Nonlinear Dynamic Analysis and Control of Chemical Processes Using Dynamic Operability

San Quan Dinh
West Virginia University, sqd0001@mix.wvu.edu

Follow this and additional works at: https://researchrepository.wvu.edu/etd

Part of the Process Control and Systems Commons

Recommended Citation
https://researchrepository.wvu.edu/etd/11792

This Dissertation is protected by copyright and/or related rights. It has been brought to you by the The Research Repository @ WVU with permission from the rights-holder(s). You are free to use this Dissertation in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you must obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/or on the work itself. This Dissertation has been accepted for inclusion in WVU Graduate Theses, Dissertations, and Problem Reports collection by an authorized administrator of The Research Repository @ WVU. For more information, please contact researchrepository@mail.wvu.edu.
Nonlinear Dynamic Analysis and Control of Chemical Processes Using Dynamic Operability

San Dinh

Dissertation submitted
to the Benjamin M. Statler College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements for the degree of Doctor of Philosophy in
Chemical Engineering

Fernando V. Lima, Ph.D., Chair
Debangsu Bhattacharyya, Ph.D.
Stephen E. Zitney, Ph.D.
Robert Mnatsakanov, Ph.D.
Heleno Bispo, Ph.D.
Juan Salazar, Ph.D.
Department of Chemical and Biomedical Engineering

Morgantown, West Virginia
2023

Keywords: nonlinear dynamics, operability analysis, model predictive control, dynamic discrepancy, reduced-order modeling

Copyright 2023 San Dinh
Abstract

Nonlinear Dynamic Analysis and Control of Chemical Processes Using Dynamic Operability

San Dinh

Nonlinear dynamic analysis serves an increasingly important role in process systems engineering research. Understanding the nonlinear dynamics from the mathematical model of a process helps to find the boundaries of all achievable process conditions and identify the system instabilities. The information on such boundaries is beneficial for optimizing the design and formulating a control structure. However, a systematic approach to analyzing nonlinear dynamics of chemical processes considering such boundaries in a quantifiable and adaptable way is yet to exist in the literature. The primary aim of this work is to formulate theoretical concepts for dynamic operability, as well as develop the practical implementation methods for the analysis of dynamic performance in chemical processes.

Process operability is a powerful tool for analyzing the relationships between the input variables, the output variables, and the disturbances via the geometric computation of variable sets. The operability sets are described by unions of polyhedra, which can be translated to sets of inequality constraints, so the results of the operability analysis can be used for process optimization and advanced process control. Nonetheless, existing process operability approaches in the literature are currently limited for steady-state processes and a generalized definition of dynamic operability that retains the core principles of steady-state operability as a controllability measure. A unified dynamic operability concept is proposed in this dissertation with two different adaptations to represent the complex relationships between the design, control structure, and control law of a given process.

The existing operability mapping methods discretize the input space by partitioning the ranges of each input variable evenly, and all possible input combinations are simulated to achieve the output sets. The procedure is repeated for each value in the expected disturbance set to find the output regions that are guaranteed to be achieved regardless of the disturbance scenario. However, for dynamic systems, the same set of manipulated inputs can take different values at different time intervals, so the number of possible input combinations, which is also the number of simulations required, increases exponentially with the number of time intervals. This tractability challenge motivates the development of novel dynamic operability mapping approaches.

A linear time-invariant dynamic system is first considered to tackle the dynamic mapping of achievable output sets. For a linear system, the achievable output set (AOS) at a fixed predicted time is the smallest convex hull that contains all the images of the extreme points of the available input set (AIS) when propagated through the dynamic model. Given a collection of AOS’s at all predicted times, referred to as the achievable funnel, a set of output constraints is infeasible if its intersection with the achievable funnel is empty. Under the influence of a stochastic disturbance, the achievable funnel is shifted according to the definition of the expected disturbance set (EDS). If the EDS is bounded, the intersection of all achievable funnels at each disturbance realization is the tightest
set of transient output constraints that is operable. Additionally, given a fixed setpoint, an AOS is referred to as a feasible AOS if a series of inputs from the AIS always brings any output to the setpoint regardless of the realization of the disturbance within the EDS. Thus, novel developed theories and algorithms to update the dynamic operability mapping according to the current state variables and the disturbance propagations are proposed to reduce the online computational time of the constraint calculation task.

Dynamic operability mapping for nonlinear processes is an expansion of the above linear mapping. A novel state-space projection mapping is proposed by taking advantage of the discrete-time state-space structure of the dynamic model to reduce the number of input mapping combinations. This method augments the AIS at the current step to include the AOS of the state variables from the previous time step. The nonlinear dynamic operability mapping framework consists of three components: the AOS inspector, the AIS divider, and the merger of the AOS from the previous time with the AIS. Specifically, the AOS inspector evaluates if the current input-output combinations are approximately accurate to the real AOS when all input combinations are mapped to the output space. If the AOS inspector gauges that the current AOS is not sufficiently precise, the AIS divider systematically generates more input-output combinations based on the current AOS. This feedback process is repeated until an accuracy tolerance is reached.

Finally, a novel grey-box model identification algorithm for process control is developed by integrating dynamic operability mapping and Bayesian calibration. The proposed dynamic discrepancy reduced-order model-based approach calibrates the rates of changes of the grey-box model to match the plant instead of compensating for the time-varying output differences. The model reduction framework is divided into three steps: formulating the dynamic discrepancy terms, calibrating the hyperparameters, and selecting the least complex model that is neither underfitted nor overfitted. To demonstrate the effectiveness of the reduced-order model, the developed approach is implemented into a model predictive controller for a high-fidelity model as the simulated plant.
Acknowledgement

I would like to express my deepest gratitude to my supervisor, Dr. Fernando V. Lima, for his invaluable guidance, unwavering support, and encouragement throughout my Ph.D. journey. His patience, expertise, and insightful feedback have been instrumental in shaping my research and refining my writing skills. More importantly, Dr. Lima gave me the trust and freedom to pursue the directions that I was truly interested in, especially in the beginning when all that I had to show to him was limited to mere intuition. That intuition grew into ideas, and those ideas laid the foundation for the frameworks in this dissertation. In that regard, I could not have asked for a better advisor.

I would also like to thank the members of my dissertation committee, Dr. Debangsu Bhattacharyya, Dr. Stephen E. Zitney, Dr. Robert Mnatsakanov, Dr. Heleno Bispo, and Dr. Juan Salazar, for their constructive feedback and insightful suggestions that have helped me improve my research and enrich my understanding of the field.

I am grateful to the staff and faculty of the Department of Chemical and Biomedical Engineering at West Virginia University, who have created a stimulating and supportive academic environment that has enabled me to flourish and grow as a researcher. Especially, I would like to thank Linda Rogers and Monica Cebulak for their unimaginable help with non-research-related tasks.

I would like to thank the members of my CODES research group for being more than just colleagues and some of them I consider to be my friends: Dr. Xin He, Dr. Shuyun Li, Dr. Gaurav Mirlekar, Dr. Vitor Gazzaneo, Dr. Rebecca Kim, Dr. Thiago Mesquita, Dr. Brent Bishop, and the future doctors Selorme Agbleze, Daniel Kestering, Ronald Alexander, Hunter Barber, Victor Alves, Vitor Gama, Claudemi Nascimento, Savannah Sakhai, Beatriz Dantas, Michael Fouts, Antonio Mascaro, and Dean Sweeney.
I owe a debt of gratitude to my friends who have provided me with emotional support, and encouragement during the highs and lows of my Ph.D. candidate years: Dr. Nguyen Quang Minh, Hieu Nguyen, Nguyen Hoang Phuong Nguyen, Sung Min Choi Hong, William Gordon, and Ben Martin. Additionally, I would like to thank my furry friend, Be Luom, for begging for cat treats at some of my busiest times and lifting my spirit whenever I come home.

For giving me high hope when I doubted myself and giving me grace under pressure, I would like to thank Queen, Led Zeppelin, Eagles, Van Halen, Metallica, Lynyrd Skynyrd, Bon Jovi, Rush, Green Day, Sammy Hagar, Serj Tankian, Daron Malakian, Shavo Odadjian, John Dolmayan, and many more artists.

I would like to express my sincere appreciation to my parents and my sister, Dr. Ha Dinh, for their constant support and encouragement throughout my doctoral journey. Their unwavering love, motivation, and sacrifices have been crucial in helping me achieve this significant milestone in my academic career. Their belief in me, their advice, and their ability to uplift me during challenging times have been invaluable. I am truly grateful for the role they have played in my success. Without their support, I would not have been able to overcome the obstacles and challenges that come with pursuing a Ph.D.

Finally, I would like to thank my best friend, my loving partner, and my strange attractor: Doctor-to-be Nhat Anh Thieu. Your perseverance inspired me, and your diligence drove me to be a better person. All the little things you did for me gave my life balance, and I am truly happy to share these amazing years in graduate school with you.

Thank you all for your support. I could not have done this without you.
### List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>A three-step operability framework for design and control of modular units</td>
<td>20</td>
</tr>
<tr>
<td>3.2</td>
<td>Steady-state operability analysis schematic to find the feasible design region</td>
<td>24</td>
</tr>
<tr>
<td>3.3</td>
<td>Procedure of finding the time-invariant $AOS^\infty(x_0)$</td>
<td>27</td>
</tr>
<tr>
<td>3.4</td>
<td>Schematic with inputs, outputs, and reactions of the DMA-MR</td>
<td>29</td>
</tr>
<tr>
<td>3.5</td>
<td>Steady-state operability analysis to find the feasible design region of the</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>DMA-MR</td>
<td></td>
</tr>
<tr>
<td>3.6</td>
<td>Feasibility of fixed DMA-MR designs at different natural gas wells</td>
<td>35</td>
</tr>
<tr>
<td>3.7</td>
<td>Initialization of dynamic operability analysis from the results of the steady-</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>state operability analysis</td>
<td></td>
</tr>
<tr>
<td>3.8</td>
<td>Dynamic achievable output sets for a fixed value of the disturbance</td>
<td>37</td>
</tr>
<tr>
<td>3.9</td>
<td>Dynamic operability analysis for two disturbance sequences</td>
<td>38</td>
</tr>
<tr>
<td>3.10</td>
<td>Effects of $EDS^k$ on the dynamic operability via the intersections of $AOS_u^\infty(d)$</td>
<td>39</td>
</tr>
<tr>
<td>3.11</td>
<td>$AOS_G(k)$ of a DMA-MR with closed-loop NMPC</td>
<td>41</td>
</tr>
<tr>
<td>3.12</td>
<td>Closed-loop performance analysis with dynamic operability of the DMA-MR</td>
<td>42</td>
</tr>
<tr>
<td>4.1</td>
<td>Framework for fast online computation of dynamic operability analysis</td>
<td>50</td>
</tr>
<tr>
<td>4.2</td>
<td>Offline mapping of the nominal dynamic funnel space with state-space projec-</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>tion procedure</td>
<td></td>
</tr>
<tr>
<td>4.3</td>
<td>Simplified schematic of the HYPER process</td>
<td>61</td>
</tr>
<tr>
<td>4.4</td>
<td>Offline constructed funnel of the HYPER process</td>
<td>63</td>
</tr>
<tr>
<td>4.5</td>
<td>Online calculation of the updated mean state funnel</td>
<td>65</td>
</tr>
<tr>
<td>4.6</td>
<td>Online update of the HYPER process operability funnel</td>
<td>66</td>
</tr>
<tr>
<td>4.7</td>
<td>Effects of disturbance ranges on the achievable output funnel</td>
<td>67</td>
</tr>
</tbody>
</table>
List of Tables

3.1 Summary of operability set notations ........................................ 14
3.2 Comparison between different dynamic operability mappings ............. 29
3.3 Steady-state operability sets of the DMA-MR: AIS, DOS, and EDS ........ 34
4.1 Variables of the HYPER process’s control structure .......................... 62
3.4.1 Dynamic modeling of the DMA-MR ............................. 30
3.4.2 Feasible design region of the DMA-MR .................. 33
3.4.3 Operable design region of the DMA-MR .................. 35
3.4.4 Closed-loop control analysis of the DMA-MR .......... 40

4 Linear Operability for Time-Invariant System Analysis 43
  4.1 Introduction ................................................... 43
  4.2 Background .................................................. 44
    4.2.1 Preliminaries .......................................... 44
    4.2.2 Dynamic operability sets ............................ 46
  4.3 Dynamic Operability Mapping Framework .................. 48
    4.3.1 Overview of the framework ......................... 48
    4.3.2 Offline mapping of the nominal dynamic funnel .... 49
    4.3.3 Online update of the state funnel ................ 53
    4.3.4 Online update of the output dynamic funnel ....... 58
  4.4 Case study: HYPER process ................................. 60

5 Nonlinear Operability for Dynamic System Analysis 68
  5.1 Introduction ................................................ 68
  5.2 Nonlinear Feedback Operability Mapping .................. 70
    5.2.1 Polyhedra representation of operability sets ..... 70
    5.2.2 Feedback mapping ................................. 73
  5.3 Nonlinear State-Space Projection for Dynamic Operability Mapping .......... 76
    5.3.1 Generalized state-space projection for discrete-time dynamic systems 76
    5.3.2 Case study: Jacketed CSTR ....................... 78
  5.4 Multiple Steady-State Identification ....................... 81
    5.4.1 Limitation of current multiple steady-state identification methods ... 81
    5.4.2 Divergent dynamic jump method for multiple steady-state identifi-
        cations .............................................. 84
6 Dynamic Discrepancy Reduced Order Modeling

6.1 Introduction ............................................. 90

6.2 Background ............................................. 94
   6.2.1 System preliminaries .................................. 94
   6.2.2 Nonlinear model predictive control formulation ............ 96

6.3 Dynamic Discrepancy Model Reduction Framework for Advanced Process
   Control .................................................. 98
   6.3.1 Dynamic discrepancy ................................... 98
   6.3.2 Data collection and choice of basis functions ............. 102

6.4 Calibrating the Dynamic Discrepancy Terms of the Reduced-Order Model 109
   6.4.1 Bayesian model selection ............................... 109
   6.4.2 Gibbs sampling procedure .............................. 115

6.5 Case Study: Fischer-Tropsch Synthesis Slurry Bubble Column Reactor 118
   6.5.1 High-fidelity dynamic model development .................. 118
   6.5.2 DD-ROM calibration and closed-loop simulation results .... 123

7 Conclusions .............................................. 127

8 Future Recommendations .................................. 130
   8.1 Computational Methods for Operability ....................... 130
   8.2 Dynamic Discrepancy Expansion .......................... 131
   8.3 Control Applications of Dynamic Operability ............... 132
Chapter 1

Introduction

Mathematical models are powerful tools that have found extensive use in the field of process systems engineering, as they provide a systematic approach to the representation of complex systems. In recent years, there has been a notable drive towards the advancement of model development, particularly in the domains of first-principles and machine learning models, across various disciplines within the process engineering field. In parallel, modeling platforms and nonlinear programming solvers are becoming more efficient for faster simulation, process optimization, and control. Nonetheless, an analysis tool for dynamic systems using their corresponding dynamic models has not been comprehensively formulated. Understanding the nonlinear dynamics of the system will enable the determination of all possible processing conditions and the identification of system instabilities. Therefore, the main objective of this dissertation is to propose an insightful analysis tool for the nonlinear dynamics of chemical processes.

In recent years, operability analysis has emerged as a powerful tool to simultaneously examine the complex relationships between the design and operations of chemical processes[1]. Operability quantifies the relationships between input variables, output variables, and disturbances using geometric computations of variable sets. In contrast to process optimization, operability analysis aims to determine the boundaries of a given process in terms of its guaranteed achievable operating conditions, rather than seeking to identify the optimal performance of the process. To further advance the operability concepts, this work aims to expand the scope of process operability to comprehensively
include transient dynamics. This is an essential development since many chemical systems exhibit highly nonlinear behaviors, and it is critical to understand the boundaries of such behaviors to develop robust and reliable control strategies.

One of the primary objectives, and also a significant challenge when performing operability analysis, is constructing all conceivable mappings linking the input and output spaces. The current operability mapping methods rely on the extensive enumeration of all available input combinations, which is very computationally intensive. This brute-force simulation approach is significantly more challenging for dynamic processes, since the addition of a time domain to the analysis increases the number of inputs to include the length of the predictive horizon. In a state-space formulation of dynamic models, the state variables uniquely define the process conditions and all the corresponding output variables. Motivated by this property, a state-space projection mapping method is first developed for a linear time-invariant dynamic system, and it is later generalized for nonlinear dynamic systems. Additionally, a fast set intersection procedure for achievable output sets with additive ellipsoid disturbance deviations is introduced with the potential to be applied for generalized operability analyses, and a feedback mapping to replace input generations based on a grid discretization is proposed.

In the process of finding a tractable computational approach for dynamic mapping, the necessary conditions for constructing a dynamic discrepancy reduced-order model were investigated and discovered. Dynamic discrepancy is a grey-box modeling technique that compensates for the plant-model mismatch in the rate of change equations instead of the time-varying outputs. The approach provides some advantages in calibrating the hyperparameters and balancing simplicity and accuracy for model predictive control purposes. A different discovery while working on the dynamic mapping is a multiple steady-state identification approach by manipulating the stability of the dynamic model. The novel identification method is shown to be more effective than the existing arc-length continuation and numerical deflation for chemical processes.

The following specific aims are proposed to obtain the above objectives:

1. Develop the generalized dynamic operability concepts for process design
and control. Dynamic models are typically constructed in the continuous-time or discrete-time domains. Thus, the proposed dynamic operability concepts must transcend the boundaries of discretization, while maintaining the fundamental properties of each operability set. Additionally, the differences between design inputs, which are immutable during the operations, and manipulated inputs, which are time-varying, are considered in the formulation as well as the operability set operations.

2. Develop tractable operability mapping algorithms for dynamic processes. For a linear dynamic system, the operability mapping is performed as a two-step process. The first step utilizes a novel state-space projection technique to efficiently construct dynamic funnels for longer horizons; this computationally expensive step is performed offline before the online operation begins. The second step is updating the online achievable funnel according to the new measurements and disturbances propagations. State-space projection is generalized for nonlinear processes in this step.

