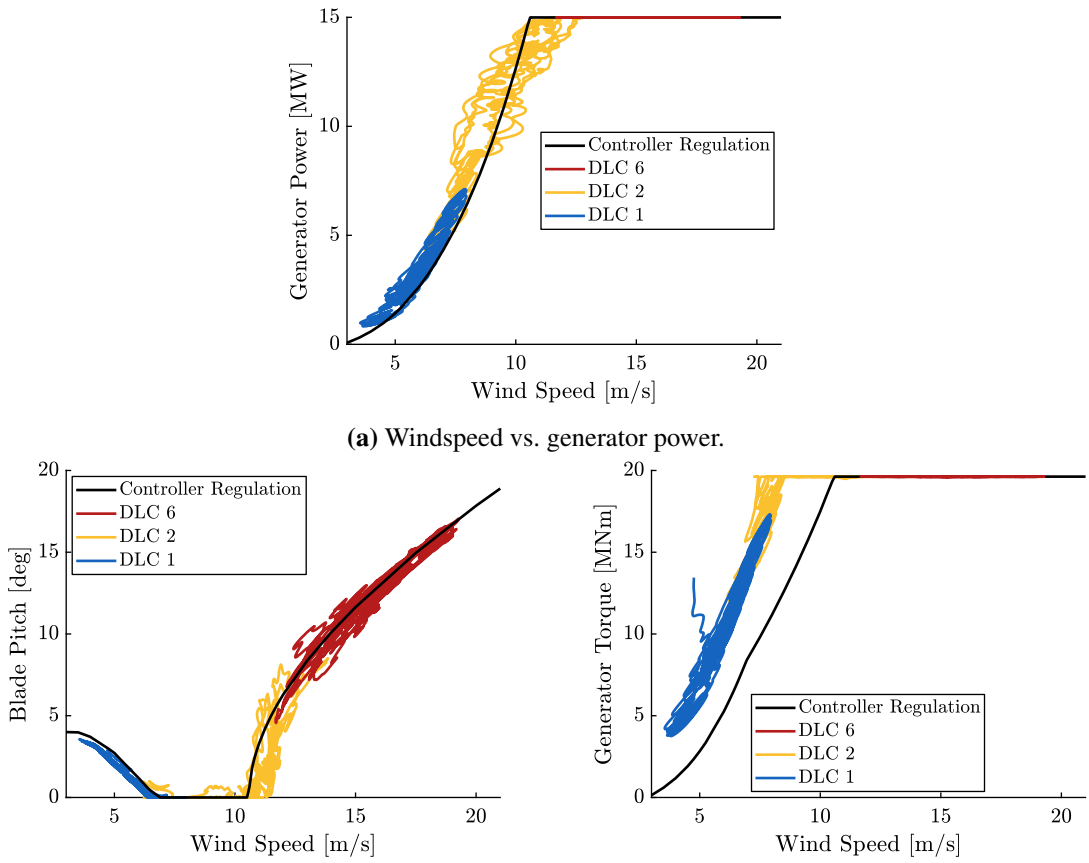

**(b)** Controls $\beta$ and $\tau_g$.




**(c)** Power $\tau_g \omega_g$.

**Figure 5.9:** Optimal control results with $m_r = 0.7$, DLC 6, and $\Theta_p \leq 4°$.

To better understand the optimal control results in other operating regions, Fig. 5.10 was constructed to show the behavior with nominal mass $m_r = 1$ and the largest pitch constraint value $\Theta_p \leq 6°$. We see in Figs. 5.10b and 5.10c that the results generally follow the expected trends when compared to the controller regulation from Fig. 5.2. Overall, the optimization-based approach seems to favor larger torque values and slower speeds than the regulation curves. The results from the DLCs in the below-rated and transition regions are encouraging, as a combination of torque and pitch control is utilized. In some regions, the pitch control is active while torque is held constant and vice versa. Therefore, the optimizer identifies results for all regions in agreement with traditional wind turbine controls. Overall, these results, in combination with the model validation in Sec. 5.4, demonstrate the validity of the considered LPV models in FOWT open-loop control studies.




**(a)** Windspeed vs. generator power.

**(b)** Windspeed vs. $\beta$.

**(c)** Windspeed vs. $\tau_g$.

**Figure 5.10:** Select optimal control results using LPV model vs. controller regulation curves with $m_r = 1$ and $\Theta_p \leq 6°$.

### 5.6.2 Average Output Power vs. Platform Mass

In Fig. 5.11, the trends between the average power $\bar{P}_6^*(m_r)$ for DLC 6 are shown for the four tested values of $\Theta_{p,\max}$. The primary method used to control the platform pitch and ensure system stability is the blade pitch $\beta$, but $\beta$ is also tightly coupled to the generator speed. To satisfy smaller, more challenging values of $\Theta_{p,\max}$, the optimal control solution has higher values of blade pitch, sacrificing generator speed. Thus, for these more challenging constraint values, the power produced is lower on average.