3. Relate the theoretical concepts to process systems engineering applications. Dynamic operability is an effective tool for simultaneous consideration of the design and operations of modular and intensified processes. From the nonlinear dynamic behavior quantification, a novel framework for dynamic model reduction is proposed specifically for process control applications. The key to multiple-steady identification is demonstrated by manipulating the transient dynamics of chemical processes.

4. Apply the methodology to nonlinear chemical processes. The developed approaches are applied to a simplified nonisothermal CSTR, a membrane reactor for direct methane aromatization, a hybrid power generation system, and a Fischer-Tropsch bubble column slurry reactor.
1.1 Research Products

1.1.1 Publications

The research conducted in this dissertation has yielded the following peer-reviewed publications:


1.1.2 Presentations

The following selected oral presentations resulted from the research in this dissertation.


1.1.3 Software development

In addition to the above publications, the research in this dissertation has contributed to the development of the Python Process Operability code package. At the time of writing this dissertation, the repository for this code package is yet to be released to the public, but the future link will be provided in the CODES research website (fernandolima.faculty.wvu.edu).
Chapter 2

Literature Review

In the last decades, process operability has emerged as an approach to quantify the operating region of a certain design and the feasibility of reaching all setpoints within this region[2]. Process operability concepts appear in many areas of process systems engineering, such as process resiliency and optimization-based flexibility, and each area has a unique definition of an operable process. Process resiliency research defines operability as a dynamic performance index, which is formulated as a mean squared error associated with a closed-loop control system[3]. Thus, resiliency-based operability is dependent on the controller’s formulation and does not reveal the dynamic capability of a process design. In flexibility research, operability is defined as the ability to satisfy inequality process constraints during operation[4], and it is generalized as a combination of flexibility and risk assessment. In the context of simultaneous design and control, the input-output geometric operability concept[2] is the most suitable for finding feasible designs and operations of modular systems since it provides a meaningful quantification involving the inputs, outputs, and disturbances. Although the flexibility-based operability and the geometric operability concepts share the same objective of analyzing the ability to achieve desired output specifications under the presence of disturbances, flexibility-based operability uses a max-min-max constrained optimization to find the best performance under the worst-case scenario of the disturbances, while the geometric operability explores all possible scenarios of a process at every value of the disturbances. Furthermore, since the geometric operability concept is an extension of the controllability concept from modern
control theory, operability in this case, and as used in this dissertation, is an inherent characteristic of a dynamic system and thus independent of the control law. Additionally, the operability analysis formulated here can be adapted to include other advanced process control notions, such as null controllable regions[5], [6] and positive invariant sets[7], so that the proposed operability algorithms provide researchers a convenient tool to calculate, visualize and study such novel sets.

A brief summary of the benchmark contributions in operability research along with a comprehensive overview of past efforts is provided in references[8]. Additionally, high-dimensional steady-state operability approaches were demonstrated to be effective in the case study of the Tennessee Eastman process[9], [10], but the concept was not generalized for different design and control analyses. In the area of process modularization, a parallelized nonlinear programming problem (NLP) was formulated to find the optimal points for intensification of energy systems within the feasible design region given by the operability analysis[11]. However, such NLP approach was shown to be computationally expensive, and a linearized multi-model approximation was proposed, in which a mixed-integer linear programming problem (MILP) is formulated to replace the NLP[12]. The MILP-based algorithm demonstrated significantly reduced computational time, while still being able to find a solution that was an asymptote to the NLP solution when applied to a multi-layer design and control framework. However, this MILP-based framework did not consider the process disturbances for the design analysis, and the process dynamics for the control analysis. Dynamic operability was previously defined considering the minimal time to reach a new steady-state as a measure[13]. This concept was later extended to construct an output funnel for feasible output transient constraints of advanced controllers[14] and time-varying output and disturbance sets for a batch process[15]. While the existing works on dynamic operability all provide meaningful results for their respective applications, a generalized definition of dynamic operability that retains the original motivation as a controllability measure[16] is yet to be introduced in the literature. This literature gap is addressed in Chapter 3.

Since dynamic operability analysis reflects the relationships between the inputs, out-
puts, and disturbances of a given process, different operability mappings may have to be
considered to perform case studies. A similar concept to operability is the design of exper-
iment methodologies, which aim to describe and explain the variation of information (or
inputs) under conditions hypothesized to reflect the variation (or process outputs). In the
past, the design of experiments has been applied to simplify the operability mapping[9],
but a generalization for the approach is yet to exist. The discretization of the input space
in operability mapping resembles the full factorial experimental designs[17]. Alternatively,
pseudo-random sampling methods, such as Latin hypercube sampling, have been utilized
to approximate the operability’s available input sets[18]. In addition to the developments
in approximating the input set, the mappings between the input and output spaces have
been accelerated using homotopy continuation with implicit function theorem and sur-
rogate models[18]. Nevertheless, there is still a lack of developed operability mapping
methods for dynamic processes. In Chapter 4 of this work, a computational approach for
dynamic operability mapping is first proposed for linear dynamic systems. The approach
is generalized for nonlinear dynamic systems in Chapter 5.

Input and output multiplicity is a significant challenge for process operability analy-
sis. While this unique challenge was recognized in the early stages of operability develop-
ment[19], the current solution relies on black-box automatic bifurcation analysis software,
such as AUTO[20]. These programs employ solution branch following techniques, such
as arc-length continuation, and branch switching techniques, such as null-space projec-
tion[21]. Although the bifurcation-based methods are standardized and effective, they
are based on updating a new solution from an existing solution in its neighborhood, so
the computational efforts can be demanding. In the field of process systems engineering,
multiplicity behaviors coincide with multiple steady-states, but the current identifica-
tion methods require an in-depth understanding of the physical process. Thus, multiple
steady-state identifications are performed on a case-by-case basis, such as for the study
of the instability based on control structures of a distillation column[22]. In Chapter 5,
a novel multiple steady-state identification method is introduced, which is motivated by
the proposed nonlinear dynamic operability mapping method.
The identification and analysis of transient dynamic phenomena through dynamic operability analysis can provide valuable insights for enhancing the effectiveness of dynamic model reduction. Model reduction is particularly important for advanced controllers such as model predictive control since the model complexity and accuracy affect the closed-loop performances[23]. Model reduction methodologies are categorized into two distinct groups in the literature: the mathematical-based approaches and the heuristic-based approaches[24]. The heuristic-based approaches are based on the physical understanding of the actual process, and they have been successfully applied in conjunction with machine learning techniques for model reduction purposes[25]–[27]. Mathematical reduction approaches rely on their inherent structural characteristics; and therefore do not require a priori knowledge of the underlying plant process. Some standard mathematical model reduction methods are the projection-based methods[28], lumping methods[29], and transformation methods[30]. During the model reduction, some lost characteristics of the dynamic model can be recovered by introducing discrepancy functions[31]. In Chapter 6, a novel dynamic discrepancy formulation framework is introduced based on the state-space projection for model predictive control applications.
Chapter 3

Dynamic Operability Concepts for Design and Control

3.1 Introduction

Past dynamic operability studies extended the steady-state input-output mapping to a dynamic mapping to evaluate the operability index during process operations and the transient time to reach a new steady-state[2]. However, the formal definitions of dynamic operability have not been generalized for chemical processes with different analysis objectives. To fill this gap, the current chapter aims to propose the concept of dynamic funnels as the counterparts of the operability sets in dynamic operability. The concepts are applied to improve modular process design and evaluate different closed-loop control operations.

Process modularization is an emerging process design framework that provides an alternative to the typical stick-built construction approach for chemical processes. In this framework, individual modular units are a series of standardized units that are fabricated in manufacturing facilities, and then installed and replaced based on the needs of the production site to construct a modular plant. Since modular units are assembled in a factory, higher quality control is achieved, and worker safety is increased[32]. Additionally, modularization of a process can potentially save on capital cost, deployment cost, and
project timeline[33]–[35]. In recent years, process intensification has been combined with modularization resulting in many significant improvements in process design[36], [37]. Through process intensification, new process designs with favorable scaling characteristics can be created as a means of further expanding modular manufacturing. Because an intensified modular process incorporates different phenomena (e.g., reaction, separation, heat transfer) into a single piece of equipment (e.g., membrane reactor, reactive distillation column), it is typically more efficient than its conventional counterparts.

While modularization and intensification provide the aforementioned advantages, they bring up a unique set of challenges. In traditional chemical plants, the process design is oversized to increase operability under uncertainties, but the design of a modular process aims to be more compact, which may cause the process to reduce its operable regions[38]. Furthermore, modular plants are more difficult to control because of the coupling of phenomena that occurs during process intensification. This coupling of phenomena is the reason for greater efficiency in modular intensified processes, but the trade-off is fewer degrees of freedom for the controller because there are physically fewer manipulable streams and control valves in the intensified process when compared to the stand-alone units[36]. Lastly, the dimensions of a modular unit are constrained by its means of transportation. For instance, when processing natural gas from the Marcellus Shale Formation around West Virginia and Pennsylvania, modular units can be shipped by road, resulting in their dimensions not exceeding those of commercial trucks[12]. To address these challenges, a comprehensive framework for the design and control of modular units under the effects of process disturbances is proposed in this chapter.

### 3.2 Process Operability Concepts

A process is operable if the desired steady-state and dynamic performance requirements can be achieved from the available inputs regardless of the realization of the disturbances[2]. In order to perform the operability analysis, a mathematical model is required, that relates the inputs and the disturbances to the outputs of a given process. When a first-principles-based model is not available, a data-driven model (for example machine
learning-based [18]) can be used for the operability analysis. Because the operability sets in the following sections are defined from the inputs, outputs, and disturbances of any given process, the definitions in these sections still hold regardless of the nature of the employed mathematical model. In this work, a first-principles model is assumed to be available to describe the system behavior, in which the state-space representation is given by the following differential-algebraic system of equations:

\[
\mathcal{M} = \begin{cases}
\dot{x}(t) = F(x(t), u(t), d(t)) \\
y(t) = h(x(t), u(t), d(t)) \\
c_{eq}(\dot{x}(t), x(t), y(t), u(t), d(t)) = 0 \\
c_{ineq}(\dot{x}(t), x(t), y(t), u(t), d(t)) \geq 0
\end{cases}
\] (3.1)

in which at any given time \( t \), \( x(t) \in \mathbb{R}^{n_x} \) is the state vector, \( u(t) \in \mathbb{R}^{n_u} \) is the manipulated/input vector, \( d(t) \in \mathbb{R}^{n_d} \) is the disturbance vector, and \( y(t) \in \mathbb{R}^{n_y} \) is the process output vector. The time derivative vector of the state is denoted as \( \dot{x}(t) \). The dynamics of the process are embedded in the rate of change equations \( F: \mathbb{R}^{n_x+n_u+n_d} \rightarrow \mathbb{R}^{n_x} \). Also, the projections of the state vector onto the output vector corresponding to the nonlinear mapping \( H: \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_y} \). The equality constraints and the inequality constraints are respectively \( c_{eq} \) and \( c_{ineq} \).

In the following sections, operability sets are introduced as means to quantify the relationships between the inputs and outputs of a process. The inputs of a process are often bounded above and below. For example, an irreversible flow input is bounded below by zero and bounded above by its equipment’s limitations. A measure, \( \mu \), is introduced as an effective means to compare two sets in the same space:

\[
\mu(S) = \begin{cases}
-\infty & \text{if } S = \{\emptyset\} \\
Hypervolume(S) & \text{if } S \neq \{\emptyset\}
\end{cases}
\] (3.2)

in which the set \( S \subset \mathbb{R}^{n_x} \) can either be an input set, an output set, or a disturbance set. The measure \( \mu \) of the set \( S \) is an extension of the hypervolume, which is also the Lebesgue measure, in the Euclidean space containing \( S \). If \( S \) is an interval in \( \mathbb{R} \), \( \mu \) measures its
length. If \( S \) is a compact set in \( \mathbb{R}^2 \), \( \mu \) measures its area. If \( S \) is a compact set in \( \mathbb{R}^3 \), \( \mu \) measures its volume. If \( S \) is a compact set in a higher dimension, \( \mu \) measures its hypervolume. When \( \mu(S) \) is zero, \( S \) is not empty, but the set contains an infinitesimal subset of its enclosing space. For instance, the measure of a line segment in a two-dimensional space is 0, but the segment is not an empty set. This zero-measure implication is particularly important in the following sections, where process operability is determined by whether the achievable output set is empty. Also, if the desired input set or the desired output set is empty, a sufficiently large negative value is given to their measure to distinguish it from the nonempty set with zero measure. Thus, the measure of an empty operability set is stipulated as negative infinity. In the computational application of this operability measure, since the non-negative hypervolume condition can be checked with an if statement, this measure can also serve as an “index” for the emptiness of a set.

In the following sections, different operability sets will be defined, and the summary of the different sets that are used in this work and their corresponding notations is provided in table 3.1. In particular, for the set nature "steady-state", only steady-state values are included, and the transient dynamics are not considered. For "dynamic funnel," the set contains all possible values from the initial time 0 until the moment immediately preceding a given time \( t \) or \( k \). The "dynamic snapshot" denotes the set of values at a fixed time, instead of a whole time interval as in the "dynamic funnel".

The numerical computation of the operability sets in the following sections is an adaptation of the multimodel operability framework[12]. In this approach, space discretization techniques are employed to model the input-output mapping relationship as series of linearized models. In practice, the input sets are partitioned into a union of convex polyhedra, and the output sets are the union of the images of these input polyhedra’s projections onto the output space. Input and output polytopes in these models are connected, which simplifies the computation of space intersections and hypervolumes. By using polytopes and barycentric interpolations, the inverse model and other calculations can be performed efficiently.
### Table 3.1: Summary of operability set notations

<table>
<thead>
<tr>
<th>Operability Set</th>
<th>Description</th>
<th>Nature</th>
</tr>
</thead>
<tbody>
<tr>
<td>( AIS )</td>
<td>Available Input Set</td>
<td>Steady-state</td>
</tr>
<tr>
<td>( AIS_{op} )</td>
<td>Available Input Set for Operations</td>
<td>Steady-state</td>
</tr>
<tr>
<td>( AIS_{des} )</td>
<td>Available Input Set for Design</td>
<td>Steady-state</td>
</tr>
<tr>
<td>( AIS^t ) or ( AIS^k )</td>
<td>Dynamic Available Input Set for Operations from time 0 to time ( t ) or time ( k )</td>
<td>Dynamic funnel</td>
</tr>
<tr>
<td>( EDS )</td>
<td>Expected Disturbance Set</td>
<td>Steady-state</td>
</tr>
<tr>
<td>( EDS^t ) or ( EDS^k )</td>
<td>Dynamic Expected Disturbance Set from time 0 to time ( t ) or time ( k )</td>
<td>Dynamic funnel</td>
</tr>
<tr>
<td>( AOS )</td>
<td>Achievable Output Set</td>
<td>Steady-state</td>
</tr>
<tr>
<td>( AOS_u(d, x_0, t) ) or ( AOS_u(d, x_0, k) )</td>
<td>Dynamic Achievable Output Set at time ( t ) or time ( k ) given a disturbance sequence ( d ) and an initial state ( x_0 )</td>
<td>Dynamic snapshot</td>
</tr>
<tr>
<td>( AOS(x_0, t) ) or ( AOS(x_0, k) )</td>
<td>Dynamic Achievable Output Set at time ( t ) or time ( k ) given an initial state ( x_0 ) regardless of the disturbances</td>
<td>Dynamic snapshot</td>
</tr>
<tr>
<td>( AOS_G(x_0, k) )</td>
<td>Dynamic Achievable Output Set at time ( t ) or time ( k ) given a feedback law ( G ) and an initial state ( x_0 )</td>
<td>Dynamic snapshot</td>
</tr>
<tr>
<td>( AOS^\infty(x_0) )</td>
<td>Time-invariant Achievable Output Set given an initial state ( x_0 )</td>
<td>Dynamic snapshot</td>
</tr>
<tr>
<td>( DOS )</td>
<td>Desired Output Set</td>
<td>Steady-state</td>
</tr>
<tr>
<td>( DOS(t) )</td>
<td>Dynamic Desired Output Set at time ( t )</td>
<td>Dynamic snapshot</td>
</tr>
<tr>
<td>( DIS_{des} ) or ( DIS )</td>
<td>Desired Input Set of all feasible designs</td>
<td>Steady-state</td>
</tr>
<tr>
<td>( DIS_{op} )</td>
<td>Conventional Desired Input Set of operational variables</td>
<td>Steady-state</td>
</tr>
</tbody>
</table>

### 3.2.1 Steady-state operability concepts

For a steady-state process, the time index \( t \) in equation (3.1) is removed and the time derivative \( \dot{x}(t) \) is set to 0. The input-output mapping for the steady-state operability analysis is simplified to the following expression:

\[
y = \mathcal{M}(u, d)
\]

(3.3)

The objective of the steady-state operability analysis is to determine whether a desired steady-state performance requirement can be achieved considering the available inputs regardless of the realizations of the disturbances. In the following section, steady-state operability is used to find a modular unit’s feasible design region, which guarantees that
the product specifications are reached in the presence of disturbances or uncertainties. Using the mapping given by equation (3.3), the operability sets that describe the readily accessible information and feasible results of interest are specified below.

*Available Input Set (AIS):* Set of input variables that can be freely selected from a given range provided by the process constraints. In the previous operability work[12], AIS was divided into $AIS_{des}$ and $AIS_{op}$. The $AIS_{des}$ includes the design specifications (e.g., reactor sizes, membrane properties, etc.), and the $AIS_{op}$ includes the operating conditions (e.g., pressure, flow rates, etc.). Mathematically, the AIS is defined as:

$$AIS = \{ u \mid u_{\min} \leq u \leq u_{\max}\}$$  \hspace{1cm} (3.4)

*Desired Output Set (DOS):* Set of output variable targets that are needed for the given system. The DOS may contain the product specifications and the output constraints. Mathematically, the DOS is defined as:

$$DOS = \{ y \mid y_{\min} \leq y \leq y_{\max}\}$$  \hspace{1cm} (3.5)

*Expected Disturbance Set (EDS):* Set of realizations of the disturbances, which are not manipulated inputs. The EDS can represent uncertain parameters (e.g., activation energy, kinetic parameters, etc.) and process disturbances (e.g., ambient temperature, inconsistent feed streams, etc.). In the traditional operability definition, if only the upper and the lower bounds of the disturbances are considered, the EDS is given by:

$$EDS = \{ d \mid d_{\min} \leq d \leq d_{\max}\}$$  \hspace{1cm} (3.6)

In this work, the disturbances, $d \in \mathbb{R}^{n_d}$, are assumed to be in a Gaussian random vector with a mean $\bar{d}$ and a covariance matrix $\Sigma$. Since a Gaussian distribution has unbounded support, the EDS is defined as the region of 99% highest density of the disturbances. If the disturbance is a scalar, the EDS is defined as the interval of six standard deviations that center at the mean. Mathematically, then the EDS is represented by an ellipsoid
with the scale $l^2$ equals to the inverse cumulative distribution function of the chi-squared distribution with $n_d$ degrees of freedom:

$$EDS = \{ d \mid (d - \bar{d})^\top \Sigma^{-1} (d - \bar{d}) \leq l^2; l^2 = \text{Inv}_{\chi^2}(99\%; n_d) \} \quad (3.7)$$

**Achievable Output Set ($AOS_u(d)$):** Set of all reachable output variables given the process and the $AIS$ at a fixed value of the disturbance $d \in EDS$. Mathematically, the $AOS_u(d)$ is defined as:

$$AOS_u(d) = \{ y \mid y = M(u, d), u \in AIS \} \quad (3.8)$$

**Desired Input Set ($DIS_u(d)$):** Set of all input values that map to the $DOS$ when the disturbance is held at a constant value $d \in EDS$. In the traditional operability definition, $DIS_u(d)$ is defined by an inverse mapping of equation (3.1), which does not cover all possible inputs in the presence of input-output multiplicities. A more generalized definition of $DIS_u(d)$ is formulated here, in which its mathematical description is the following:

$$DIS_u(d) = \{ u \mid \exists y \in DOS : y = M(u, d) \} \quad (3.9)$$

### 3.2.2 Dynamic operability concepts

Dynamic operability corresponds to an extension of the aforementioned steady-state operability analysis. Similar to steady-state operability, dynamic operability is the ability to reach the desired dynamic performance requirements given the input ranges and regardless of the process disturbances. While steady-state operability analysis determines the feasibility of a process design, dynamic operability gauges the effectiveness of a control structure during online operations. In other words, dynamic operability assesses whether the manipulated inputs can compensate for the effects of the disturbances during dynamic operations. In this time-varying setting, the definitions of the operability sets are extended as below.

For a dynamic process, the dynamic model given in equation (3.1) is simplified to the
following input-output mapping:

\[ y(t) = \tilde{\mathcal{M}}(x_0, \{u(\tau)\}_{0}^{t}, \{d(\tau)\}_{0}^{t}) \]  

(3.10)

\[ \{u(\tau)\}_{0}^{t} = \{u(\tau) | 0 \leq \tau < t \} \]  

(3.11)

\[ \{d(\tau)\}_{0}^{t} = \{d(\tau) | 0 \leq \tau < t \} \]  

(3.12)

in which \( x_0 \) is the initial value of the state vector \( x(t) \), \( \{u(\tau)\}_{0}^{t} \) is the set of all control actions from the time 0 to time \( t \), and \( \{d(\tau)\}_{0}^{t} \) is the set of all recorded disturbances from the time 0 to time \( t \). Since the output vector, \( y(t) \), results from the initial condition, \( x_0 \), the manipulated input sequence and the disturbance sequence up to time \( t \), the dynamic mapping notation, \( \tilde{\mathcal{M}} \), is introduced here to distinguish from the steady-state mapping notation above, \( \mathcal{M} \).

**Available Input Set at time \( t \) (AIS\(^t\)):** Set of all inputs from the initial time 0 until the moment immediately preceding time \( t \) that can be freely manipulated during the operations of a given process for a given range provided by the process constraints. Dynamic operability examines the ability to change the inputs to compensate for the disturbances, but the design of a plant is not changed after its construction. For this reason, when choosing the inputs for the dynamic operability analysis, design variables (e.g., reactor sizing, materials, etc.) are not considered.

\[ AIS^t = \{ \{u(\tau)\}_{0}^{t} | u_{\text{min}} \leq u(\tau) \leq u_{\text{max}}, \forall t \in [0, t) \} \]  

(3.13)

**Expected Disturbance Set at time \( t \) (EDS\(^t\)):** Set of all possible values of the disturbances from the initial time 0 until the moment immediately preceding time \( t \). In this framework, the process disturbances are assumed to be independent and identically distributed Gaussian random variables. Similarly to the steady-state operability \( EDS \), the \( EDS^t \) is bounded by an ellipsoid region around the highest probability density, and its
mathematical description is the following:

\[
EDS^t = \left\{ \{d(\tau)\}_{0}^{t} \mid (d(\tau) - \bar{d})^\top \Sigma^{-1} (d(\tau) - \bar{d}) \leq l^2 \right\}, \forall t \in [0, t]; l^2 = \text{Inv}_{\chi^2}(99\%; n_d) \tag{3.14}
\]

**Achievable Output Set at time** \(t\) (AOS\(_u\)(d, \(x_0\), \(t\))): Set of all reachable output vectors at time \(t\) from the initial condition \(x_0\) given the AIS\(^t\) and a fixed sequence \(\{d(\tau)\}_{0}^{t} \in EDS^t\) of the disturbances. While the dynamic operability sets in the input and disturbance spaces (AIS\(^t\) and EDS\(^t\)) are defined based on a time horizon, the dynamic operability set in the output space, such as the AOS\((x_0, t)\), is defined as a "snapshot" of a fixed moment in time. This formulation allows the AOS\((x_0, t)\) to have the same output dimensions at different times, which leads to a simpler comparison between different output sets. Mathematically, the AOS\((x_0, t)\) is defined as:

\[
AOS_u(d, x_0, t) = \left\{ y(t) \mid \exists \{u(\tau)\}_{0}^{t} \in AIS^t : y(t) = \mathcal{N}(x_0, \{u(\tau)\}_{0}^{t}, d) \right\} \tag{3.15}
\]

**Desired Output Set at time** \(t\) (DOS\((t)\)): Set of output vector targets at time \(t\). This set corresponds to a collection of specified control intervals for the process outputs (e.g., range of output concentrations, temperatures, etc.). The DOS\((t)\) can be time-varying or time-invariant. A time-varying DOS\((t)\) represents a scenario where the control objectives change with time, such as the outputs of a load-following power plant. Mathematically, a time-varying DOS\((t)\) is a compact set that is defined by a set of time-varying inequalities:

\[
DOS(t) = \left\{ y(t) \mid c_{\text{DOS}}(y(t), t) \leq 0 \right\} \tag{3.16}
\]

In this work, a time-invariant DOS\((t)\) is assumed to be sufficient to represent the output specifications of a modular unit. Thus, the DOS\((t)\) is simplified to a bounded set on the outputs.

\[
DOS(t) = \left\{ y(t) \mid y_{\text{mix}} \leq y(t) \leq y_{\text{max}} \right\} \tag{3.17}
\]

**Desired Input Set at time** \(t\) (DIS\(_u\)(d, \(x_0\))): Set of all input sequences \(\{u(\tau)\}_{0}^{t}\) that bring the output vector from the initial state, \(x_0\), to a vector in the DOS\((t)\) for a fixed sequence,
\{d(\tau)\}_{t_0}^{t} \in EDS^t,\) of the disturbance. Mathematically, the \(DIS^t_{u}(d, x_0)\) is defined as:

\[
DIS^t_{u}(d, x_0) = \left\{ \{u(\tau)\}_{t_0}^{t} \mid \exists y(t) \in DOS(t) : y(t) = \tilde{M}(x_0, \{u(\tau)\}_{t_0}^{t}, d) \right\}
\]

Due to the fact that the above operability sets are defined in continuous time, the operability mappings between the input sets, disturbance sets, and output sets may be intractable as there may exist infinitely many \(AIS^t, AOS_{u}(d, x_0, t)\) in a finite time interval \([0, t]\). To circumvent this issue, the time domain of the continuous model in (3.1) is discretized as follows:

\[
t = \Delta t \times k, \forall k \in \mathbb{N}
\]

Since there is a diverse collection of discretization methods in the literature and the operability concepts are applicable regardless of the chosen discretization, without loss of generality, a zero-order hold is applied to the inputs and the disturbances. The input-output mapping in (3.10) is then obtained by approximating the solutions of the differential algebraic equations in (3.1). Additionally, the discrete-time operability sets are formulated similarly to their respective continuous sets above with a time index \(k\) instead of \(t\) as a superscript.

### 3.3 Proposed Approach

#### 3.3.1 Proposed framework for design and control of modular units

Modular plants consist of a series of modular units that can be independently replaced and upgraded as needed. Each modular unit is a self-contained steel structure (i.e., a module) with integrated process equipment, pipings, instrumentations, and an integrated control system[39]. While modular units may be more efficient due to process integration and intensification, they may have fewer operational degrees of freedom[36]. Furthermore, the manufacturing cost of modules is typically higher than for traditional units due to the additional requirement to withstand the stress of transportation and installation. To
address these challenges, the upfront engineering of a modular unit has to ensure that it can be operated within a wide range of feed conditions while guaranteeing product specifications. Subsequently, mass-producing a sufficient number of modules can reduce their cost to be less than or equal to their conventional counterparts[40].

In this work, an operability-based framework is proposed to analyze the relationship between the design and control of a modular unit. The overall objective is to identify the set of feasible designs for a modular unit that can reach the desired product quality specifications while operating under different conditions. This objective is divided into three aims, in which each aim is discussed in the following subsections. An illustration of the proposed framework is provided in figure 3.1.

Figure 3.1: A three-step operability framework for design and control of modular units

The first aim is to identify all the feasible designs, for which the desired outputs are achievable considering all values of the disturbances. For process design, two sets of considered inputs are the design inputs and the operations inputs. The design inputs are the malleable inputs during the manufacturing of modular units that are not changeable during the operation. The operations inputs are the manipulated inputs during the operation, such as valve positions. Since most processes are operated around some steady-state conditions, the steady-state operability in this aim is necessary to quantify the relation-
ships between the design and control of a modular unit. This aim is further explored in subsection 3.3.2.

The second aim is to find all the operable designs from the feasible designs identified in the first aim. A control structure is defined here as the selection of manipulated input variables and output variables. A control structure of a design is operable if its manipulated inputs are capable of compensating for the effects of the disturbances during the online operation. Since the operation of a modular unit corresponds to a dynamic process, the steady-state operability analysis in the previous step is not sufficient to evaluate the control structure of a process. Thus, using dynamic operability analysis for a fixed disturbance set is employed in this step. This aim is discussed in more details in subsection 3.3.3.

The third aim is to introduce a measure for the closed-loop performance of a modular system with a fixed control law. A control law is defined as the policy of changing the manipulated variables according to the current state of the system to achieve a pre-defined setpoint. Some examples of control laws are classical feedback PID controls, and nonlinear model predictive control (NMPC). In advanced process control such as NMPC, a layer of state estimation is usually paired with the controller to estimate the process states from the measurements. However, the replacement of the actual full state information with estimated state information does not affect the fixed control law dynamic operability analysis performed in this step. Thus, in this work, the states of a dynamic process are assumed to be available as needed, and a suggestion on the modification of this step with a state estimator is provided below. The final aim is developed in subsection 3.3.4.

3.3.2 Steady-state operability analysis for feasible design region

A feasible design is considered to be a design of a modular unit that has some steady-state outputs contained within the DOS given by equation (3.5) for all values of the process disturbances in the EDS. The ranges of the design variables in the AIS are given by the manufacturing capability of the modular unit’s production facility and the shipment requirements of the transportation method, such as container trucks or freight
trains. The DOS is assumed to be a set of output specifications that the targeted modular plant will have to meet. Since identical modular units are potentially shipped to build modular plants that will be operating at different conditions, the EDS represents different manufacturing conditions and external disturbances that can be defined and characterized before the construction of the modular plant. Once the AIS, EDS, DOS, and the steady-state mapping are defined, the $AOS_u(d)$ and $DIS_u(d)$ sets can be obtained for every fixed disturbance value $d$.

In the previous multilayer operability framework[12], the steady-state operability analyses are sequentially performed on the $AIS_{des}$ and the $AIS_{op}$ to find a feasible design region and to rank different designs based on the operations. However, by excluding the operational variables from the $AIS_{des}$, only the nominal outputs instead of not all achievable outputs are covered in the design’s operability analysis. In this work, the steady-state operability analysis is generalized to include the $AIS_{des}$ and optionally the $AIS_{op}$. In the steady-state forward mapping from the input space to the output space, an output vector represents a feasible steady-state condition. Since the considered modular process given by equation (3.1) is dynamic, a feasible steady-state condition exists, but the process may never be able to be constantly maintained at that steady-state condition during transient due to the presence of disturbances. However, if a steady-state condition does not exist (or is unreachable from the given AIS), the outputs of the dynamic process are guaranteed to never approach the neighborhood of the infeasible (or nonexisting) steady-state condition, and the following dynamic operability analyses are not needed. Thus, the purpose of the steady-state operability analysis is not to find the set of all feasible designs, but to eliminate all infeasible designs under the most ideal conditions of disturbances and operational inputs. For this reason, design variables must be included in the steady-state AIS, but the inclusion of operational variables in the AIS is optional.

Since a design is immutable after the manufacturing of a modular unit is finished, the choice of a steady-state achievable output can be freely selected from the $AOS_u(d)$, but it is fixed permanently after being chosen. If operational inputs are included in the steady-state AIS, an achievable output is considered as a steady-state setpoint, which is also
fixed with respect to the established design. Additionally, the $AIS$ and the $AOS_u(d)$ only represent the boundaries of the achievable designs and the achievable steady-state outputs without revealing exactly which design maps to which output. As a result, comparing all different $AOS_u(d)$ against the $DOS$ reveals no information on which design is feasible regardless of the disturbance realization in the $EDS$. However, the inverse mapping of the $DOS$ to different $DIS_u(d)$ in the input space distinguishedly identifies the achievable desired input set ($DIS_{des}$) via the following set intersection:

$$DIS_{des} = AIS \cap \left\{ \bigcap_{d \in EDS} DIS_u(d) \right\}$$  \hspace{1cm} (3.20)

In the traditional operability concept[2], only the $AIS_{op}$ is considered for the steady-state operability analysis. Unlike the input in the $AIS_{des}$, inputs in the $AIS_{op}$ are changeable after a modular unit is manufactured. Thus, the traditional definition of the $DIS$, or $DIS_{op}$ in this work, is an extended input region of the manipulated variables that is needed to reach all the points in the $DOS$. Mathematically, the traditional $DIS_{op}$ is formulated as following:

$$DIS_{op} = \left\{ \bigcup_{d \in EDS} DIS_u(d) \right\}$$  \hspace{1cm} (3.21)

In the current work, the $DIS$ notation is simplified to only denote the $DIS_{des}$ in equation (3.21). As the $DIS$ is the subset of the $AIS$ that always maps to the outputs in the $DOS$ regardless of the value of $d \in EDS$, the $DIS$ by definition is the set of feasible designs for all values of the disturbance. If the feasible design region does not exist, then the $DIS$ is an empty set, and one of the following scenarios can be considered. If the intersection of all $DIS_u(d)$ is nonempty, then the range of the design variables should be increased such that the $AIS$ has some overlap with this intersection. If the intersection of all $DIS_u(d)$ is empty, an alternative modular design should be considered. Mathematically, the feasible design region exists if its $\mu$ measure from equation (3.2) is nonnegative:
\[ \mu(DIS) \geq 0 \]  \hspace{1cm} (3.22)

An example of steady-state operability analysis to find a feasible design region is illustrated in figure 3.2. The \textit{AIS} in the example represents two design variables that cannot be changed after being chosen. Two different designs, \textit{A} and \textit{B}, are selected, and each design is represented by their respective point in figure 3.2a. The \textit{EDS} in this illustrative example is assumed to contain two values \( EDS = \{1, 2\} \), and the achievable outputs of each design are indexed by a subscript of the disturbance value. In figure 3.2b, \textit{A}_1 and \textit{A}_2 are respectively the achievable outputs of the design \textit{A} when the disturbance takes the value of 1 and 2. The achievable outputs \textit{B}_1 and \textit{B}_2 are defined similarly. Since design \textit{B} is contained in the \textit{DIS}, both of its achievable outputs lie in the \textit{DOS} regardless of the value of the disturbance, and design \textit{B} is thus a feasible design. On the contrary, the disturbance shifts the achievable output of design \textit{A} outside the \textit{DOS}, so design \textit{A} is not a feasible design. Let the images of the \textit{AIS} at disturbance values of 1 and 2 respectively be \textit{AOS}_1 and \textit{AOS}_2, the intersection of \textit{AOS}_1, \textit{AOS}_2, and \textit{DOS} is not sufficient to identify the set of all feasible designs. Without an inverse mapping, the achievable outputs \textit{A}_1 and \textit{B}_1 both belong to the aforementioned intersection, but only design \textit{B} is feasible. Thus, the set of all feasible designs are the intersection of the \textit{AIS} and \textit{DIS}_1 and \textit{DIS}_2, which are respectively the inversely mapped image of the \textit{DOS} at disturbance values 1 and 2.

![Figure 3.2](image-url)
3.3.3 Dynamic operability analysis for operable region

The dynamic operability concept is an extension of the controllability concept in modern control theory. A process is controllable if there always exists a manipulated input sequence to reach any arbitrary point in the output space in finite time\[41\]. While controllability is well-defined with consistent mathematical conditions, such as the full-rank condition of a controllability matrix for a linear system, this property is not always applicable to an actual control system with input constraints, output constraints, and process disturbances. To bridge this gap between controllability and the operations of physical systems, dynamic operability is defined as the ability to reach a \(DOS(k)\) in a finite time given a manipulated input sequence from \(AIS^k\) considering the effects of the disturbances. In other words, a dynamic process is fully operable if there is a manipulated input sequence that satisfies the constraints and brings the outputs to desirable values for all scenarios of the disturbances. Similar to controllability, dynamic operability is an inherent property of the system, and it is independent of the formulation of the selected control laws.