Additionally, the platform mass has a significant effect on the average power production. We see that heavier platforms satisfy the stability constraints with little to no compromise on power generation (i.e., average power is nearly at 15 MW). In comparison, lighter platforms have to

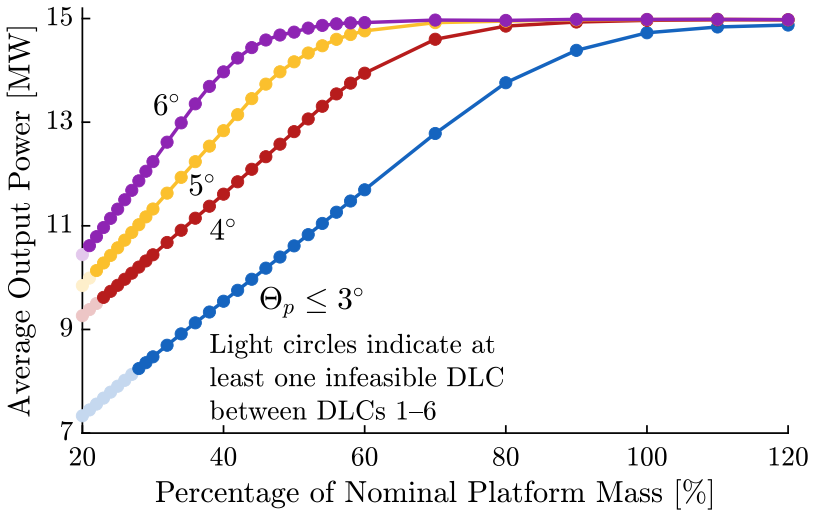


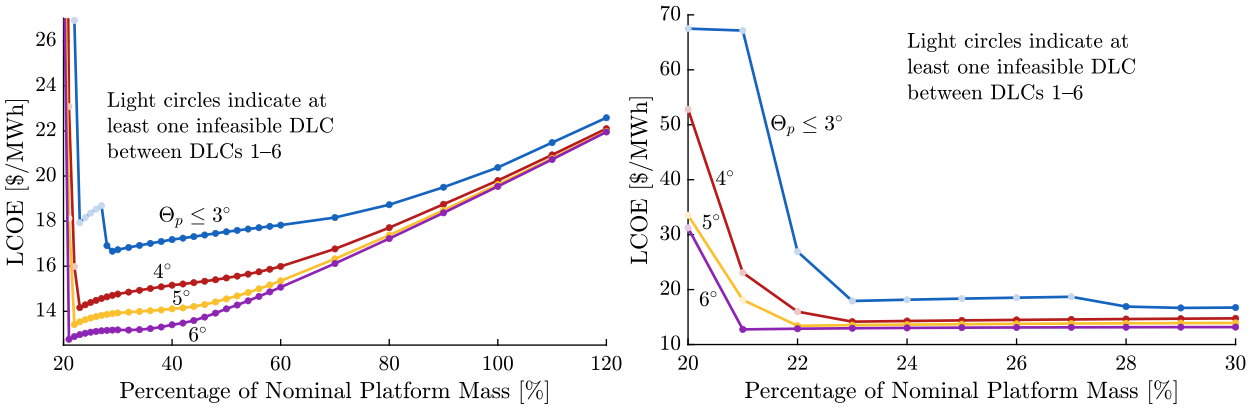

**Figure 5.11:** Average output power for DLC 6 vs. platform mass.

sacrifice power generation. Furthermore, in Fig. 5.11 several points are marked in a lighter shade to indicate that at least one infeasible DLC was identified, i.e., under the constraints imposed in the control subproblem and with the given platform mass, no feasible solution was found[6]. Two trends are observed for these infeasible cases; they happen predominantly for platforms with lower masses and DLCs in the transition region, e.g., DLCs 2 and 3 from Fig. 5.8. As mentioned previously, the blade pitch is used to control the platform pitch. For platforms with lower masses, the pitch control needs to be close to or at its maximum value to satisfy the tighter platform pitch constraints. For the infeasible cases, the maximum allowable pitch value is not sufficient to satisfy the constraints.

### 5.6.3 LCOE vs. Platform Mass

Now, combining the DLCs using the weighting scheme in Eq. (5.15), we can determine the total energy output. In addition, utilizing the total cost model from Sec. 5.5.1, LCOE can be estimated. These estimates of LCOE have been calculated for $n = 30$ years with a $r = 7\%$ discount rate. It is important to emphasize that this estimate of LCOE only considers the capital cost of the turbine and platform components and maintenance cost. Other "balance of system" costs [117] are not considered for this study as we are only looking at a single turbine. As mentioned previously,

[6]These infeasible cases are being further investigated, as we believe that infeasibility may be due to several factors including the physical constraints and the selected DT discretization.




**(a)** LCOE vs. platform mass.

**(b)** LCOE vs. platform mass for lower mass percentages.

**Figure 5.12:** LCOE vs. platform mass.

some values of the constraints are infeasible, and the infeasible results are included with zero generated energy. The summarized LCOE results are shown in Fig. 5.12.

From these results, we see that the optimal value for LCOE depends on the stability constraint; for large values of $\Theta_{p,\max}$, the optimal mass decreases. This is because as the platform mass increases, the capital cost increases. But, as indicated in Fig. 5.11, lower mass values can still result in nearly maximum average power production. Therefore, there is some optimal mass point where the conflicting decisions of increasing energy production are balanced with increasing platform capital costs.

For the considered reference IEA-15 turbine described in Refs. [3, 114], $\Theta_p$ was constrained to $6°$ using the nominal platform mass. While keeping the other plant parameters constant, we see that the lowest LCOE, in this case, is achieved using $40\%$ of the nominal platform mass. However, this result is subject to modeling assumptions, optimal control operation, and lack of safety factors, but it can still help guide the final design. Additionally, the hydrodynamic and hydrostatic stability of the different platforms has not been evaluated in this study. These investigations will also limit the bounds on the platform mass and impact the final design.

---

This study highlighted the importance of taking into account the effect of advanced control strategies and integrated design approaches on LCOE-based design studies. We see that optimal

control strategies can help reduce the plant's overall cost, making FOWTs more economically feasible. Additionally, CCD proves to be a rigorous integrated design approach that can be used to design complex coupled systems like a FOWT.

# Chapter 6

# Summary and Conclusion

This thesis concentrated on finding efficient methods to construct and solve DO problems with linear-quadratic elements and simultaneous and nested CCD problems using direct transcription. There are three main studies undertaken in this thesis. They are summarised individually in the following sections.