The achievable output set \((AOS(x_0, k))\) at a discrete-time \(k\) from an initial condition \(x_0\) of the dynamic process given by equation (3.1) is mathematically defined as:

\[
AOS(x_0, k) = \left\{ \bigcap_{d \in EDS} AOS_u(d, x_0, k) \right\}
\]

Unlike the design inputs for the steady-state \(AIS\), the manipulated inputs can be moved in the \(AIS^k\) at any given time. Each \(AOS_u(d, x_0, k)\) at a fixed \(d\) represents all possible values of the outputs by manipulating the inputs, and the transformations of \(AOS_u(d, x_0, k)\) according to \(d\) correspond to the effect of the disturbances on the outputs. Thus, every output in the \(AOS(x_0, k)\) is achievable if the exact disturbance sequence \(\{d(\tau)\}_{0}^{k}\) is provided at time 0. In practice, \(\{d(\tau)\}_{0}^{k}\) is only available at time \(k\), so the \(AOS(x_0, k)\) represents the best attempt of a controller to reject the predicted disturbances. Furthermore, a process is dynamically operable if its \(AOS(x_0, k)\) always overlaps with the \(DOS(k)\) after a finite time value \(\hat{k}\). Mathematically, the dynamic operability condition is
thus defined as follows:

\[
\exists \hat{k} < \infty : \mu(AOS(x_0, k) \cap DOS(k)) \geq 0, \quad \forall k \geq \hat{k}
\]  

(3.24)

According to the condition in (3.24), dynamic operability analysis can be computationally intractable since the \(AOS(x_0, k)\) at all values of \(k\) are needed. Fortunately, if the process given in equation (3.1) is input-to-state stable[42], the \(AOS_u(x_0, k)\) converges to a time-invariant set \(AOS^\infty(x_0)\) as \(k\) approaches \(\infty\). Since the disturbances can be considered as random inputs of a process, the disturbances can be defined as admissible inputs for the stability analysis. Therefore, for any fixed value of the disturbance and manipulated inputs, the outputs are covered by a compact set. Thus the set intersections in (3.23) and (3.24) are also bounded by compact sets. Additionally, the generation of \(AOS_u(d, x_0, k + 1)\) in (3.23) is equivalent to the union of all one-step mappings with an initial condition drawn from \(AOS_u(d, x_0, k)\). As a result, the measure \(\mu(AOS(x_0, k))\) is always less than or equal to the measure \(\mu(AOS_u(d, x_0, k + 1))\). Because the measure \(\mu(AOS_u(d, x_0, k))\) is increasing with \(k\) and bounded above, it converges to a constant value. Mathematically, the time-invariant set \(AOS^\infty(x_0)\) is reached at time \(k_{\text{max}}\) if the difference in measure of two consecutive \(AOS_u(d, x_0, k)\) is less than a pre-defined threshold \(\varepsilon\):

\[
k_{\text{max}} = \min \{ k \text{ s.t. } |\mu(AOS_u(d, x_0, k)) - \mu(AOS_u(d, x_0, k - 1))| < \varepsilon \} 
\]  

(3.25)

\[
AOS^\infty(x_0) = AOS(x_0, k_{\text{max}})
\]  

(3.26)

The time-invariant set \(AOS^\infty(x_0)\) can be found by generating \(\mu(AOS(x_0, k))\) one at a time until the threshold \(\varepsilon\) on the difference in measure \(\mu\) is met. Since the \(\mu(AOS(x_0, k))\) will always converge to the same \(AOS^\infty(x_0)\) regardless of different initializations \(x_0\), a steady-state solution from the feasible region obtained in equation (3.20) is chosen. An illustration of this procedure is provided in figure 3.3.

The dynamic operability concept in this subsection can be extended to obtain a concept
of stochastic operability. If the $EDS^k$ given by equation (3.14) is limited to the highest density region of probability, $\alpha$, instead of 99\%, the $AOS^\infty(x_0)$ will represent the set of outputs that can be achieved for at least $\alpha$ percent of the disturbance values.

Note that in this, the dynamic operability analysis is dependent on the initial state $x_0$, and the role of the initial state falls into one of the following two scenarios. In the first scenario, the initial state can be manipulated by the operators, and the $AIS^t$ is expanded to include $x_0$ as a manipulated input. An example of this scenario is the start-up of a modular process. In the second scenario, the initial state is a random vector with bounded support, and the $EDS^t$ is expanded to include $x_0$ as an additional process disturbance. An example of this scenario is when dynamic operability mapping is performed along with the online operation, and the states are deviated by the process disturbances. In both scenarios, the dynamic operability can be performed as presented above with the modification of the $AIS^t$ or the $EDS^t$.

3.3.4 Closed-loop performance measure with dynamic operability

The result of the steady-state operability analysis is a feasible design region. From each point of the feasible design region, dynamic operability analysis is performed to find the
operable design region that is a subset of the feasible design region. While this is sufficient for a modular design to meet output specifications at different modular plant conditions, an optional step for measuring the closed-loop control performance for a fixed control law is provided in this subsection. The closed-loop analysis performed here is a special case of the dynamic operability analysis proposed in subsection 3.3.3.

Consider the following control law:

$$u(k) = G(x(k)), \quad \forall k \geq 0$$

in which $G$ is a feedback control law. Since the manipulated inputs are determined by the states, they are dependent on the state measurements and are no longer freely available to be selected, and thus the set $AIS^l$ is not considered. The Achievable Output Set at time $k$ with fixed control law $G$ ($AOS_G(x_0, k)$) is the set of all possible outputs as a result of the disturbances given in $EDS^k$.

$$AOS_G(x_0, k) = \{ y(k) | \exists \{d(i)\}_{k_0}^k \in EDS^k : y(k) = \tilde{M} (x_0, \{u(i)\}_{k_0}^k, \{d(i)\}_{k_0}^k ; u(k) = G(x(k)) \} \}$$

When the process is input-to-state stable, the time-invariant $AOS_G^\infty(x_0)$ is obtained with the procedure in figure 3.3, similarly to $AOS^\infty(x_0)$. Since $AOS_G(x_0, k)$ corresponds to the set of output deviations caused by the process disturbance, the measure $\mu(AOS_G^\infty(x_0))$ represents the hypervolume of the fluctuations of the process under closed-loop control.

The summary of the differences between dynamic mappings and operability analyses in subsection 3.3.3 and subsection 3.3.4 is provided in table 3.2.

If a layer of state estimation is considered in combination with the fixed control law $G$, the estimation errors can be a source of fluctuations in the closed-loop performance. Thus, the estimation error can be added to the $EDS^k$, and the above closed-loop analysis is executed as proposed in this subsection.
### Chapter 3

#### Section 3.4

<table>
<thead>
<tr>
<th>Aim 2: Dynamic mapping with fixed disturbance</th>
<th>Aim 3: Dynamic mapping with fixed control law</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objective</td>
<td>The objective is to analyze a control structure of a dynamic system by assessing whether the process disturbances are sufficiently rejected</td>
</tr>
<tr>
<td>Characteristics</td>
<td>Dynamic mapping is independent of the formulation of the controller</td>
</tr>
<tr>
<td>Available Input Set</td>
<td>The input set is the range of the manipulated variables</td>
</tr>
<tr>
<td>Achievable Output Set</td>
<td>The output set contains all reachable outputs via manipulating the control variables</td>
</tr>
<tr>
<td>Expected Disturbance Set</td>
<td>The disturbance set is the range of process disturbances</td>
</tr>
</tbody>
</table>

Table 3.2: Comparison between different dynamic operability mappings

#### 3.4 Modular Membrane Reactor Case Study

The proposed framework is demonstrated via a case study of a DMA-MR, which is an intensified reaction-separation unit that converts methane in natural gas to hydrogen and benzene. The reactions are carried out in the tube of the shell-and-tube DMA-MR design, and the membrane is highly selective toward hydrogen permeation. When hydrogen is removed from the reactive tube, the reaction equilibrium shifts toward the products resulting in higher methane conversion. A schematic of a co-current flow configuration of the DMA-MR is provided in figure 3.4.

![Figure 3.4: Schematic with inputs, outputs, and reactions of the DMA-MR](image-url)

Figure 3.4: Schematic with inputs, outputs, and reactions of the DMA-MR
For this study, a modular DMA-MR is assumed to be part of a modular natural gas utilization plant in the Marcellus Shale region. Since the natural gas extracted in this region corresponds to the largest recoverable shale gas reservoir in the United States\[43\], a feasible modular gas processing design for this region would be a promising candidate for mass production. The following assumptions are considered for the operability analysis of the modular DMA-MR:

- A modular DMA-MR is assumed to be transported by commercial trucks, so its design is limited by the dimensions of its shipping container\[12\];
- At different natural gas feeds, an identical modular DMA-MR is capable of producing sufficiently high benzene and hydrogen concentrations in the product streams, so that it can be fit for large-scale manufacturing;
- During online operation, uncertainties in the natural gas concentrations caused by upstream processes should not affect the achievability of the desired output specifications of the DMA-MR.

### 3.4.1 Dynamic modeling of the DMA-MR

The dynamic model of the DMA-MR is considered to have temperature and pressure controlled, i.e., the DMA-MR model is assumed to be isothermal and isobaric. The pressure drops are neglected, and the flow rate is driven by the pressure profiles in the tube and the shell. Radial and angular symmetries are adopted, and only the change of states along the length of the DMA-MR is considered. Additionally, the equation of states is assumed to be the ideal-gas law\[44\]. The dynamic model of the DMA-MR is a system of partial differential equations with differential independent variables as the time and length of the DMA-MR. To solve this model, the method of lines is applied to discretize the length of the DMA-MR into equal partitions of differential length $\Delta z$. The resulting dynamic model of the DMA-MR is a system of ordinary differential equations with respect to time.
In particular, for the non-oxidative conversion of methane, the following two-step reaction mechanism is adapted from the published literature[11], [37], [45]:

Step 1:

\[
2\text{CH}_4 \leftrightarrow \text{C}_2\text{H}_4 + 2\text{H}_2
\]

\[
r_1(z, t) = k_1 C_{t,\text{CH}_4}(z, t) \left(1 - \frac{k_1' C_{t,\text{C}_2\text{H}_4}(z, t)}{k_1 C_{t,\text{CH}_4}(z, t)^2}\right)
\]

Step 2:

\[
3\text{C}_2\text{H}_4 \leftrightarrow \text{C}_6\text{H}_6 + 3\text{H}_2
\]

\[
r_2(z, t) = k_2 C_{t,\text{C}_2\text{H}_4}(z, t) \left(1 - \frac{k_2' C_{t,\text{C}_6\text{H}_6}(z, t)}{k_2 C_{t,\text{C}_2\text{H}_4}(z, t)^3}\right)
\]

in which \( z \) indexes the spatial locations along the discretized length of the DMA-MR and \( t \) indexes the process time of all variables, \( r_1 \) and \( r_2 \) are respectively the reaction rates of steps 1 and 2, \( k_1 \) and \( k_2 \) are respectively the forward reaction constants of steps 1 and 2, \( k'_1 \) and \( k'_2 \) are respectively the inverse reaction constants of steps 1 and 2, and \( C_i \) are the concentrations of the species \( i \) in the tube.

The rate of reactions of each species is calculated based on the stoichiometry of steps 1 and 2 as shown below.

\[
R_{\text{CH}_4}(z, t) = r_1(z, t)\pi \Delta z(D_t/4)^2
\]

\[
R_{\text{C}_2\text{H}_4}(z, t) = (-r_1(z, t)/2 + r_2(z, t))\pi \Delta z(D_t/4)^2
\]

\[
R_{\text{H}_2}(z, t) = (-r_1(z, t)/2 - r_2(z, t))\pi \Delta z(D_t/4)^2
\]

\[
R_{\text{C}_6\text{H}_6}(z, t) = (-r_2(z, t)/3)\pi \Delta z(D_t/4)^2
\]

in which \( R_i \) are the reaction rates of species \( i \) in the tube, and \( D_t \) is the tube diameter.

The permeations of each species through the membrane are driven by the partial pressure gradients between the tube and the shell. For an ion-based perovskite membrane, the membrane flux expression is proportional to the difference of the partial pressure
gradient raised to the power of $1/4$[46]. The mass fluxes are represented by:

$$J_i(z, t) = \frac{Q}{\alpha_{H_2/i}} \left( P_{t,i}^{0.25}(z, t) - P_{s,i}^{0.25}(z, t) \right) \pi D_t \Delta z$$

(3.35)

in which $J_i$ is the molar flux of species $i$ from the tube to the shell, $P_{t,i}$ is the partial pressure of species $i$ in the tube, $P_{s,i}$ is the partial pressure of species $i$ in the shell, $D_t$ is the tube diameter, $Q$ is the $H_2$ permeance through the membrane and $\alpha_{H_2/i}$ is the membrane’s selectivity between $H_2$ and component $i$.

At the inlets of the DMA-MR, the total molar flow rate of the tube is denoted as $F_{t,0}$, and the total molar flow rate of the shell is denoted as $F_{s,0}$. Using the ideal-gas equation of states, the total molar concentrations in the tube $C_t$ and shell $C_s$ are calculated according to the reactor temperature $T$, the tube pressure $P_t$, and the shell pressure $P_s$. From the isobaric and isothermal assumptions and disregarding process start-up, the total concentrations are constant with respect to time and reactor length. The following molar constraints are always active in the dynamic model:

$$C_t = C_{t,CH_4}(z, t) + C_{t,C_2H_4}(z, t) + C_{t,H_2}(z, t) + C_{t,C_6H_6}(z, t)$$

(3.36)

$$C_s = C_{s,CH_4}(z, t) + C_{s,C_2H_4}(z, t) + C_{s,H_2}(z, t) + C_{s,C_6H_6}(z, t)$$

(3.37)

From the ideal-gas law assumption, the molar flow profiles in the tube and the shell of the DMA-MR are scaled linearly with the molar concentrations, as shown below.

$$F_{t,i}(z, t) = \left( C_{t,i}(z, t)/C_t \right) F_{t,0}(z, t)$$

(3.38)

$$F_{s,i}(z, t) = \left( C_{s,i}(z, t)/C_s \right) F_{s,0}(z, t)$$

(3.39)

$$F_{t,i}(0, t) = C_{t,i}(0, t)V_{tube}(t)$$

(3.40)

$$F_{s,i}(0, t) = C_{s,i}(0, t)V_{shell}(t)$$

(3.41)

in which $F_{t,i}$ and $F_{s,i}$ are respectively the molar flow rate of species $i$ in the tube and in the shell, $V_{tube}$ and $V_{shell}$ are respectively the volumetric flow rate of species $i$ at the inlet
of the tube and the inlet of the shell.

Thus, the mass balances of the dynamic model for the DMA-MR are given by the following ordinary differential equations for each species:

\[ A_t \Delta z \frac{dC_{t,i}}{dt} = F_{t,i}(z,t) - F_{t,i}(z+1,t) + R_i(z,t) - J_i(z,t) \]  \hspace{1cm} (3.42)

\[ A_s \Delta z \frac{dC_{s,i}}{dt} = F_{s,i}(z,t) - F_{s,i}(z+1,t) + J_i(z,t) \]  \hspace{1cm} (3.43)

in which \( A_t \) and \( A_s \) are respectively the cross-sectional areas of the tube and the shell. The parameters of the dynamic model are adapted here from the existing steady-state model in the literature\[47\].

The dynamic model of the DMA-MR in this subsection is formulated for equation-oriented platforms, such as MATLAB and Python. Specifically, the time evolution of the model is solved with `odeint` subroutine in Python or `ode15s` subroutine in MATLAB.

### 3.4.2 Feasible design region of the DMA-MR

Starting from the dynamic model of the DMA-MR, the steady-state model was constructed by setting the left-hand sides of equations (3.42) and (3.43) to zero. The steady-state model is thus a system of nonlinear equations, that is solved using the MATLAB’s subroutine `fsolve`. In this subsection, a feasible design of the DMA-MR is obtained from the steady-state operability analysis proposed in subsection 3.3.2. The objective of the analysis in this subsection is to find the set of DMA-MR designs that guarantee sufficiently high product concentrations when operating around different natural gas wells at the Marcellus Shale.

The steady-state AIS in this case has two design variables and one optional manipulated variable as operability inputs. The two design variables are the tube diameter and the length of the DMA-MR. Since the tube of the DMA-MR is inserted in the shell, the shell diameter is chosen to be 10 cm larger than the tube diameter in all simulations. The considered disturbance corresponds to the methane concentrations at different natural gas wells, and the disturbance range is defined based on historical data\[43\]. The design
specifications for the steady-state analysis are listed in Table 3.3.

<table>
<thead>
<tr>
<th>Input Variables</th>
<th>Available Ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length ($dm$)</td>
<td>$50 - 100$</td>
</tr>
<tr>
<td>Diameter ($dm$)</td>
<td>$1 - 5$</td>
</tr>
<tr>
<td>Feed flow rate of tube ($dm^3/h$)</td>
<td>$500 - 1000$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output Variables</th>
<th>Desired Ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td>benzene mole percent in tube outlet (%)</td>
<td>$&gt; 5$</td>
</tr>
<tr>
<td>hydrogen mole percent in shell outlet (%)</td>
<td>$&gt; 17$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Disturbance</th>
<th>Expected Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feed methane mole percent (%)</td>
<td>$78 - 93$</td>
</tr>
</tbody>
</table>

Table 3.3: Steady-state operability sets of the DMA-MR: $AIS$, $DOS$, and $EDS$

For the performed analysis, each variable range of the $AIS$ and the $EDS$ is partitioned into 20 equal segments. Each $AOS_u(d)$ is obtained by solving the steady-state model considering all combinations of the designs in the $AIS$ at every value of the disturbance in the $EDS$. The blue polytope in Figure 3.5a is the $AIS$ considered for the steady-state operability analysis. In Figure 3.5b, the results of the forward input-output mapping are 20 sets of all reachable product concentrations of the DMA-MR for different natural gas wells. In this case study, the outputs of the $DOS$ are molar concentrations, so it is implied that their upper bounds cannot exceed 100%. For illustrative purposes, Figure 5b does not show the whole $DOS$, because the $AOS_d$ would be disproportionally smaller. For each value, $d$, of the disturbance in natural gas concentration, $DIS_u(d)$ is the $AIS$ subset that contains all designs of the DMA-MR that can reach the desired output concentration, which is represented by the $DOS$. The intersection of all $DIS_u(d)$ results in the feasible design region $DIS$, as illustrated in Figure 3.5a.