## 6.1 Summary for Study 1: Using LQDO Elements in Solving NLDO Problems

The first study presented in Chapter 3 explored the various uses of linear-quadratic dynamic optimization (LQDO) and ordinary linear-quadratic (OLQ) elements in the context of nonlinear dynamic optimization (NLDO) and how these elements can be used to efficiently solve NLDO problems. In this study three general classes of strategies for the use of LQDO elements in solving NLDO problems were discussed. The first was the direct incorporation of OLQ elements into DT-based NLDO tools with the appropriate matrices. The case studies demonstrated that there is some computational benefit of their direct incorporation using either symbolic derivatives or finite-difference methods. There was generally decreasing improvements as the problem's mesh size increased and select instances where performance was slightly worse because of the differences in the derivative calculations. Additionally, different two-level optimization problems that have LQDO subproblems were discussed, including the popular nested control co-design method. This particular nested scheme is applicable to the class of NLDO problems where for fixed values of $\boldsymbol{x}_p$, the resulting subproblem is a convex LQDO problem and was investigated in the final two case studies. The nested method was compared to solutions which use finite-difference derivative methods, and was shown to be faster. Finally, the quasilinearization method was presented where the linearization of the constraints and quadratization of the objective function are performed with

respect to some reference trajectory, creating a reference LQDO problem. In the first case study, the quasilinearization method was shown to be faster than the interior-point NLP method for smaller mesh sizes. However, this approach did not converge for some of the case study problems, which included parameters. Therefore, care must be taken when utilizing the quasilinearization for solving NLDO problems.

There are a variety of potential future work items. Investigating methods that improve the convergence properties of the quasilinearization without sacrificing the observed benefits could lead to more efficient and robust NLDO tools. The impetus of the methods discussed in this article is to further exploit the specific mathematical problem structure of NLDO problems with limited or no disadvantages.

## 6.2 Summary for Study 2: Comparison of Nested and Simultaneous CCD

The second study in Chapter 4 built on the results from the previous study and established the methods and results for a fair comparison between simultaneous and nested control co-design methods. In Chapter 3 we identified that all-at-once or simultaneous optimization is computationally efficient and accurate when using symbolic derivatives, and if such high quality derivative information is not available, then bi-level or nested optimization problems are preferred. These insights were then applied to the problem of CCD of active suspension using both simultaneous and nested CCD. The implementation and results from these studies were then the basis of a fair comparison between these methods. The varied results of the case study showed that the simultaneous strategy using symbolic derivatives was generally superior while the nested strategy was preferable when any other derivative method was used with the simultaneous strategy. However, special care is needed to handle potentially infeasible inner loops.

It is important to highlight that this case study uses an LQDO-amendable CCD problem so the structure of the inner-loop problem permits efficient solution methods. For other general CCD problems, the observed trends might be different. In general, these results indicate the potential

advantages of capitalizing on problem structure, selecting the most effective implementation, and understanding the goals of a particular CCD study.

Furthermore, it is imperative to understand the trade-offs and goals of a particular CCD study (e.g., desired accuracy) and use sound judgment to make informed decisions on how to solve a chosen CCD problem. Future work will seek to understand these trade-offs and best practices in more diverse set of CCD problem scenarios moving towards a robust set of CCD implementation guidelines. Detailed theoretical and numerical investigations and approachable and effective CCD tools are critical to increasing the applicability and impact of the CCD methodology.

## 6.3 Summary for Study 3: CCD of Floating Offshore Wind Turbines

In this study, we discussed the use of linear parameter-varying (LPV) models for control co-design (CCD) of floating offshore wind turbines (FOWTs). High-fidelity models of FOWTs are described by highly complex and nonlinear models. Unfortunately, these models are often too costly to use in early-stage system design and evaluation. Using linearized models based on these nonlinear systems is a popular method to offset the computational costs involved. Here, we describe a class of linear parameter-varying (LPV) models that realize more accurate predictions of a system's dynamic behavior over a large range of operating points and are shown to be useful for early-stage CCD studies of FOWTs.

The LPV models based on the wind speed parameter showed good general agreement in both nonlinear simulation comparisons and general optimal control trends. The primary study investigated the system's dynamic stability, power production, and ultimately the levelized cost of energy (LCOE). The single plant decision in this study was the platform's mass, and the optimal LCOE results indicated that the platform mass could be reduced to 30–60% of its nominal value and still satisfy the platform pitch constraints. However, several additional factors should be investigated before making a specific recommendation.

It remains future work to incorporate more detailed and sophisticated outer-loop plant design optimization, including the impact of plant decisions like tower hub height and blade length on platform stability and power production in the context of the LCOE. With additional plant decisions, the LCOE calculation should also be amended to reflect the omitted factors. Additionally, we hope to study the effect of wave and current excitations. Finally, in order to address the realizability of the open-loop optimal control solutions, work is needed to realize robust, implementable control systems, which may be informed by the optimal operation identified in this study [6, 31].

# Bibliography


[1] A. Ghigo, L. Cottura, R. Caradonna, G. Bracco, and G. Mattiazzo, "Platform optimization and cost analysis in a floating offshore wind farm," *Journal of Marine Science and Engineering*, vol. 8, no. 11, p. 835, Oct. 2020, doi: 10.3390/jmse8110835

[2] P. Beiter, W. Musial, A. Smith, L. Kilcher, R. Damiani, M. Maness, S. Sirnivas, T. Stehly, V. Gevorgian, M. Mooney, and G. Scott, "A spatial-economic cost-reduction pathway analysis for U.S. offshore wind energy development from 2015–2030," Tech. Rep., Sep. 2016, doi: 10.2172/1324526

[3] E. Gaertner, J. Rinker, L. Sethuraman, F. Zahle, B. Anderson, G. E. Barter, N. J. Abbas, F. Meng, P. Bortolotti, W. Skrzypinski, G. N. Scott, R. Feil, H. Bredmose, K. Dykes, M. Shields, C. Allen, and A. Viselli, "IEA wind TCP task 37: Definition of the IEA 15-megawatt offshore reference wind turbine," Tech. Rep., Mar. 2020, doi: 10.2172/1603478

[4] A. L. Nash and N. Jain, "Hierarchical control co-design using a model fidelity-based decomposition framework," *Journal of Mechanical Design*, vol. 143, no. 1, Aug. 2020, doi: 10.1115/1.4047691

[5] S. Azad, M. Behtash, A. Houshmand, and M. J. Alexander-Ramos, "PHEV powertrain co-design with vehicle performance considerations using MDSDO," *Structural and Multidisciplinary Optimization*, vol. 60, no. 3, pp. 1155–1169, Apr. 2019, doi: 10.1007/s00158-019-02264-0

[6] J. Jonkman, A. Wright, G. Barter, M. Hall, J. T. Allison, and D. R. Herber, "Functional requirements for the WEIS toolset to enable controls co-design of floating offshore wind turbines," in *International Offshore Wind Technical Conference*, no. IOWTC2020-3533, Feb. 2021, url: https://www.nrel.gov/docs/fy21osti/77123.pdf

[7] J. T. Allison, T. Guo, and Z. Han, "Co-design of an active suspension using simultaneous dynamic optimization," *Journal of Mechanical Design*, vol. 136, no. 8, Jun. 2014, doi: 10.1115/1.4027335

[8] D. R. Herber, "Advances in combined architecture, plant, and control design," Ph.D. Dissertation, University of Illinois at Urbana-Champaign, Urbana, IL, USA, Dec. 2017, url: http://hdl.handle.net/2142/99394.

[9] P. Y. Papalambros and D. J. Wilde, *Principles of Optimal Design: Modeling and Computation*, 3rd ed. Cambridge University Press, 2017, doi: 10.1017/9781316451038

[10] M. Arora, "Active suspension co-design for lateral stability of rail vehicles," M.S. Thesis, University of Illinois at Urbana-Champaign, Urbana, IL, USA, Dec. 2017. [Online]. Available: http://hdl.handle.net/2142/99422

[11] J. T. Allison and D. R. Herber, "Multidisciplinary design optimization of dynamic engineering systems," *AIAA Journal*, vol. 52, no. 4, pp. 691–710, Apr. 2014, doi: 10.2514/1.j052182

[12] M. Garcia-Sanz, "Control co-design: An engineering game changer," *Advanced Control for Applications: Engineering and Industrial Systems*, vol. 1, no. 1, p. e18, Oct. 2019, doi: 10.1002/adc2.18

[13] G. E. Barter, A. Robertson, and W. Musial, "A systems engineering vision for floating offshore wind cost optimization," *Renewable Energy Focus*, vol. 34, pp. 1–16, Sep. 2020, doi: 10.1016/j.ref.2020.03.002

[14] D. R. Herber, "Dynamic system design optimization of wave energy converters utilizing direct transcription," M.S. Thesis, University of Illinois at Urbana-Champaign, Urbana, IL, USA, May 2014, url: http://hdl.handle.net/2142/49463.

[15] A. Baheri, J. Deese, and C. Vermillion, "Combined plant and controller design using bayesian optimization: A case study in airborne wind energy systems," in *Volume 3: Vibration in Mechanical Systems; Modeling and Validation; Dynamic Systems and Control*

*Education; Vibrations and Control of Systems; Modeling and Estimation for Vehicle Safety and Integrity; Modeling and Control of IC Engines and Aftertreatment Systems; Unmanned Aerial Vehicles (UAVs) and Their Applications; Dynamics and Control of Renewable Energy Systems; Energy Harvesting; Control of Smart Buildings and Microgrids; Energy Systems*. American Society of Mechanical Engineers, Oct. 2017, doi: 10.1115/DSCC2017-5242

[16] J. Deese, P. Tkacik, and C. Vermillion, "Gaussian process-driven, nested experimental co-design: Theoretical framework and application to an airborne wind energy system," *Journal of Dynamic Systems, Measurement, and Control*, vol. 143, no. 5, Dec. 2020, doi: 10.1115/1.4049011

[17] L. Y. Pao, D. S. Zalkind, D. T. Griffith, M. Chetan, M. S. Selig, G. K. Ananda, C. J. Bay, T. Stehly, and E. Loth, "Control co-design of 13 MW downwind two-bladed rotors to achieve 25% reduction in levelized cost of wind energy," *Annual Reviews in Control*, vol. 51, pp. 331–343, 2021, doi: 10.1016/j.arcontrol.2021.02.001

[18] K. Dykes, R. Meadows, F. Felker, P. Graf, M. M. Hand, M. Lunacek, J. Michalakes, P. Moriarty, W. Musial, and P. Veers, "Applications of systems engineering to the research, design, and development of wind energy systems," Tech. Rep., Dec. 2011, doi: 10.2172/1032664

[19] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. Society for Industrial and Applied Mathematics, Jan. 2010, doi: 10.1137/1.9780898718577

[20] L. T. Biegler, *Nonlinear Programming*. Society for Industrial and Applied Mathematics, Jan. 2010, doi: 10.1137/1.9780898719383

[21] ——, "An overview of simultaneous strategies for dynamic optimization," *Chemical Engineering and Processing: Process Intensification*, vol. 46, no. 11, pp. 1043–1053, Nov. 2007, doi: 10.1016/j.cep.2006.06.021