In Figure 3.6, the significance of the feasible design region is demonstrated. This figure compares the impacts of the disturbance on an infeasible steady-state design and a possibly feasible steady-state design. For each design, different disturbance values shift the steady-state outputs along a segment, and a feasible design is the one with the segment contained within the $DOS$. Specifically, if an infeasible steady-state design is chosen for the DMA-MR, that is a point in the $AIS$ but not in the $DIS$, then the benzene percentage is not sufficiently high to be included in the $DOS$, for example, the case in which this membrane reactor is operated at a natural gas well with low methane...
3.4.3 Operable design region of the DMA-MR

The feasible design region of the DMA-MR given by the $DIS$ only reflected the ability of the system to operate at a steady-state condition, so dynamic operability is needed to evaluate whether the modular membrane reactor is now capable of rejecting the disturbance after installation at a modular gas processing plant. For the dynamic operability
analysis, the manipulated variables of the DMA-MR are the inlet volumetric flow rates of the tube and the shell. The range of the inlet tube flow rate in the $AIS^k$ obtained considers 20% deviations of the steady-state values from the steady-state $DIS$, and the range of the inlet shell flow rate in the $AIS^k$ is chosen between $1000 \, dm^3/h$ and $1400 \, dm^3/h$. The time-varying disturbance of the $EDS^t$ corresponds to a normally distributed random methane concentration in the feed stream with 82% mean and 2.5% standard deviation, which is estimated from the characterization of the natural gas wells in the Marcellus region.

All simulations of the dynamic operability analysis are initialized with their respective steady-state inputs from the $DIS$. In the following subsection 3.4.4, the start-up and shut-down of the modular unit are not considered, and the controller objective is simply disturbance rejection operation once the membrane reactor reaches the desired steady-state. Therefore, the feasible steady-state solutions from the steady-state operability analysis of the DMA-MR are used as both the initial conditions and the setpoints for the MPC in the dynamic operability study. To simplify the notation, the dependency on $x_0$ is implied from here on and thus removed from the dynamic operability sets of the DMA-MR. The initializations of the dynamic operability analyses are shown in figure 3.7.

For every time step, the inlet flow rates and the inlet methane concentration could take any value within their given ranges. For the dynamic operability mapping, each manipulated input is discretized into 5 evenly spaced points between their boundaries, and each time-step requires $(5 \times 5)$ simulations for two manipulated inputs. The threshold $\varepsilon$ for the time-invariant $AOS^\infty$ calculated using equation (3.25) is set to 0.1, and the discrete-time difference $\Delta t$ in equation (3.19) is chosen to be 1 minute. As a result, all $AOS_u(d,k)$ reached their time-invariant sets in 8 discrete-time steps, and each set of $AOS_u(d,k)$ required $(5 \times 5)^8 \approx 4.29 \times 10^6$ simulations. This is due to the fact that the number of simulations increases as a scenario-tree for the inputs as the time grows. Because the number of simulations increases exponentially with the number of time steps, the 8-step discrete-time horizon to reach time-invariancy is found using a trial and error
Figure 3.7: Initialization of dynamic operability analysis from the results of the steady-state operability analysis

For each feasible steady-state input

\[
\begin{align*}
\text{Initialize: } x_0 &= x^{feasible} \\
\text{Setpoint: } y^{sp} &= y^{feasible}
\end{align*}
\]

method for one fixed design, and it is then applied to all dynamic operability mapping cases of feasible designs. Figure 3.8 shows a set of \( AOS_u(d, k) \) at the mean value of the disturbance sequence, in which the reactor length is 70 dm, the reactor diameter is 4 dm, and the time-zero steady-state has the nominal natural gas flow rate of 500 \( dm^3/h \). The red and purple color gradients of the AOS index the time differences, and the green lines are the Monte Carlo simulations of randomly selected available manipulated inputs to demonstrate that the time-varying \( AOS_u(d, k) \) covers all possible values of the outputs.

Figure 3.8: Dynamic achievable output sets for a fixed value of the disturbance
Before the full disturbance range in the $EDS^k$ is considered, a simplified case of dynamic operability is provided in figure 3.9 to demonstrate the analysis proposed in subsection 3.3.3. In this example, the $EDS^k$ only has two sequences $\{d(i)\}_{0}^{k}$ of the disturbance. The first disturbance sequence results in the set of $AOS_u(d, k)$ that is illustrated by the green bordered polytope funnel, and the second disturbance sequence results in the set of $AOS_u(d, k)$ that is illustrated by the red bordered polytope funnel. Each funnel is generated until both sets of $AOS_u(d, k)$ reach their time-invariant set. For every output $y(k)$ in the intersection of both $AOS_u(d, k)$ sets, which are illustrated with a blue-filled polytope funnel, there exists a sequence of input manipulated variables $\{u(i)\}_{0}^{k}$ for each disturbance sequence that map to $y(k)$. Thus, the blue funnel is the set of all achievable outputs regardless of the assumed disturbance sequences.

For the complete dynamic operability analysis, the $EDS^k$ is discretized at each time step $k$ by partitioning the 99% highest density interval of the disturbance into 5 evenly spaced values. Since the disturbance at every time $k$ is assumed independent and identically distributed, the $EDS^k$ included $5^8 \approx 3.9 \times 10^5$ disturbance sequences. At every time step $k$, the intersection of all $AOS_u(d, k)$ at different disturbance sequences is done similarly to the simplified case. In figure 3.10, $AOS_u^\infty(d)$ are represented by the empty-filled polytopes with the boundary’s color gradients indicating different values of the distur-

Figure 3.9: Dynamic operability analysis for two disturbance sequences
bance sequences. The resulting time-invariant $AOS^\infty$ sets are shown as filled polytopes, and their colors represent the ranges of a fixed-time disturbance. A design of a modular DMA-MR is operable if the intersection, which is illustrated as the red-filled polytope, of all $AOS^\infty_u(d)$ has a nonnegative $\mu$ measure. If an output setpoint is chosen in $AOS^\infty$, then this setpoint would be operable, meaning it could be achieved with a bounded input sequence in $AIS^t$ regardless of the disturbance sequences in $EDS^k$. Since the disturbances in the $EDS$ are Gaussian random variables, the operability condition could be relaxed by considering smaller ranges of disturbances realizations, and the $EDS$ satisfaction percentage is the probability of achieving a stochastic setpoint after the relaxation. The stochastic operability concept is especially important when the manipulated inputs fail to compensate for the disturbance effects. In figure 3.10b, the variance of the inlet methane concentration is increased from 2.5 to 5, and the $AOS^\infty$ associated with the expanded $EDS^k$ is now empty. While the DMA-MR is not operable in this case, it is stochastically operable with an $EDS$ satisfaction percentage of 85%. In other words, a setpoint in the small yellow-filled polytope in figure 3.10b could be reached 85% of the time. This could potentially be used to help define chance constraints for stochastic model predictive controllers.

![Figure 3.10: Effects of $EDS^k$ on the dynamic operability via the intersections of $AOS^\infty_u(d)$](image)

In the case study of the DMA-MR, all designs in the $DIS$ have nonnegative measures of their $AOS^\infty$ with the $EDS^t$ corresponding to the 99% highest density region. Therefore, the feasible design region is also the operable design region for this modular membrane
3.4.4 Closed-loop control analysis of the DMA-MR

The controller considered for the closed-loop analysis in this subsection is a nonlinear model predictive control (NMPC). NMPC is formulated as a constrained nonlinear programming problem that optimizes a control objective while adopting the DMA-MR dynamic model as equality constraints. At every time step, the following optimization problem is solved:

$$\min_{u(k), x(k)} \sum_{k=1}^{N} \left( (y(k) - y^{sp})^T Q_{MPC} (y(k) - y^{sp}) + (u(k) - u(k-1))^T R_{MPC} (u(k) - u(k-1)) \right)$$

subject to

$$x(0) = x_k$$

$$x(k+1) = f(x(k), u(k))$$

$$y(k) = h(x(k), u(k))$$

$$c_{ineq}(x(k), u(k)) \leq 0$$

in which the constraints in equations (3.46)-(3.48) represent the dynamic model of the DMA-MR given in subsection 3.4.1, the predictive horizon $N$ is chosen to be 10 minutes, the state-weighting matrix $Q_{MPC}$ is a diagonal matrix with each weight equals to 100, the manipulated input suppression matrix $R_{MPC}$ is an identity matrix, the setpoint $y^{sp}$ are the steady-state outputs mapped from the operable design region, and the internal dynamic model of the NMPC is initialized by the current state $x_k$. In practice, the current state $x_k$ is calculated from the measurements by a state estimation layer. However, here the state estimator is not considered in the closed-loop performance analysis of the DMA-MR, and $x_k$ is assumed to be directly measured.

The closed-loop analysis using the operability mapping proposed in subsection 3.3.4 is applied here for the DMA-MR. The control law $G(x(k))$ in equation (3.27) is chosen
to be the above NMPC. The ranges of the process disturbances included in the $EDS^k$ are assumed to be the same as the disturbance ranges in the dynamic operability analysis of subsection 3.3.3. For each operable design, the $AOS_G(k)$ is calculated until a time-invariant $AOS_G^\infty$ is achieved with $\varepsilon = 0.1$ at the 8 minute mark. For an operable DMA-MR, the $AOS_G(k)$ are defined as the tightest polytopes that bound the fluctuations of the output concentrations under closed-loop NMPC. An example of the $AOS_G(k)$ obtained for the DMA-MR is provided in figure 3.11. Since the feedback control law is fixed according to the state variables, the only expanding factors of the $AOS_G(k)$ for this case are the process disturbances. So the measure of the $AOS_G(k)$ represents the hypervolume of deviations from the setpoints caused by the disturbances.

![Figure 3.11: $AOS_G(k)$ of a DMA-MR with closed-loop NMPC](image)

For each operable design, the closed-loop analysis is performed until the $AOS_G(k)$ reaches their respective time-invariant sets. The hypervolumes of different $AOS_G^\infty$ sets for different operable designs are compared with each other to analyze their closed-loop performances. The summary of the measure $\mu$ of the $AOS_G^\infty$ obtained for the DMA-MR is given in figure 3.12. From this figure, the operable design coupled with an NMPC that gives the smallest setpoint deviations can be identified and shown as the point with the least output fluctuations.
Figure 3.12: Closed-loop performance analysis with dynamic operability of the DMA-MR
Chapter 4

Linear Operability for Time-Invariant System Analysis

4.1 Introduction

Process operability is defined as the ability to achieve desired performance from the given available inputs regardless of the realization of the disturbances. If the operability analysis is able to be carried out considering process operations, not only the achievable portions of the desired output sets can be known, but also the feasible output constraints can be provided for model predictive control to guarantee feasibility. However, the currently available operability analysis involves an exhaustive generation of the input combinations to map the achievable output region, and this approach quickly becomes intractable for a dynamic process considering the addition of time domain.

In this chapter, a dynamic operability mapping is developed to find a dynamically operable funnel for a linear time-invariant system. This funnel will help to determine optimal control laws for implementation considering stochastic disturbances and measurement noises. A novel two-step calculation is proposed, which includes an offline calculation of the nominal funnel by constructing a convex hull of the manipulated variable projections and an online update to shrink the funnel to an operable region in the presence of uncertainties.
4.2 Background

4.2.1 Preliminaries

The following discrete-time linear time-invariant dynamic system is considered for the dynamic operability analysis:

\[ x(k+1) = Ax(k) + Bu(k) + Gw(k); \quad x(0) = x_0 \]  

\[ y(k) = Cx(k) + Du(k) + v(k) \]  

in which at every discretized time step \( k \), \( x(k) \in \mathbb{R}^{n_x} \) is the vector of state variables, \( u(k) \in \mathbb{R}^{n_u} \) is the vector of manipulated inputs, \( w(k) \in \mathbb{R}^{n_w} \) is the vector of process disturbances, \( y(k) \in \mathbb{R}^{n_y} \) is the vector of output variables or controlled outputs, \( v(k) \in \mathbb{R}^{n_y} \) is the vector of measurement noises. The process disturbances and the measurement errors are simulated as zero-mean random vectors, which are drawn from multivariate Gaussian distributions, \( \mathcal{N}(0, \Sigma_w) \) and \( \mathcal{N}(0, \Sigma_v) \). If the disturbances and the measurement errors have non-zero means, there exists a bijective mapping that transforms the dynamic process at every time step to an equivalent zero-mean disturbance process. Thus, the non-zero assumption is valid without loss of generality. By definition, the covariance matrices, \( \Sigma_w \) and \( \Sigma_v \), are symmetric and positive definite. The matrices \( A, B, C, D, \) and \( G \) represent the transient dynamics and the projection of the states and manipulated inputs on to the output space. In the formulation of an MPC, \( k = 0 \) is defined to be the current time instead of the beginning of the closed-loop operation, and the current state variables, \( x_0 \), are given by a state estimation layer. The proposed dynamic operability analysis is applicable for all state estimation methods if the estimation errors can be assumed as Gaussian random vectors, as stated above.

In a linear dynamic system, the uncertainty propagation of Gaussian disturbances preserves its statistical properties, so the resulting state variables and controlled outputs will also be Gaussian. Therefore, the mean vectors and covariance matrices are sufficient to characterize the probability distributions of the states and the outputs at each time
step, and their sequences are computed as follows:

\[
\tilde{x}(k+1) = A\tilde{x}(k) + Bu(k) \quad \text{(4.3)}
\]

\[
\bar{y}(k) = C\tilde{x}(k) + Du(k) \quad \text{(4.4)}
\]

\[
\Sigma_x(k+1) = A\Sigma_x(k)A^T + G\Sigma_w G^T \quad \text{(4.5)}
\]

\[
\Sigma_y(k) = C\Sigma_x(k)C^T + \Sigma_v \quad \text{(4.6)}
\]

in which the vector of states, \( x(k) \), is a realization of the normal distribution, \( \mathcal{N}(\tilde{x}(k), \Sigma_x(k)) \), and the vector of outputs, \( y(k) \), is a realization of the normal distribution, \( \mathcal{N}(\bar{y}(k), \Sigma_y(k)) \).

While the initial condition or the present values of the state vector, \( x_0 \), are not affected by the present and future process disturbances, it can also be a random vector because of the uncertainty associated with the state estimation layer.

In operability analysis, the realization of the disturbance is often adapted as the expected disturbance set, and the support of a normally distributed random vector \( p \in \mathbb{R}^{n_p} \sim \mathcal{N}(\bar{p}, \Sigma_p) \) is the entire n-dimensional vector space in \( \mathbb{R}^{n_p} \). To ensure the tractability of the operability analysis computation, a bounded disturbance set is needed to study the effectiveness of a given bounded manipulated input set to achieve the controlled outputs, so the 95\% highest density region of the random vector \( p \), \( \text{HDR}(p) \), is chosen to approximate the support. For a Gaussian distribution, the \( \text{HDR}(p) \) corresponds to the ellipsoid with the scale \( l_p^2 \) equals to the inverse cumulative distribution of the chi-squared distribution with \( n_p \) degrees of freedom as shown in the equation below.

\[
\text{HDR}(p) = \left\{ p \mid (p - \bar{p})^T \Sigma_p^{-1} (p - \bar{p}) \leq l_p^2 \right\}, l_p^2 = \text{Inv}_{\chi^2}(95\%; n_p) \quad \text{(4.7)}
\]

The considered disturbance set can thus be relaxed or tightened by decreasing or increasing the 95\% threshold of the inverse cumulative chi-square distribution in (4.7).
4.2.2 Dynamic operability sets

A system is operable if it can reach the desired output performance requirements with the given manipulated input ranges regardless of the realization of the disturbances[2]. By assessing whether the inputs can compensate for the disturbances, dynamic operability evaluates the effectiveness of a control structure during online operations. The following operability sets are constructed using a process model, and the relationships between different variables are quantified with topological set operations in the Euclidean vector space.