[22] D. R. Herber and A. K. Sundarrajan, "On the uses of linear-quadratic methods in solving nonlinear dynamic optimization problems with direct transcription," in *International Mechanical Engineering Congress & Exposition*, no. IMECE2020-23885, Nov. 2020, doi: 10.1115/IMECE2020-23885

[23] D. R. Herber and J. T. Allison, "Nested and simultaneous solution strategies for general combined plant and control design problems," *Journal of Mechanical Design*, vol. 141, no. 1, p. 011402, Jan. 2019, doi: 10.1115/1.4040705

[24] H. K. Fathy, J. A. Reyer, P. Y. Papalambros, and A. G. Ulsoy, "On the coupling between the plant and controller optimization problems," in *American Control Conference*, 2001, doi: 10.1109/acc.2001.946008

[25] S. S. Rao, "Combined structural and control optimization of flexible structures," *Engineering Optimization*, vol. 13, no. 1, pp. 1–16, Jan. 1988, doi: 10.1080/03052158808940943

[26] H. K. Fathy, P. Y. Papalambros, A. G. Ulsoy, and D. Hrovat, "Nested plant/controller optimization with application to combined passive/active automotive suspensions," in *American Control Conference*, 2003, doi: 10.1109/acc.2003.1244053

[27] J. A. Reyer, H. K. Fathy, P. Y. Papalambros, and A. G. Ulsoy, "Comparison of combined embodiment design and control optimization strategies using optimality conditions," in *Design Engineering Technical Conference*, no. DETC2001/DAC-21119, 2001.

[28] M. Garcia-Sanz, "Control co-design: an engineering game changer," *Advanced Control for Applications: Engineering and Industrial Systems*, vol. 1, no. 1, p. e18, Oct. 2019, doi: 10.1002/adc2.18

[29] A. P. Deshmukh, D. R. Herber, and J. T. Allison, "Bridging the gap between open-loop and closed-loop control in co-design: A framework for complete optimal plant and control architecture design," in *American Control Conference*, Chicago, IL, USA, Jul. 2015, pp. 4916–4922, doi: 10.1109/ACC.2015.7172104

[30] J. A. Reyer and P. Y. Papalambros, "Combined optimal design and control with application to an electric DC motor," *Journal of Mechanical Design*, vol. 124, no. 2, pp. 183–191, May 2002, doi: 10.1115/1.1460904

[31] A. P. Deshmukh and J. T. Allison, "Multidisciplinary dynamic optimization of horizontal axis wind turbine design," *Structural and Multidisciplinary Optimization*, vol. 53, no. 1, pp. 15–27, Aug. 2015, doi: 10.1007/s00158-015-1308-y

[32] ——, "Design of dynamic systems using surrogate models of derivative functions," *Journal of Mechanical Design*, vol. 139, no. 10, Aug. 2017, doi: 10.1115/1.4037407

[33] T. Cui, J. T. Allison, and P. Wang, "Reliability-based co-design of state-constrained stochastic dynamical systems," in *Scitech 2020 Forum*, Jan. 2020, doi: 10.2514/6.2020-0413

[34] D. R. Herber and J. T. Allison, "A problem class with combined architecture, plant, and control design applied to vehicle suspensions," *Journal of Mechanical Design*, vol. 141, no. 10, May 2019, doi: 10.1115/1.4043312

[35] J. M. Hegseth, E. E. Bachynski, and J. R. R. A. Martins, "Design optimization of spar floating wind turbines considering different control strategies," *Journal of Physics: Conference Series*, vol. 1669, p. 012010, Oct. 2020, doi: 10.1088/1742-6596/1669/1/012010

[36] J. Jonkman, A. Wright, G. Barter, M. Hall, J. T. Allison, and D. R. Herber, "Functional requirements for the WEIS toolset to enable controls co-design of floating offshore wind turbines," in *International Offshore Wind Technical Conference*, no. IOWTC2020-3533, Boston, MA, USA, Oct. 2020, url: https://www.osti.gov/biblio/1677426.

[37] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. Society for Industrial and Applied Mathematics, Jan. 2010, doi: 10.1137/1.9780898718577

[38] R. Foust, S.-J. Chung, and F. Y. Hadaegh, "Solving optimal control with nonlinear dynamics using sequential convex programming," in *AIAA Scitech Forum*, Jan. 2019, doi: 10.2514/6.2019-0652

[39] G. van Straten, G. van Willigenburg, E. van Henten, and R. van Ooteghem, *Optimal Control of Greenhouse Cultivation*. CRC Press, Nov. 2011, doi: 10.1201/b10321

[40] A. P. Deshmukh and J. T. Allison, "Multidisciplinary dynamic optimization of horizontal axis wind turbine design," *Structural and Multidisciplinary Optimization*, vol. 53, no. 1, pp. 15–27, Aug. 2015, doi: 10.1007/s00158-015-1308-y

[41] P. Chen and S. M. N. Islam, *Optimal Control Models in Finance*. Kluwer Academic Publishers, 2005, doi: 10.1007/b101888

[42] J. T. Betts, "A collection of optimal control test problems," Applied Mathematical Analysis, LLC, Tech. Rep., 2015.

[43] A. E. Bryson, Jr. and Y.-C. Ho, *Applied Optimal Control*. Taylor & Francis, 1975.

[44] E. S. Lee, *Quasilinearization and Invariant Imbedding: With Applications to Chemical Engineering and Adaptive Control*. Academic Press, 1968.

[45] A. V. Rao, "A survey of numerical methods for optimal control," *Advances in the Astronautical Sciences*, vol. 135, no. 1, pp. 497–528, 2010, url: https://www.anilvrao.com/Publications/ConferencePublications/trajectorySurveyAAS.pdf.

[46] A. Sideris and J. E. Bobrow, "An efficient sequential linear quadratic algorithm for solving nonlinear optimal control problems," in *American Control Conference*, 2005, doi: 10.1109/ACC.2005.1470308

[47] L. Pontryagin, *Mathematical Theory of Optimal Processes*. Routledge, May 2018.

[48] T. Guo, "Design of genetic regulatory networks," M.S. Thesis, University of Illinois at Urbana-Champaign, Urbana, IL, USA, May 2014. [Online]. Available: http://hdl.handle.net/2142/49667

[49] M. Kelly, "An introduction to trajectory optimization: How to do your own direct collocation," *SIAM Review*, vol. 59, no. 4, pp. 849–904, Jan. 2017, doi: 10.1137/16m1062569

[50] The MathWorks, "fmincon," [Online], url: https://www.mathworks.com/help/optim/ug/fmincon.html

[51] A. U. Raghunathan and S. D. Cairano, "Optimal step-size selection in alternating direction method of multipliers for convex quadratic programs and model predictive control," in *International Symposium on Mathematical Theory of Networks and Systems*, Jul. 2014, pp. 807–814, url: https://www.merl.com/publications/docs/TR2014-070.pdf.

[52] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. Springer, 2006, doi: 10.1007/978-0-387-40065-5

[53] J. T. Betts and W. P. Huffman, "Path-constrained trajectory optimization using sparse sequential quadratic programming," *Journal of Guidance, Control and Dynamics*, vol. 16, no. 1, pp. 59–68, Jan. 1993, doi: 10.2514/3.11428

[54] "The DTQP project," [Online], url: https://github.com/danielrherber/dt-qp-project

[55] J. Åkesson, "Optimica—an extension of modelica supporting dynamic optimization," in *International Modelica Conference*, 2008.

[56] M. A. Patterson and A. V. Rao, "GPOPS–II: a matlab software for solving multiple-phase optimalcontrol problems using hp-adaptive gaussian quadrature collocation methods and sparse nonlinear programming," *ACM Transactions on Mathematical Software*, vol. 41, no. 1, pp. 1–37, Oct. 2014, doi: 10.1145/2558904

[57] V. M. Becerra, "PSOPT optimal control solver user manual," User Manual Release 4.0.0 build 2019.02.24, 2019.

[58] The Mathworks, "sparse," [Online], url: https://www.mathworks.com/help/matlab/ref/sparse.html

[59] D. R. Herber and J. T. Allison, "Nested and simultaneous solution strategies for general combined plant and control design problems," *Journal of Mechanical Design*, vol. 141, no. 1, Oct. 2018, doi: 10.1115/1.4040705

[60] H. Jaddu and M. Vlach, "Closed form solution of nonlinear-quadratic optimal control problem by state-control parameterization using chebyshev polynomials," *International Journal of Computer Applications*, vol. 91, no. 10, pp. 1–7, Apr. 2014, doi: 10.5120/15914-5281

[61] H. Jaddu, "Direct solution of nonlinear optimal control problems using quasilinearization and Chebyshev polynomials," *Journal of the Franklin Institute*, vol. 339, no. 4-5, pp. 479–498, Jul. 2002, doi: 10.1016/S0016-0032(02)00028-5

[62] M. Li and H. Peng, "Solutions of nonlinear constrained optimal control problems using quasilinearization and variational pseudospectral methods," *ISA Transactions*, vol. 62, pp. 177–192, May 2016, doi: 10.1016/j.isatra.2016.02.007

[63] X. Liu, P. Lu, and B. Pan, "Survey of convex optimization for aerospace applications," *Astrodynamics*, vol. 1, no. 1, pp. 23–40, Sep. 2017, doi: 10.1007/s42064-017-0003-8

[64] M. J. Tenny, S. J. Wright, and J. B. Rawlings, "Nonlinear model predictive control via feasibility-perturbed sequential quadratic programming," *Computational Optimization and Applications*, vol. 28, no. 1, pp. 87–121, Apr. 2004, doi: 10.1023/B:COAP.0000018880.63497.eb

[65] L. T. Biegler, "Solution of dynamic optimization problems by successive quadratic programming and orthogonal collocation," *Computers ans Chemical Engineering*, vol. 8, no. 3-4, pp. 243–247, 1984, doi: 10.1016/0098-1354(84)87012-X

[66] L. Lu, "Separable nonlinear model predictive control via sequential quadratic programming for large-scale systems," *IFAC-PapersOnLine*, vol. 48, no. 23, pp. 495–500, 2015, doi: 10.1016/j.ifacol.2015.11.327

[67] J. F. Bard, "Convex two-level optimization," *Mathematical Programming*, vol. 40, no. 1-3, pp. 15–27, Jan. 1988, doi: 10.1007/BF01580720

[68] T. Liu, S. Azarm, and N. Chopra, "Decentralized multisubsystem co-design optimization using direct collocation and decomposition-based methods," *Journal of Mechanical Design*, vol. 142, no. 9, May 2020, doi: 10.1115/1.4046438

[69] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson, "Optimal parameter selection for the alternating direction method of multipliers (ADMM): quadratic problems," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 644–658, Mar. 2015, doi: 10.1109/TAC.2014.2354892

[70] C. M. Chilan, D. R. Herber, Y. K. Nakka, S.-J. Chung, J. T. Allison, J. B. Aldrich, and O. S. Alvarez-Salazar, "Co-design of strain-actuated solar arrays for spacecraft precision pointing and jitter reduction," *AIAA Journal*, vol. 55, no. 9, pp. 3180–3195, Sep. 2017, doi: 10.2514/1.J055748

[71] R. Bellman and R. Kalaba, *Quasilinearization and Nonlinear Boundary Value Problems*. Elsevier, 1965.