Achievable Input Set at the discretized time $k$ ($AIS^k$): The set of all feasible manipulated input sequences potentially impacting the predictive output $y(k)$. This consists of the manipulated inputs (e.g., pressure, valve positions, flow rates, etc.) from the initial time $0$ until the time step immediately preceding the predictive time $k$ that can be freely adjusted during the operations. In practice, the inputs are limited by a lower bound, $u_{\min}$, and an upper bound, $u_{\max}$, due to equipment specifications or physical constraints (such as choked flow).

$$AIS^k = \{ u_k = [u(0)^T, u(1)^T, \ldots, u(k-1)^T]^T | u_{\min} \leq u(i) \leq u_{\max} ; \forall 0 \leq i \leq k - 1 \}$$

(4.8)

Expected Disturbance Set at the discretized time $k$ ($EDS^k$): The set of all realizations of disturbances from the initial time $0$ until the time step immediately preceding the predictive time $k$. The vector of disturbances in the operability analysis is a random vector with a joint probability distribution from the process disturbances and the measurement noises. Thus, a vector of disturbances at time $k$, $d(k)$, is a zero-mean Gaussian random vector of dimension $n_d = n_w + n_y$ and a covariance matrix $\Sigma_d = diag(\Sigma_w, \Sigma_v)$. At every time step, the $EDS^k$ is bounded by the highest probability density region, which is an
ellipsoid described by the following quadratic constraints.

\[
EDS^k = \left\{ d_k = [d(0)^\top, d(1)^\top, \ldots, d(k-1)^\top]^\top \left| \begin{array}{l}
  d(i) = [w(i)^\top, v(i)^\top]^\top \\
  d(i)^\top \Sigma_d^{-1} d(i) \leq l_d^2 \\
  l_d^2 = \text{Inv} \chi^2 (95\%; n_d) \\
  \forall 0 \leq i \leq k - 1
\end{array} \right. \right\}
\]

(4.9)

**Achievable Output Set at the discretized time \( k \) given a fixed sequence of disturbance \( d_k \) (AOS\(_y\)(\(d_k, k\)))**: The set of all possible controlled outputs resulting from the mapping AIS\(_k\) and the disturbance sequences \( d_k \) through the dynamic model (4.1)-(4.2). In the following discussion, the mapping outputs are not necessarily the controlled outputs because the proposed approach indirectly compute the controlled output sets via the achievable state sets. To distinguish the two sets, a subscript \( y \) of AOS\(_y\) is introduced to indicate that the elements of the set are the vectors \( y(k) \), and other achievable output sets will follow the same notation convention. Unlike the AIS\(_k\) and the EDS\(_k\), the elements of the AOS\(_y\) have the exact dimensions at different time steps. This formulation allows a consistent comparison between different AOS\(_y\) of the same system. To distinguish the operability sets that admit time-series elements from the operability sets that admit snapshots in time, the superscript \( k \) is added to the sets with time-varying values, and the index \( k \) is added to the input argument of the sets with fixed-time values.

\[
AOS_y(d_k, k) = \left\{ y(k) \left| \begin{array}{l}
  (1), (2) \text{ are satisfied} \\
  u(i) \subset u_k \in AIS^k, d(i) \subset d_k \\
  \forall 0 \leq i \leq k - 1
\end{array} \right. \right\}
\]

(4.10)

**Achievable Output Set at the discretized time \( k \) (AOS\(_y\)(\(k\)))**: The set of all achievable controlled outputs at the \( k^{th} \) time step for all values of the disturbances. The AOS\(_y\) is the intersection of all AOS\(_y\)(\(d_k, k\)) for every value of \( d_k \in EDS^k \). When the disturbance sequence is fixed at \( d_k \), any output setpoints selected in the AOS\(_y\)(\(d_k, k\)) can be achieved by manipulating the inputs in the AIS\(_k\). As a result, for every setpoint in the intersection of all AOS\(_y\)(\(d_k, k\)), there exist at least one manipulated input sequence that can match the
outputs with the given setpoints. In other words, the $AOS_y(k)$ is the set of all reachable output setpoints regardless of the disturbances, and it represents the greatest extent to which the controller can compensate for the effects of the disturbances.

$$AOS_y(k) = \bigcap_{d_k \in EDS^k} AOS_y(d_k, k)$$  \hspace{1cm} (4.11)

*Desired Output Set at the discretized time $k$ ($DOS(k)$):* The set of all targeted output vectors at the $k^{th}$ time step. This set represents the satisfactory output specifications, such as sufficiently high yield or manageable temperature ranges. In general, the $DOS(k)$ is a set of time-varying nonlinear constraints that describes the desirable transient and steady-state outputs[1]. In the online operability analysis of closed-loop dynamic systems, the $DOS(k)$ is used as the reference to evaluate the $AOS_y(k)$, and it is simplified to a time-invariant bounded region.

$$DOS = \{ y(k) \mid y^{\text{min}} \leq y(k) \leq y^{\text{max}} \}$$  \hspace{1cm} (4.12)

By definition, a process is operable if the intersection between the $AOS_y(k)$ and the $DOS(k)$ is non-empty, and the emptiness is validated quickly by solving for a basic feasible solution of a linear programming problem that takes the $AOS(k)$ and $DOS(k)$ to be the constraints. Thus, the existence of the intersection, which indicates operability, can be computed efficiently by following Phase I of a simplex algorithm. If the intersection exists, it can be accurately obtained via the convex hull of suitable geometric duals with respect to the feasible solution[48].

4.3 Dynamic Operability Mapping Framework

4.3.1 Overview of the framework

In a real-time application, the computation speed of the MPC algorithm determines the responsiveness of the closed-loop system to mitigate the disturbance effects. Similarly, the dynamic operability analysis must be performed efficiently to be compatible with
online operations. However, the presently existing operability mapping approaches are based on partitioning the achievable input sets evenly between their respective extreme values, and the achievable output sets are approximated by simulating the process at all possible input combinations. Because the manipulated inputs can have different values at each time instance, the number of simulations required to construct the dynamic funnel $AOS_y(k)$ increases exponentially with the number of predictive steps $k$, as a scenario tree.

The dynamic operability mapping for linear time-invariant systems is divided into two steps to address this challenge, and an illustration of the proposed framework is exhibited in Figure 4.1. The first step is constructing the achievable state sets, $AOS_x(0, k)$, for all values of $k$ in the prediction horizon, assuming the state variables are at the nominal values and the disturbances are fixed at their mean values. The collection of $AOS_x(0, k)$ at every time step $k$ of the prediction horizon is referred to as the nominal dynamic funnel. Calculating the nominal dynamic funnel in this step is the most computationally demanding stage in operability mapping, and it can be performed offline before the beginning of the process operations. During the online operation, the second step of the dynamic mapping updates the nominal funnel according to the newly arrived state information and the disturbance propagation described in (4.3)-(4.6). The updating step will be shown to be a set of linear algebra operations that can be rapidly executed online.

### 4.3.2 Offline mapping of the nominal dynamic funnel

The following assumptions are considered during the offline mapping step of the dynamic funnel and will be addressed during the online update step.

1. The current state variables are located at the origin of the state-space, i.e., $x_0 = 0_{n_x \times 1}$.

2. The $AOS(k)$ admits the state variables as output variables, i.e., $C = I_{n_x \times n_x}; D = 0_{n_x \times n_u}$.

3. The process disturbances and measurement errors are neglected, i.e., $w(k) = 0; v(k) = 0 \forall k$. 

49
From the above assumptions, the $AOS_\bar{x}(0, k)$ is the linear transformation of the $AIS^k$ in the state vector space via the mapping $\bar{B}_k = [A^{k-1}B \ A^{k-2}B \ \ldots \ AB \ B]$.

$$AOS_\bar{x}(0, k) = \{x(k) \mid x(k) = \bar{B}u_k; u_k \in AIS^k\} \quad (4.13)$$

From the definition (4.8), the $AIS^k$ is a bounded convex polyhedron. Thus, the $AOS_{\bar{x}}(0, k)$ resulting from (4.13) is precisely the tightest convex hull that contains all the projections of the available manipulated input sequences. Due to the linearity of the dynamic model, any linear combinations of $AIS^k$ inputs is a manipulated input sequence that equates to the respective linear combinations of their images in the state vector space. Thus, for an achievable state $x(k)$ to be a vertex, or an extreme point, of the $AOS_{\bar{x}}(0, k)$, its preimage, $u_k$, must also be a vertex of the $AIS_k$. The $AOS_{\bar{x}}(0, k)$ can be computed by taking the convex hull of the vertices of the $AIS_k$, which are the manipulated input sequences that only admit values from either the lower bound, $u^{\text{min}}$, or the upper bound, $u^{\text{max}}$.

$$AOS_{\bar{x}}(0, k) = \text{convexhull} \ \{\bar{B}u_k \mid u_k^T e_i \in \{(u^{\text{min}})^T e_i, (u^{\text{max}})^T e_i\}, \forall i \leq k \times n_u\} \quad (4.14)$$
in which \( e_i = [0, 0, \ldots, 0, 1, 0, \ldots, 0]^{\top} \in \mathbb{R}^{k \times n_u} \) is a standard basis vector for which only the \( i^{th} \) location has the value of 1, and the remaining elements are 0. From a set of achievable nominal state vectors, the Quickhull algorithm provides an efficient method of finding the convex hull\([49]\), which is the set of only vertices of \( AOS_{\tilde{z}}(0, k) \).

As the mapping procedure of the nominal dynamic funnel is performed offline, the computational time required for this step has no bearing on the online calculation of the dynamic funnel. However, the method of constructing the \( AOS_{\tilde{z}}(0, k) \) in (4.14) requires the exhaustion of all possible manipulated input sequences with boundary values. Thus, the number of simulations required to construct \( AOS_{\tilde{z}}(0, k), 2^{k \times n_u}, \) increases exponentially with the time step \( k \), and the offline computation will be intractable with a sufficient long prediction horizon.

To address this challenge, a novel state-space projection mapping procedure is proposed to compute the nominal \( AOS_{\tilde{z}}(0, k) \) sets. Instead of computing the \( AOS_{\tilde{z}}(0, k) \) directly as before, a chain of achievable output sets is constructed sequentially until the time \( k \) is reached.

\[
AOS_{\tilde{z}}(k+1) = \text{convexhull} \ \{x(k+1) = Ax(k) + Bu(k) \mid x(k) \in AOS_{\tilde{z}}(0, k); u(k) \in AIS^1\}
\]  

(4.15)

An achievable output vector, \( x(k) \), is in the \( AOS_{\tilde{z}}(0, k) \) constructed in (4.14) if, and only if, it also belongs to the set \( AOS_{\tilde{z}}(0, k) \) formulated in (4.15). The "only if" portion of the statement is proven by direct substitution of the linear state-space model and the fact that \( AIS^k \) is a concatenation of \( AIS^{k-1} \) and \( AIS^1 \). The "if" statement is proven using the induction method. For \( k = 1 \), the \( AOS_{\tilde{z}}(0, k) \) sets calculated using (4.14) and (4.15) are identical. If every element of (4.14) is also an element of (4.15) at \( k \), then the linearity of the dynamic process ensures the existence of a vector of states in \( AOS_{\tilde{z}}(0, k+1) \) according to (4.15) for each vector of states in \( AOS_{\tilde{z}}(0, k) \) according to (4.14). Based on these results, the convex hull of the outputs mapped from extreme points of the immediately preceding \( AOS_{\tilde{z}}(0, k - 1) \) is also the tightest polyhedron representing the \( AOS_{\tilde{z}}(0, k) \).

A flowchart that illustrates the offline state-space projection mapping for the nominal
dynamic funnel is shown in Figure 4.2.

![Figure 4.2: Offline mapping of the nominal dynamic funnel space with state-space projection procedure](image)

Using the above state-space projection, the nominal dynamic funnel is sufficiently constructed without considering all of the conceivable manipulated sequences within the bounds. The convexification at each time step is an effective method of pruning the manipulated input sequences that do not contribute to the alteration of the subsequent achievable state sets. Furthermore, the current work conjectures that the minimum number of simulations needed to generate the dynamic funnel grows polynomially with the time horizon length of the funnel. This conjecture is based on observing the impact of the $AIS^1$ on the set of achievable states as it progresses through time. At every fixed state vector, the set of all immediately subsequent achievable states is a linear transformation of the projection of the orthotopes $AIS^1$ on the state vector space. In other words, the $AOS_x(0, k)$ is the union of parallelotopes that have at most $2^{n_u}$ hyperplanes. Additionally, for any two parallelotopes generated from two distinguished fixed states of the previous time step, each hyperplane of one parallelotope has an affine hyperplane counterpart in the other parallelotope. For these reasons, the union of parallelotopes that results in the $AOS_x(0, k)$ must only have at most $2 \times n_u$ more hyperplanes than the polyhedron of the
4.3.3 Online update of the state funnel

In the previous subsection, the nominal dynamic funnel is represented with a set of convex hulls, \( AOS_x(0, k) \), which are convex polyhedra bounded by their vertices. According to the Minkowski-Weyl’s Theorem, every polyhedron is equivalently described by its vertex representation and its hyperplane representation, and thus the \( AOS_x(0, k) \) can be identically converted to a set of linear constraints using the Double Description Method[50].

\[
AOS_x(0, k) = \{ x(k) \mid \bar{H}_k x(k) \leq \bar{l}_k \} \quad (4.16)
\]

The advantage of the hyperplane representation is the ability to quickly validate if a setpoint is achievable by checking whether all linear constraints in (4.16) are satisfied. Furthermore, the dynamic funnel is calculated online by updating these linear constraints according to a new state measurement without repeating the procedure in Figure 4.2.

In the absence of process disturbances and measurement noises, the online update of the dynamic funnel can be established by substituting the linear dynamic state-space model into (4.16), resulting in the following expression for the mean dynamic funnel in its inequality form:

\[
AOS_x(0, k) = \{ x(k) \mid \bar{H}_k x(k) \leq \bar{l}_k; \bar{l}_k = \bar{l}_k + \bar{H}_k A^k x_0 \} \quad (4.17)
\]

in which the matrices \( \bar{H}_k x(k) \), \( \bar{H}_k A^k \), and the vector \( \bar{l}_k \) are unaffected by the new state vector \( x_0 \) and are computed offline. For each linear inequality of the nominal achievable set in (4.16), the updated value of \( x_0 \) shifts the hyperplane that defines its feasible half-space without changing the direction of the respective normal vector. Therefore, after the update, some redundant half-spaces may exist, i.e., removing one of these associated inequality constraints will not change the achievable region. However, the removed inequality can potentially be relevant for the calculation of the \( AOS_x(0, k) \) at different values of measured states \( x_0 \), so the online updates must always be computed using the nominal
dynamic funnel as reference instead of updating it from the dynamic funnel calculated at the previous time step.

From the linear Gaussian uncertainty propagation, the process disturbance effects are additive on the state variables. Thus, for each achievable state set at a fixed disturbance sequence, \( AOS_x(d_k, k) \), there is an identical achievable state set that is achieved by translating the mean achievable state set, \( AOS_x(0, k) \), by a vector \( w_x(k) \in HDR(w_x \sim N(0, \Sigma_x(k))) \). The \( AOS_x(d_k, k) \) defined according to the disturbance sequence similar to (4.10) is equally expressed as the following \( AOS_x(w_x(k), k) \) defined according to the deviations from the mean state vector:

\[
AOS_x(w_x(k), k) = \{ x(k) = \bar{x}(k) + w_x(k) \mid \bar{x}(k) \in AOS_x(0, k) \} \tag{4.18}
\]

The expected disturbance set is redefined according to the deviation, \( w_x(k) \), from the mean value of the state vector. The covariance matrix of \( w_k \) is the covariance matrix of the state vector, \( \Sigma_x(k) \), which is calculated online according to (4.3)-(4.6). The reformulated expected disturbance set is a set of disturbances at a fixed time step \( k \) instead of a constant disturbance sequence from time 0 to time \( k \).

\[
EDS_x(k) = \{ w_x(k) \mid w_x(k)^T \Sigma_x^{-1}(k) w_x(k) \leq t_x^2; \tilde{t}_x^2 = \text{In v}_{\chi^2} (95\%; n_x) \} \tag{4.19}
\]

The set of achievable states for all of the process disturbance realizations, \( AOS_x(k) \), is the intersection of every \( AOS_x(w_x(k), k) \) set for all values of \( w_x(k) \) in \( EDS_x(k) \).

\[
AOS_x(k) = \bigcap_{w_x(k) \in EDS_x(k)} AOS_x(w_x(k), k) \tag{4.20}
\]

An essential property of \( \Sigma_x(k) \) is the independence from the manipulated inputs, \( u(k) \), as shown in (4.5). Therefore, the dynamic operability analysis is valid for the prediction horizon for all formulations of the control laws. In a generalized dynamic operability analysis, the disturbance set in the achievable intersection is evaluated with a dense set of discretized values. Instead of calculating all \( EDS_x(k) \) approximations with a partition
of the ellipsoid, the exact polyhedra intersection in (4.20) is calculated by shifting inward each hyperplane of the \( AOS_x(0,k) \) by a distance bounded by \( EDS_x(k) \).

The independent and identically distributed property of the disturbances, \( d(k) \), at different values of \( k \) can be ignored without affecting the currently proposed method. This assumption is held valid in this work for the simplification of notation. If the framework is applied in a real process, the covariance matrices \( \Sigma_w \) and \( \Sigma_v \) will be replaced by the appropriate time-varying sequences \( \Sigma_w(k) \) and \( \Sigma_v(k) \) that represent the snapshots at the predefined discrete-time of the identified Gaussian process disturbances.

Let \( \Sigma_x(k) = R_k R_k^\top \) be the Cholesky factorization of the positive definite covariance \( \Sigma_x(k) \), in which \( R_k \) is a lower triangular matrix. At each time step \( k \), the bijective mapping \( L_k = R_k^{-1} \) is introduced to transform the state vector \( \hat{x}(k) = L_k x(k) \). The achievable output set at time \( k \) of the mapped state vector according to \( L_k \) at the mean value of the disturbance is defined as follows.

\[
AOS_{\hat{x}}(0,k) = \left\{ \hat{x}(k) \mid \hat{x}(k) = L_k x(k) \quad \forall x(k) \in AOS_x(0,k) \right\}
\] (4.21)

The hyperplane representation of the transformed achievable output set is achieved by substituting the mapping \( L_k \) into (4.16).

\[
AOS_{\hat{x}}(0,k) = \left\{ \hat{H}_k L_k^{-1} \hat{x} \leq \hat{l}_k \right\}
\] (4.22)

The objective of this transformation is to morph the ellipsoid region of the \( EDS_x(k) \) into an n-sphere for a more straightforward calculation of the \( AOS_x(k) \) in (4.20). Since \( x(k) \) is a Gaussian distributed random vector, the linear transformation, \( L_k \), results in a Gaussian process. The n-sphere shape of the newly transformed disturbance set is reflected through the covariance matrix of \( \hat{x} \) as an identity matrix.

\[
\Sigma_{\hat{x}}(k) = L_k \Sigma_x(k) L_k^\top = R_k^{-1} R_k R_k^\top \left( R_k^{-1} \right)^\top = I_{n_x \times n_x}
\] (4.23)

The expected disturbance set of the transformed state variables is defined as the
variation from the mean values, and it is mathematically described in (4.24). Since the linear mapping $L_k$ is bijective, the newly state vectors, $\hat{x}(k) \in \mathbb{R}^{n_x}$, have the same dimensionality as the original state vector, $x(k) \in \mathbb{R}^{n_x}$, so the scale of the semi-axes, $l_x^2$, in (4.19) is applied to the highest density region of $\hat{x}(k)$.

$$EDS_{\hat{x}}(k) = \left\{ w_{\hat{x}}(k) \mid w_{\hat{x}}(k)^\top w_{\hat{x}}(k) \leq l_{\hat{x}}^2; l_{\hat{x}}^2 = \text{Inv} \chi^2 (95%; n_x) \right\}$$ (4.24)

The transformed set of achievable states at all values of the transformed disturbances, $AOS_{\hat{x}(k)}$, is the intersection of all $AOS_{\hat{x}}(w_{\hat{x}}(k), k)$. This set is also a linear transformation of the $AOS_x(k)$ via the mapping $L_k$.

$$AOS_{\hat{x}}(k) = \bigcap_{w_{\hat{x}}(k) \in EDS_{\hat{x}}(k)} AOS_{\hat{x}}(w_{\hat{x}}(k), k) = \left\{ \hat{x}(k) \mid \hat{H}_k L_k^{-1} (\hat{x}(k) - w_{\hat{x}}(k)) \leq \tilde{l}_k \right\}$$ (4.25)

Since $AOS_x(w_x(k), k)$ is a translation of $AOS_x(0, k)$ according to the ellipsoid $EDS_x(k)$, the intersection of all $AOS_x(w_x(k), k)$ contains only the $x(k) \in AOS_x(0, k)$ that remains in $AOS_x(0, k)$ after shifted by a vector in $EDS_x(k)$. As a result, the computation of each hyperplane of $AOS_x(k)$ involves finding the normal direction of the hyperplane pointing toward its feasible half-space, and a geometric ray with the normal direction is then placed at the center to find its intersection with the ellipsoid $EDS_x(k)$ to find the hyperplane translation distance. This convoluted procedure is bypassed when the ellipsoid is an n-sphere according to Theorem 1.