[72] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta Numerica*, vol. 4, pp. 1–51, Jan. 1995, doi: 10.1017/S0962492900002518

[73] R. Quirynen, B. Houska, M. Vallerio, D. Telen, F. Logist, J. V. Impe, and M. Diehl, "Symmetric algorithmic differentiation based exact hessian SQP method and software for economic MPC," in *IEEE Conference on Decision and Control*, Dec. 2014, doi: 10.1109/CDC.2014.7039811

[74] D. R. Herber, "Basic implementation of multiple-interval pseudospectral methods to solve optimal control problems," Engineering System Design Lab, Urbana, IL, USA, Tech. Report UIUC-ESDL-2015-01, Jun. 2015, url: http://hdl.handle.net/2142/77888

[75] D. Augustin and H. Maurer, "Sensitivity analysis and real-time control of a container crane under state constraints," in *Online Optimization of Large Scale Systems*. Springer, 2001, pp. 69–82, doi: 10.1007/978-3-662-04331-8_4

[76] E. B. Canto, J. R. Banga, A. A. Alonso, and V. S. Vassiliadis, "Restricted second order information for the solution of optimal control problems using control vector parameterization," *Journal of Process Control*, vol. 12, no. 2, pp. 243–255, Feb. 2002, doi: 10.1016/S0959-1524(01)00008-7

[77] G. Clarizia, "Co-design optimization of a tethered multi drone system," M.S. Thesis, Politecnico di Milano, Milano, Italy, 2019.

[78] A. K. Sundarrajan and D. R. Herber, "Towards a fair comparison between the nested and simultaneous control co-design methods using an active suspension case study," in *American Control Conference*, New Orleans, LA, USA, May 2021.

[79] S. Azad and M. J. Alexander-Ramos, "A single-loop reliability-based MDSDO formulation for combined design and control optimization of stochastic dynamic systems," *Journal of Mechanical Design*, Jul. 2020, doi: 10.1115/1.4047870

[80] J. R. R. A. Martins and A. B. Lambe, "Multidisciplinary design optimization: a survey of architectures," *AIAA Journal*, vol. 51, no. 9, pp. 2049–2075, Sep. 2013, doi: 10.2514/1.j051895

[81] W. K. Belvin and K. C. Park, "Structural tailoring and feedback control synthesis - an interdisciplinary approach," *Journal of Guidance, Control, and Dynamics*, vol. 13, no. 3, pp. 424–429, May 1990, doi: 10.2514/3.25354

[82] J. Onoda and R. T. Haftka, "An approach to structure/control simultaneous optimization for large flexible spacecraft," *AIAA Journal*, vol. 25, no. 8, pp. 1133–1138, Aug. 1987, doi: 10.2514/3.9754

[83] X. Hu, S. J. Moura, N. Murgovski, B. Egardt, and D. Cao, "Integrated optimization of battery sizing, charging, and power management in plug-in hybrid electric vehicles," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 3, pp. 1036–1043, May 2016, doi: 10.1109/TCST.2015.2476799

[84] J. Dongarra, J. L. Martin, and J. Worlton, "Computer benchmarking: paths and pitfalls," *IEEE Spectrum*, vol. 24, no. 7, pp. 38–43, 1987, doi: 10.1109/MSPEC.1987.6448963

[85] A. K. Sundarrajan, Y. H. Lee, J. T. Allison, and D. R. Herber, "Open-loop control co-design of floating offshore wind turbines using linear parameter-varying models," in *(to appear) ASME 2021 International Design Engineering Technical Conferences*, Virtual, Online, Aug. 2021.

[86] F. Sandner, D. Schlipf, D. Matha, and P. W. Cheng, "Integrated optimization of floating wind turbine systems," in *Ocean Renewable Energy*, Jun. 2014, doi: 10.1115/omae2014-24244

[87] P. A. Fleming, I. Pineda, M. Rossetti, A. D. Wright, and D. Arora, "Evaluating methods for control of an offshore floating turbine," in *Volume 9B: Ocean Renewable Energy*. American Society of Mechanical Engineers, Jun. 2014, doi: 10.1115/omae2014-24107

[88] J. T. Allison, T. Guo, and Z. Han, "Co-design of an active suspension using simultaneous dynamic optimization," *Journal of Mechanical Design*, vol. 136, no. 8, Jun. 2014, doi: 10.1115/1.4027335

[89] A. L. Nash and N. Jain, "Combined plant and control co-design for robust disturbance rejection in thermal-fluid systems," *IEEE Transactions on Control Systems Technology*, 2019, doi: 10.1109/tcst.2019.2931493

[90] D. S. Zalkind, G. K. Ananda, M. Chetan, D. P. Martin, C. J. Bay, K. E. Johnson, E. Loth, D. T. Griffith, M. S. Selig, and L. Y. Pao, "System-level design studies for large rotors," *Wind Energy Science*, vol. 4, no. 4, pp. 595–618, Nov. 2019, doi: 10.5194/wes-4-595-2019

[91] R. G. Coe, G. Bacelli, S. Olson, V. S. Neary, and M. B. R. Topper, "Initial conceptual demonstration of control co-design for WEC optimization," *Journal of Ocean Engineering and Marine Energy*, vol. 6, no. 4, pp. 441–449, Nov. 2020, doi: 10.1007/s40722-020-00181-9

[92] P. G. Brodrick, C. A. Kang, A. R. Brandt, and L. J. Durlofsky, "Optimization of carbon-capture-enabled coal-gas-solar power generation," *Energy*, vol. 79, pp. 149–162, Jan. 2015, doi: 10.1016/j.energy.2014.11.003

[93] S. Gros, "An economic NMPC formulation for wind turbine control," in *IEEE Conference on Decision and Control*, Dec. 2013, doi: 10.1109/cdc.2013.6760013