**Theorem 1.** Let $[H]_i$ denote the $i^{th}$ row of a matrix $H : \mathbb{R}^{n_1} \to \mathbb{R}^{n_2}$. Given a bounded polyhedron in the form of $P_x = \{ x \in \mathbb{R}^{n_x} \mid H x \leq l \}$ and its image under a bounded translation according to an n-sphere $P_x(d) = \{ \hat{x} \mid \hat{x} = x + d; H x \leq l; d^\top d \leq l_d^2 \}$, the intersection of all $P_x(d)$ is given by:

$$P = \bigcap_{d^\top d \leq l_d^2} P_x(d) = \left\{ x \mid H x \leq \hat{l}; [\hat{l}]_i = [l]_i - l_d \sqrt{[H]_i^\top [H]_i} \forall i \leq n_2 \right\}$$ (4.26)

**Proof.** For each hyperplane $[H]_i x \leq [l]_i$, the hyperplane $[H]_i x \leq [l]_i - l_d \sqrt{[H]_i^\top [H]_i}$ is the parallel hyperplane shifted toward the feasible half-space by a distance of $l_d$. Thus, a
translation of all feasible points in $[H]_{i} x \leq [l]_i$ by a distance $d$ can only violate $[H]_{i} x \leq [\hat{l}]_i$, if, and only if, $d > l_d$. Therefore, $Hx \leq \hat{l}$ is the intersection of all hyperplanes $[H]_{i} x \leq [l]_i$ when the translation distance is less than or equal to $l_d$. \hfill \Box

Theorem 1 is the motivation of the transformation $L_k$ to transform the $EDS_x(k)$ to a special form that is suitable for a direct calculation of the intersection set, $AOS_x(k)$. Furthermore, the theorem is applicable for any transformation that morphs the $EDS_x(k)$ to a unit n-sphere, and an alternative $L_k$ can be derived from the eigenvalue decomposition of the covariance matrix, $\Sigma_x(k) = V_x(k)S_x(k)V_x^{-1}(k)$. Since $S_x(k)$ is a diagonal matrix with nonnegative elements, let $S^{-0.5}_x(k)$ denote a diagonal matrix of the inverse square root of the elements in $S_x(k)$. A bijective mapping $L_k = S^{-0.5}_x(k) V_x^T$ also transforms the dynamic operability sets into applicable forms for Theorem 1. The above Theorem 1 is applied to find the intersection of all transformations of $AOS_x(k)$ set by the respective disturbance set $EDS_x(k)$.

\begin{equation}
AOS_x(k) = \left\{ \hat{x}(k) \mid \bar{H}_k L_k^{-1} \hat{x}(k) \leq \hat{l}_k; \left[\hat{l}_k\right]_i = \left[\bar{l}_k + \bar{H}_k A^k x_0\right]_i - l_x \sqrt{\left[\bar{H}_k L_k^{-1}\right]^T_i \left[\bar{H}_k L_k^{-1}\right]_i} \right\}
\end{equation}

(4.27)

Since the $L_k$ is a bijective mapping, the achievable output set in the original vector space is obtained by inverting the transformed state, $\hat{x}(k)$, to the state variable, $x(k)$.

\begin{equation}
AOS_x(k) = \left\{ x(k) \mid \bar{H}_k x(k) \leq \hat{l}_k; \left[\hat{l}_k\right]_i = \left[\bar{l}_k + \bar{H}_k A^k x_0\right]_i - l_x \sqrt{\left[\bar{H}_k L_k^{-1}\right]^T_i \left[\bar{H}_k L_k^{-1}\right]_i} \right\}
\end{equation}

(4.28)

In this subsection, the nominal funnel of achievable state variables, which is computed offline following the procedure in subsection 4.3.2, is firstly updated with the new estimated state vector given by the state estimation layer. The process disturbance effects on the funnel are then reflected by the mean and the covariances of the Gaussian distributed state variables. Since such effects are additive, the intersection of all achievable funnels at different values of the disturbances is quickly calculated according to the highest density regions by shifting each hyperplane of the mean-valued funnel. Since the manipulated inputs are capable of steering the states to any interior point of the funnel for a fixed
value of the disturbance, the resulting commonly intersected funnel is the collection of achievable state sets at different discretized time steps regardless of the realization of the disturbances. The above fast set intersection method is shown to be a series of linear algebra operations. These operations are function evaluations and are not a recursion, so they can be efficiently computed online.

4.3.4 Online update of the output dynamic funnel

While the achievable state funnel is sufficient to characterize the transient behaviors of a dynamic process, the controlled outputs are the variables of interest for process analysis and control. In the previous subsection, the nominal state funnel, $AOS_x(0, k)$, is updated to an achievable state funnel in the presence of process disturbances, $AOS_x(k)$. The final stage of the online update step for dynamic operability mapping is projecting the $AOS_x(k)$ on the relevant output variables and accounting for the measurement errors. Let $AOS_y(0, k)$ denote the achievable output set at time $k$ given the achievable state set, $AOS_x(k)$, while neglecting the measurement errors.

$$AOS_y(0, k) = \{ y(k) = Cx(k) + u(k) \mid x(k) \in AOS_x(k); u(k) \in AIS^1 \}$$ (4.29)

The preimage set of the $AOS_y(0, k)$ before the linear mapping given by (4.2) is a union of the $AOS_x(k)$ and the $AIS^1$, and this set is a bounded convex polyhedron. Thus, all vertices of the preimage set are constructed by combining the extreme points of the $AOS_x(k)$ with the upper bounds and lower bounds of the manipulated inputs, and their respective controlled outputs are sufficient to construct the vertex representation of $AOS_y(0, k)$. The hyperplane representation of $AOS_y(0, k)$ is obtained via the Double Description Method[50]. Alternatively, the inequality constraints of $AOS_x(k)$ given in (4.28) are incorporated with the manipulated input constraints, and the combined inequality constraints are mapped to the output space. Accordingly, the hyperplane representation of $AOS_y(0, k)$ is the minimal hyperplane representation of the mapped inequalities.

$$AOS_y(0, k) = \{ y(k) \mid H_k y \leq \bar{b}_k \}$$ (4.30)
In a similar fashion to the reformulation of the $AOS_x(w_x(k), k)$ and $EDS_x(k)$ in (4.18) and (4.19), the measurement errors are considered the disturbances on the controlled outputs, and their effects are represented as the deviations from the expected values given the uncertainty propagation in (4.4) and (4.6). For every time step $k$, the achievable output set, $AOS_y(v_y(k), k)$, given an output deviation in $v_y(k)$ and the expected disturbance set on the outputs, $EDS_y(k)$, are mathematically reformulated as follows:

\[
AOS_y(v_y(k), k) = \{ y(k) = \bar{y}(k) + v_y(k) \mid \bar{y}(k) \in AOS_y(0, k) \} \tag{4.31}
\]

\[
EDS_y(k) = \{ v_y(k) \mid v_y(k)^\top \Sigma_y(k)v_y(k) \leq t_y^2; t_y^2 = \text{Inv} \chi^2(95\%; n_y) \} \tag{4.32}
\]

The achievable output set for all the disturbance values and measurement errors is the intersection of every $AOS_y(v_y(k), k)$ with the errors bounded by $EDS_y(k)$.

\[
AOS_y(k) = \bigcap_{v_y(k) \in EDS_y(k)} AOS_y(v_y(k), k) \tag{4.33}
\]

While the following mathematical expression of $AOS_y(k)$ is identical to the $AOS_x(k)$ in (4.20), the interpretations of the emptiness of these sets are different. If the intersection yields an empty achievable state funnel, $AOS_x(k)$, the process disturbances $w(k)$ overwhelm the manipulated variables $u(k)$, and control actions to bring the output to an exact setpoint do not exist. If the achievable output set, $AOS_y(k)$, is empty, the exact position of the outputs cannot be inferred from the measurements. However, the emptiness of the achievable state funnel and the achievable output funnel are not the same as the system being uncontrollable or unobservable. In some cases, the controlled outputs are guaranteed to be within a bounded region without knowing their exact values.

The exact intersection of the achievable output set in (4.33) is computed efficiently by applying Theorem 1. This theorem requires a bijective linear transformation that converts the $EDS_y(k)$ from an ellipsoid to an n-sphere. Let $\Sigma_y(k) = U_k U_k^\top$ be the Cholesky factorization of the output covariance matrix $\Sigma_y(k)$, in which $U_k$ is a lower triangular matrix. Let the linear mapping $L_{y,k}$ be the bijective mapping needed for Theorem 1, and the resulting output vector is $\hat{y}(k) = L_{y,k}y(k)$. The mean achievable output fun-
nel, $AOS_y(0, k)$, is a collection of achievable output sets if the measurement noises are neglected. Mathematically, the transformation and the hyperplane representation of the new achievable output funnel are the following expressions.

$$AOS_y(0, k) = \{ \hat{y}(k) = L_{y,k}y(k) \mid y(k) \in AOS_y(0, k) \}$$

$$= \{ H_k L_{y,k}^{-1} \hat{y}(k) \leq \bar{b}_k \}$$

(4.34)

The transformed disturbance sets, $EDS_y(k)$, that characterize the effects of measurement noises is an n-sphere because the covariance matrix of $\hat{y}(k)$ is an identity matrix according to the linear mapping $L_{y,k}$.

$$EDS_y(k) = \{ v_y(k) \mid v_y(k)\top v_y(k) \leq l_y^2; l_y^2 = \text{Inv}_{\chi^2}(95\%; n_y) \}$$

(4.35)

For the transformed linear dynamic system, the intersection of different achievable output sets, $AOS_y(v_y(k), k)$, at all values of the measurement noises in $EDS_y(k)$ is the following set of linear inequalities.

$$AOS_y(k) = \left\{ \hat{y}(k) \mid H_k L_{y,k}^{-1} \hat{y}(k) \leq b_k; [b_k]_i = [\bar{b}_k]_i - \sqrt{[H_k L_{y,k}^{-1}]}_i [H_k L_{y,k}^{-1}]_i \right\}$$

(4.36)

At each time $k$, the true achievable output set, $AOS_y(k)$, for all values of process disturbances and measurement errors are achieved from $AOS_y(k)$ by inverting the bijective mapping $L_{y,k}$.

$$AOS_y(k) = \left\{ y(k) \mid H_k y(k) \leq b_k; [b_k]_i = [\bar{b}_k]_i - \sqrt{[H_k L_{y,k}^{-1}]}_i [H_k L_{y,k}^{-1}]_i \right\}$$

(4.37)

### 4.4 Case study: HYPER process

The proposed framework for linear dynamic operability mapping is demonstrated through a selected case study of a hybrid energy system. The linear dynamic system chosen is the cyber-physical fuel cell-gas turbine hybrid power system in the HYbrid PERformance (HYPER process) project at the National Energy Technology Laboratory (NETL) in
Morgantown, WV. The HYPER process converts coal into a high-purity carbon monoxide (CO) stream, which is then used to generate electricity in solid oxide fuel cells (SOFCs). Compared to traditional coal-fired power plants, the process could reduce the carbon footprint of coal-based power generation by up to 90%. Additionally, it is capable of producing high-value chemicals from CO$_2$ emissions, which can be used in a variety of industrial applications. Figure 4.3 is a simplified schematic of the HYPER process, and a more detailed description of the process is available in the literature[51], [52].

![Figure 4.3: Simplified schematic of the HYPER process](image)

In the current work, the linear dynamics of the HYPER process are assumed to be accurately represented by the transfer function models derived from the physical gas turbine-fuel cell system at NETL. The primary focus of this case study is demonstrating a potential application of the proposed linear dynamic operability mapping. Thus, full consideration of the nonlinear dynamics of the HYPER process is beyond the scope of this chapter. For a more practical approach, the mismatch between the actual dynamics and the linear dynamics can be accounted for by adding a disturbance vector that represents this discrepancy. However, this plant-model mismatch is neglected in the following analysis due to the lack of first-principles mathematical model and operational data considered.
The following results should be interpreted as a conceptual example instead of a practical conclusion on the HYPER process.

Equation (4.38) describes the transfer function plant model, and equation (4.39) represents the additive effects of the disturbances on the controlled output.[53]

\[
\begin{bmatrix}
y_1 \\
y_2 \\
\end{bmatrix} = \begin{bmatrix}
-0.25e^{-0.1s} & -0.065e^{-0.5s} \\
-0.22e^{-0.56s} & -1.43e^{-0.7s} \\
\end{bmatrix} \begin{bmatrix}
u_1 \\
u_2 \\
\end{bmatrix} \quad (4.38)
\]

\[
\begin{bmatrix}
y_1 \\
y_2 \\
\end{bmatrix} = \begin{bmatrix}
0.17e^{-0.1s} & 0.03 \\
0.06e^{-0.56s} & -1.23e^{-0.64s} \\
\end{bmatrix} \begin{bmatrix}
w_1 \\
w_2 \\
\end{bmatrix} \quad (4.39)
\]

The control structure and the process variables of the above transfer function are summarized in Table 4.1. The process disturbances are assumed to be Gaussian random vectors that center at their respective nominal values, and the variances are assumed to be 15% of the nominal values. The measurement noises are assumed to be zero-mean Gaussian random vectors with variances equal to 5% of the controlled output nominal values. Since the transfer model is identified around specific operating conditions, the ranges of the manipulated inputs are implemented as \(u_{\text{min}}\) and \(u_{\text{max}}\) in the constructions of \(AIS_k\), and the admitted range of the electricity load, \(u_1\), is between 30 kW and 50 kW. The admitted range of the bypass valve opening for cold-air, \(u_2\), is between 40% and 80%.

<table>
<thead>
<tr>
<th>Notations</th>
<th>Process Variables</th>
<th>Nominal Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>(y_1) (controlled output)</td>
<td>Turbine speed (rpm)</td>
<td>40,500</td>
</tr>
<tr>
<td>(y_2) (controlled output)</td>
<td>Cathode airflow (kg/s)</td>
<td>0.75</td>
</tr>
<tr>
<td>(u_1) (manipulated input)</td>
<td>Electricity load (kW)</td>
<td>40</td>
</tr>
<tr>
<td>(u_2) (manipulated input)</td>
<td>Cold-air bypass (%)</td>
<td>40</td>
</tr>
<tr>
<td>(w_1) (disturbance)</td>
<td>Fuel valve (%)</td>
<td>50</td>
</tr>
<tr>
<td>(w_2) (disturbance)</td>
<td>Hot-air bypass (%)</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 4.1: Variables of the HYPER process’s control structure

In the current case study, the transfer function model is converted to a continuous-time state-space model using the MATLAB function \(tf2ss\), and the time discretization is performed using a zero-order hold with a sample time of 1 min. The resulting linear time-invariant dynamic model of the HYPER process follows the structure of (4.1) and (4.2),
and it consists of 4 state variables, 2 manipulated input variables, 2 process disturbances, 2 controlled outputs, and 2 measurement noises. The linear time-invariant dynamic system of the HYPER process is validated to be stable, controllable, and observable with the calculated dynamic matrices[53].

Since the transfer functions are identified from collected data of the experimental HYPER process, the state variables derived from the transfer functions do not correspond to physical properties. However, these states characterize the transient responses of dynamic behaviors. In the offline dynamic operability mapping, a nominal state funnel, $AOS_{x(0,k)}$, of the HYPER process is constructed using the state-space projection procedure described in Figure 4.2, and the result is shown in Figure 4.4. The nominal funnel is a set of linear constraints that bounds all the reachable states from a nominal initial condition given the bounds on the manipulated inputs. A Monte Carlo simulation is performed to validate the accuracy and precision of the proposed offline mapping method. In the Monte Carlo simulation, the HYPER process is initialized at the conditions in Table 4.1, and the manipulated variables at each time step $k$, $u(k)$, are randomly selected within the range given by $AI_{S}^{k}$. All Monte Carlo sample paths satisfy the constraint sets given by the hyperplane representation of $AOS_{x(0,k)}$, and this result is visually observed in Figure 4.4, in which no simulations exceed the interiors of the nominal funnel. An important result is the number of simulations performed to map the nominal funnel in Figure 4.4. For a 20 time-step prediction horizon, since the dynamic process has 2 manipulated inputs, the

![Nominal State Funnel and Monte Carlo Simulations](image)
number of possible input sequences with individual values either at the upper or lower bounds is \(2^{20 \times 2} \approx 1.1 \times 10^{12}\). However, the proposed state-space projection mapping in Figure 4.2 obtains the exact regions of \(AOS_{x(0,k)}\) with only \(1.1 \times 10^4\) simulations by truncating the repeated branches in the input scenario tree. The enhanced computational efficiency allows the dynamic operability analysis to be performed in a longer prediction horizon.

It is imperative to note that the Monte Carlo simulation shown in Figure 4.4 is not an uncertainty propagation but is a validation of the limit of input effects on the dynamic process. The purpose of the Monte Carlo simulation is to demonstrate that the constructed dynamic funnel contains every possible control action given the constraints on the manipulated inputs. Thus, the randomization of the inputs represents all the possible control actions that a controller or an operator can take rather than the disturbance effects. Additionally, because some state variables can be achieved from multiple different input sequences, some regions of the \(AOS_{x(0,k)}\) appear to have more Monte Carlo samples than other regions.