[94] Y. Kikuchi and T. Ishihara, "Upscaling and levelized cost of energy for offshore wind turbines supported by semi-submersible floating platforms," *Journal of Physics: Conference Series*, vol. 1356, p. 012033, Oct. 2019, doi: 10.1088/1742-6596/1356/1/012033

[95] W. D. Musial, P. C. Beiter, P. Spitsen, J. Nunemaker, and V. Gevorgian, "2018 offshore wind technologies market report," Tech. Rep., Sep. 2019, doi: https://doi.org/10.2172/1572771

[96] B. L. Ennis and D. T. Griffith, "System levelized cost of energy analysis for floating offshore vertical-axis wind turbines." Tech. Rep., Aug. 2018, doi: 10.2172/1466530

[97] M. Johannessen, "Concept study and design of floating offshore wind turbine support structure," Degree Project, KTH Royal Institute of Technology, 2018, url: http://www.diva-portal.org/smash/get/diva2:1285492/FULLTEXT01.pdf

[98] A. L. H. Hopstad, K. Argyriadis, A. Manjock, J. Goldsmith, and K. O. Ronold, "DNV GL standard for floating wind turbines," in *ASME International Offshore Wind Technical Conference*, Nov. 2018, doi: 10.1115/iowtc2018-1035

[99] DNV, "Design of offshore wind turbine structures." Jun. 2013.

[100] L. Y. Pao and K. E. Johnson, "A tutorial on the dynamics and control of wind turbines and wind farms," in *American Control Conference*, 2009, doi: 10.1109/acc.2009.5160195

[101] P. J. Moriarty and S. B. Butterfield, "Wind turbine modeling overview for control engineers," in *American Control Conference*, 2009, doi: 10.1109/acc.2009.5160521

[102] F. Lemmer, "Low-order modeling, controller design and optimization of floating offshore wind turbines," 2018, doi: 10.18419/OPUS-10526

[103] F. Lemmer, W. Yu, B. Luhmann, D. Schlipf, and P. W. Cheng, "Multibody modeling for concept-level floating offshore wind turbine design," *Multibody System Dynamics*, vol. 49, no. 2, pp. 203–236, Feb. 2020, doi: 10.1007/s11044-020-09729-x

[104] E. Smilden, J.-T. H. Horn, A. J. Sørensen, and J. Amdahl, "Reduced order model for control applications in offshore wind turbines," *IFAC-PapersOnLine*, vol. 49, no. 23, pp. 386–393, 2016, doi: 10.1016/j.ifacol.2016.10.435

[105] H. Pereira, A. Cupertino, R. Teodorescu, and S. Silva, "High performance reduced order models for wind turbines with full-scale converters applied on grid interconnection studies," *Energies*, vol. 7, no. 11, pp. 7694–7716, Nov. 2014, doi: 10.3390/en7117694

[106] J. M. Jonkman and B. J. Jonkman, "FAST modularization framework for wind turbine simulation: full-system linearization," *Journal of Physics: Conference Series*, vol. 753, p. 082010, Sep. 2016, doi: 10.1088/1742-6596/753/8/082010

[107] J. Jonkman, A. D. Wright, G. Hayman, and A. N. Robertson, "Full-system linearization for floating offshore wind turbines in OpenFAST: Preprint," Tech. Rep., Dec. 2018, doi: 10.2172/1489323

[108] J. W. van Wingerden, I. Houtzager, F. Felici, and M. Verhaegen, "Closed-loop identification of the time-varying dynamics of variable-speed wind turbines," *International Journal of Robust and Nonlinear Control*, vol. 19, no. 1, pp. 4–21, Jan. 2009, doi: 10.1002/rnc.1320

[109] F. Bianchi, R. Mantz, and C. Christiansen, "Gain scheduling control of variable-speed wind energy conversion systems using quasi-LPV models," *Control Engineering Practice*, vol. 13, no. 2, pp. 247–255, Feb. 2005, doi: 10.1016/j.conengprac.2004.03.006

[110] F. Lescher, Y. Zhao, and A. Martinez, "Multiobjective $H_2/H_\infty$ control of a pitch regulated wind turbine for mechanical load reduction," Jan. 2006, url: http://www.icrepq.com/icrepq06/248-LESCHER.pdf

[111] "WEIS," [Online, version 52d0b88], url: https://github.com/WISDEM/WEIS

[112] "OpenFAST," [Online], url: https://github.com/OpenFAST/openfast

[113] "IEA-15-240-RWT," [Online, version 9aa6ce4], url: https://github.com/IEAWindTask37/IEA-15-240-RWT

[114] C. Allen, A. Viscelli, H. Dagher, A. Goupee, E. Gaertner, N. Abbas, M. Hall, and G. Barter, "Definition of the UMaine VolturnUS-S reference platform developed for the IEA wind 15-megawatt offshore reference wind turbine," Tech. Rep., Jul. 2020, doi: 10.2172/1660012

[115] G. S. Bir, "Users guide to mbc3: multi-blade coordinate transformation code for 3-bladed wind turbine," Tech. Rep., Sep. 2010, doi: 10.2172/989416

[116] "WISDEM," [Online, version 52d0b88], url: https://github.com/WISDEM/WISDEM

[117] L. Fingersh, M. Hand, and A. Laxson, "Wind turbine design cost and scaling model," Tech. Rep., Dec. 2006, doi: 10.2172/897434

[118] D. J. Malcolm and A. C. Hansen, "WindPACT turbine rotor design study: June 2000–June 2002 (revised)," Tech. Rep., Apr. 2006, doi: 10.2172/15000964

[119] B. Maples, M. Hand, and W. Musial, "Comparative assessment of direct drive high temperature superconducting generators in multi-megawatt class wind turbines," Tech. Rep., Oct. 2010, doi: 10.2172/991560