Since the HYPER system is observable, it is assumed that there is an ideal state estimation layer that provides full state information. The resulting achievable state funnel neglecting disturbance effects, \(AOS_{x(0,k)}\), is shown in Figure 4.5. As stated above, the current case study assumes that a linear dynamic model is an accurate representation of the HYPER process. To clearly illustrate the dynamic operability analysis in the scenario that the manipulated inputs do not sufficiently compensate for the disturbances, the new initial state vector is chosen to correspond to the steady-state outputs at values speed of 40,588 \(rpm\) and 133 \(kg/min\). This initial condition is likely to lie beyond the accurate output ranges of the transfer functions in (4.38) and (4.39) represent. Therefore, the following analyses only hold true for the linear dynamics, and they may not accurately reflect the true nonlinear dynamics of the HYPER process.

The nominal state vector and the nominal achievable funnel are respectively illustrated by the red star marker and the red polyhedra. In Figure 4.5a, a new initial state vector, which is depicted by the blue star marker, alters the orientation of the new achievable state
funnel, which is indicated with the blue polyhedra funnel. So far, process disturbances are not accounted for, and the updated state funnel is the collection of deterministic values.

In the online update, instead of regenerating the $1.1 \times 10^4$ simulations of the nominal funnel, the hyperplanes of nominal funnel are shifted according to (4.17). Additionally, Figure 4.5a shows that the mean funnel, $AOS_x(0, k)$, asymptotically converges to the nominal funnel, $AOS_x(0, k)$, as the time step $k$ increases. This behavior is expected for a stable process, and it also indicates the maximum horizon length for the HYPER process to recover from the perturbation that shifts the initial states. A Monte Carlo simulation with a sample size of 1000 is performed to validate that no input sequence in $AIS^k$ is not accounted for in the mean achievable state funnel computation.

![Diagram](image)

(a) Perturbation of the achievable state funnel at a different value of initial states
(b) Monte Carlo validation of the updated mean state funnel

Figure 4.5: Online calculation of the updated mean state funnel

In the HYPER process example, the achievable output funnel when the process disturbances and the measurement noises are at their respective mean values is constructed as a reference to compare with the achievable output funnel considering all disturbance realizations. In Figure 4.6, the achievable output funnel without disturbances, which consists of blue polyhedra, is the projection of the mean achievable state funnel, $AOS_x(0, k)$, in Figure 4.5. The achievable output funnel resulting from the two set intersections in (4.20) and (4.33) is calculated by sequentially applying (4.28) and (4.37) to the $AOS_x(0, k)$.

The dynamic operability analysis is then performed on the HYPER process assuming the $DOS$ of cathode airflow between 10 and 30 kg/min and turbine speed between 40490
and 40510 rpm. In a 4 min horizon representing transient dynamics, the intersection of the achievable output sets for all disturbances and noises is empty, so the HYPER process is inoperable because no output setpoints are guaranteed to be reached. Between the upcoming 4 and 8 min marks, non-empty $AOS_x(k)$ sets are observed, which indicates the manipulated inputs’ capability to regain operability from the perturbation on the initial condition and the disturbance effects. However, only after 8 minutes can the process guarantee to achieve the controlled outputs in the $DOS$ entirely. If all future disturbances and measurement noises are factored in the 95% highest density regions of the $EDS_x$ and $EDS_y$, there exists at least a sequence of available manipulated inputs that can steer the outputs to a setpoint in the $DOS$.

![Graphs](image)

Figure 4.6: Online update of the HYPER process operability funnel

The effects of the manipulated inputs and the disturbances can be compared online via dynamic operability mappings. Figure 4.7 illustrates the scenarios where variances of the considered disturbances are increased beyond 15% their nominal values. At 40% nominal values of the disturbances, the HYPER process requires a longer horizon for the controller to regain operability, and the set of guaranteed reachable output setpoints is significantly reduced. At more than 40% nominal values of the disturbances, the HYPER process is unable to compensate for all the disturbance effects with the given ranges of the available manipulated inputs. In such cases, no controller formulations are capable of operating the process, and a different control structure or a different design for the HYPER process would be needed.

From a process systems engineering perspective, the linear operability analysis demonstrates the relationship between the manipulated variable and expected disturbance ranges.
By analyzing this relationship, engineers can identify if the given control structure and manipulated input ranges are sufficient to counteract the expected disturbances. Since the analysis is independent of the control law formulation, the results apply to all controllers, including PID and model predictive control. Additionally, the output funnels are formulated as sets of inequality constraints, and the operable output funnel can be used as transient output constraints for model predictive control to keep the process within the operable regions. Since the linear dynamic operability analysis reveals the minimum time horizon for the control systems to regain operability after being affected by a large unscheduled disturbance, this horizon can also be used as the predictive horizon for model predictive control.
Chapter 5

Nonlinear Operability for Dynamic System Analysis

5.1 Introduction

Although the theoretical concepts of dynamic operability analysis have been thoroughly developed, their practical implementation is limited due to the computational tractability of the input-output mappings. The two primary sources of such limitations are the nonlinearity of the input-output relationships and the time-varying nature of operational inputs. The work in this chapter proposes two novel approaches to address these specific challenges. The first approach is a feedback operability mapping, which generates and evaluates the nonlinearity of the input-output function present. The second approach is a generalized state-space projection mapping procedure for nonlinear dynamic processes, which is motivated by the mapping solution for linear time-invariant systems.

For a linear dynamic process, a range of available inputs is sufficiently represented by the respective lower and upper bounds because any linear combination of the extreme input values will result in the linear combination at the extreme values of the outputs. However, mathematical models of chemical processes are generally nonlinear, and a range of available inputs is often partitioned into uniform segments to capture nonlinear behaviors with piecewise linear intervals to derive linearized models. When multiple inputs are
considered, the current operability mapping creates a dense grid of input combinations according to the ranges of the \textit{AIS}. Each input combination is provided to the process model, and the associated outputs can be computed as the solutions of the model. Typically, a full \textit{AIS} grid is generated prior to the calculations of the \textit{AOS}, and the number of simulations increases exponentially with the dimensionality of the \textit{AIS} assuming a fixed partition for each input. In the following section, the proposed feedback mapping avoids the input grid generation by evaluating the accuracy of the \textit{AOS} before deciding whether additional simulations are needed. Thus, the resulting \textit{AIS} of the feedback mapping is not an evenly discretized input grid, but an indexed set of available inputs, which is denser at highly nonlinear output regions.

The second major factor that contributes to the tractability challenges in dynamic operability mapping is the exponential growth of manipulated input sequences with the increase of the prediction horizon. While the process input variable set does not change with time, the input values can take different values within this set at different time instances, and each time-dependent manipulated input is considered a mapping input for the operability analysis. Combining this with the aforementioned computational challenge of fixed partition \textit{AIS} grid generation, it is challenging to create a reliable and accurate map of the behavior of a system given the large number of simulations required to capture the relationships between inputs and outputs. Motivated by this challenge, the state-space projection mapping procedure here is a newly introduced approach to quantify the \textit{AOS} of dynamic systems. Since the manipulated input sequences can be constructed by branching new input values from a previous input sequence, the simulation-generating structure of the dynamic process is similar to the construction of a scenario tree. The essential notion of the state-space projection is thus finding an effective way of pruning manipulated input sequences without losing any regions of the dynamic \textit{AOS}. In other words, it is an efficient method of trimming repeated branches of the scenario tree. Finally, as another contribution of this work, a novel multiple steady-state identification technique is proposed by observing the obtained dynamic operability mapping.
5.2 Nonlinear Feedback Operability Mapping

5.2.1 Polyhedra representation of operability sets

From the computational application perspective, process disturbances in the EDS can be interpreted as uncontrollable inputs of a dynamic process, and the EDS determines the shape of the intersection of the AOS in process operability analysis. However, the disturbances and manipulated inputs are both inputs for the mapping procedure, and they only differ in the operability set grouping for the software implementation. In the following discussion of feedback mapping, since the main objective is reducing the number of simulations needed while maintaining the necessary output region of the AOS, the disturbances are not considered. However, the feedback mapping procedure also applies to disturbance mapping by considering the disturbances as augmentation of the input vector. The following explicit mapping formula is considered:

\[ y = M(u) \]  \hspace{1cm} (5.1)

in which \( u \) is the vector of manipulated inputs, \( y \) is the vector of outputs in consideration, and \( M \) is the equivalent mapping function of a mathematical process model. The mapping \( M \) is assumed to be continuous and differentiable for the gradient \( \nabla u_y \) to exist at all values in the AIS.

In a recent operability development, the operability sets are represented with the multimodel approach, in which paired polytopes are employed to linearize the nonlinear mapping \( M \)[12]. The multimodel approach is an excellent technique to adapt the operability concept for process optimization and intensification. However, it is not suitable to perform set operations in conventional operability analysis, such as intersection and hypervolume calculation, and direct set validation is not available to verify if a variable is within a defined set. A polyhedron representation of the operability sets is presented for the proposed mapping solutions by modifying the multimodel representation, and it is described in the following paragraph. The main purpose of the polyhedron representation is to convert the operability set into vertex and hyperplane representations. In
this form, many computational geometric algorithms (e.g., minimal representation, convexification, pontryagin difference, hypervolume computations) can be applied from the current literature\[54].

The polyhedron representation keeps the pairing of AIS and AOS subregions as in the multimodel approach, but the individual vertex pairings are replaced with a union of inequality constraints for fast computational geometry calculations. Let $P_u = \{P_{u1}, P_{u2}, \ldots, P_{uK}\}$ be a non-overlapping partition of the AIS, in which $P_{ui}$ is a convex polyhedron with a hyperplane representation.

\[
P_{ui} = \{u \mid u = V_{ui}^{u} \lambda; \lambda \geq 0; 1^T \lambda = 1\} = \{u \mid A_{ui}^{u} u \leq b_{ui}^{u}\} \quad (5.2)
\]

in which for each $P_{ui}$, the vertices are concatenated to the matrix $V_{ui}^{u}$, and the inequality coefficient matrix $A_{ui}^{u}$ and vector $b_{ui}^{u}$ are obtained from the vertices using the Double Description Method\[50]. In the conventional grid partition of the AIS, the polyhedra $P_{ui}$ are identically shaped orthotopes.

Let $[V_{ui}^{u}]_j$ denote the $j^{th}$ column of the vertex matrix $V_{ui}^{u}$, and it also denotes the $j^{th}$ extreme point of the polyhedron $P_{ui}$. The vertices of the $P_{yi}$ are generated by mapping all $[V_{ui}^{u}]_j$ through the process model $\mathcal{M}$, and the Double Description Method is used to find the corresponding matrix inequalities, $A_{yi}^{u} y \leq b_{yi}^{u}$, for the hyperplane representation. Thus, the polyhedra representation of the AOS is the union of all image sets $P_{yi}$.

\[
P_{yi} = \{y \mid y = V_{yi}^{y} \lambda; [V_{yi}^{y}]_j = \mathcal{M} ([V_{ui}^{u}]_j); \lambda \geq 0; 1^T \lambda = 1\} = \{y \mid A_{yi}^{y} y \leq b_{yi}^{y}\} \quad (5.4)
\]

\[
AOS = \bigcup_{i=1}^{K} P_{yi} \quad (5.5)
\]

A union indicator function, $I_P(x)$, is then constructed for the AIS and AOS in the forms of (5.3) and (5.5). In the following expression (5.6), if the target set $P$ is the AIS, the variables $x$ are replaced with the process inputs, $u$, and the $\{A_i, b_i\}$ pair is replaced with $\{A_{ui}^{u}, b_{ui}^{u}\}$ in (5.2). Similarly, if the target set $P$ is the AOS, the variables $x$ are
replaced with the process outputs, \( y \), and \( \{A_i, b_i\} \) pair is replaced with \( \{A_i^y, b_i^y\} \) in (5.4).

\[
I_P(x) = \sum_i \prod_j \frac{1}{1 + e^{-k_s([A_i]_j x - b_i)}}
\]

(5.6)
in which \( k_s \) is a nonnegative smoothing constant to reduce the stiffness between the transition regions.

A demonstration of the union indication function proposed in (5.6) is illustrated in Figure 5.1. In this figure, a nonconvex AOS is represented by a union of multiple polyhedra, which are locally convex regions of the AOS. If input-multiplicity exists, some output polyhedra will overlap with other polyhedra, which are shown as the purple polyhedra in the upper-right corner of the figure. The \( I_P \) function is evaluated in the regions between 0 and 10 of each output variable, and the result is displayed in Figure 5.1b. In this figure, the indicator function values that approach zero for the outputs are not included in the AOS. For the outputs in the AOS, the values of the indicator function approach the number of overlapping polyhedra at those values, and the number of available input combinations that map to the same outputs can thus be identified.

![Figure 5.1: Union indication function example](image)

(a) Nonconvex overlapping Achievable Output Set. Each polyhedron represents a different subregion, which is differentiated by different colors

(b) Indicator function of the Achievable Output Set

In addition to the input-multiplicity enumeration, the proposed union indication function, \( I_P \), is continuous and twice differentiable. Thus, the operability sets can potentially
be translated into inequality constraints using the indicator for the previously proposed
operability-based process intensification and optimization[11], [12]. For example, a con-
straint \( y \in AOS \) can be replaced with \( I_{AOS}(y) \geq \epsilon \), which is a compatible expression with
typical nonlinear programming solvers.

5.2.2 Feedback mapping

The feedback mapping proposed here corresponds to a method of incremental generation
of smaller subdivisions of the \( AIS \) in the process of evaluating the \( AOS \). Intuitively, it is
similar to the adaptive importance sampling technique used to estimate the probability
distribution of complex systems[55]. At the initialization of the feedback mapping, only
the input combinations at the vertices of the \( AIS \) are simulated using the mapping \( M \) for
the generation of the \( AOS \). From the available inputs, the achievable outputs, and the
process model, a linear approximation of the mapping \( M \) is constructed. The vertices of
the \( AIS \) are simulated using the linearized mapping to achieve a linearized \( AOS \). If the
linearized \( AOS \) and the mapped \( AOS \) using \( M \) are within the same regions, simulating
the process model at interior values of the current \( AIS \) will unlikely yield a substantial
enhancement in the \( AOS \) level of detail. On the contrary, if the linearized \( AOS \) and the
mapped \( AOS \) using \( M \) are significantly different, more \( AIS \) internal input evaluations
are needed to reflect the nonlinearity of the \( AOS \). Since the newly generated inputs
also divide the \( AIS \) into smaller polyhedra, the process of simulating additional inputs is
referred to as the \( AIS \) splitting step. The procedure is repeated for each recently divided
polyhedron until a termination criterion is met. The first termination criterion is when all
\( P_y^i \) that construct the \( AOS \) are approximately equal to their linearized counterparts, which
means the current polyhedra representation is an accurate piecewise-linear estimation of
the nonlinear \( AOS \). The second termination criterion is when the maximum number of
polyhedron splits is reached, which prevents the feedback mapping from being stuck in
an infinite loop. The maximum splits value of 8 is recommended for each variable range.
A schematic of the feedback mapping framework is provided in Figure 5.2.

There are many possible methods of linearizing the mapping, \( M \), and the chosen
method does not affect the overall performance of the feedback mapping. If the model equations of the process are available and differentiable, a Taylor series expansion is recommended for ease of implementation. If the process model is constructed using an equation-oriented platform, the implicit function theorem can be applied to evaluate the gradient of the Taylor expansion. If the derivatives of the model are not accessible, a simple linear regression is sufficient to generate the linearized AOS. For each subdivision of the AIS, which is represented as a \( P_i \), let \( AOS_{lin} \) denote the achievable output set generated with the linear mapping. A linearity index, \( \Gamma \), is introduced using the operability measure, \( \mu \).

\[
\Gamma = \min \left( \frac{\mu (AOS_{lin} \cap AOS)}{\mu (AOS_{lin})}, \frac{\mu (AOS_{lin} \cap AOS)}{\mu (AOS)} \right) \quad (5.7)
\]

An important property of \( \Gamma \) is its convergence to 1 when the \( AOS_{lin} \) and the original AOS are becoming indistinguishable. If the \( AOS_{lin} \) or AOS are larger than their counterparts, the hypervolume of their intersection is less than the largest set, and the index \( \Gamma \) will decrease. If both achievable output sets have similar hypervolume but the
overlapping region is not the majority in either set, the value of the $\Gamma$ will reduce. Thus, the above linearity index is the measure of the similarity between the $AOS_{lin}$ and $AOS$. If the linearity index is less than a threshold, which is recommended as 85%, the current $P_u^i$ should be split.

The $AIS$ splitting procedure is performed effectively by leveraging the hyperplane representation of each subdivision. In the first step of the split, an available input variable is selected, and a hyperplane that passes through the midpoint between the highest and lowest available value for this input with a normal direction parallel to a unit vector of the chosen variable is calculated. This hyperplane is the cutting plane, and two half-spaces divided by the cutting plane are intersected with the initial polyhedron of the $AIS$ subregion to form two new polyhedra. The intersections of the cutting plane and the splitter $P_u^i$ are the values of new input combinations to be simulated. The input variable associated with the cutting plane is iterated through all available inputs. By selecting one input variable per split, the splitting procedure is simplified to solving a system of linear equations for the intersections while ensuring the more nonlinear output regions will receive more attention, i.e., have more simulations.

Finally, the feedback mapping partially reveals the input multiplicity behaviors of the analyzed process. Bijective operability mapping regions are subsets of the input space, in which each input combination results in precisely one output combination. Input multiplicity occurs when the mapped output regions of two neighboring bijective regions intersect. Since the process models of physical systems are continuous, input multiplicity leads to a fold on the $AOS$ at the boundary of two bijective input regions. At this boundary, the partial derivative $\frac{\partial y}{\partial x}$ of the outputs with respect to the inputs approach infinity, and any linearization attempts of $M$ will result in a large discrepancy between the $AOS_{lin}$ and the $AOS$. The bijective mapping boundary can be observed in the $AIS$ of Figure 5.2 as the concentrated input combinations. The boundary is particularly important for the inverse mapping problem for operability analysis, such as for evaluating the $DIS$. 
5.3 Nonlinear State-Space Projection for Dynamic Operability Mapping

5.3.1 Generalized state-space projection for discrete-time dynamic systems

In the state-space projection mapping, the following discrete-time state-space process is considered:

\[ x(k+1) = f(x(k), u(k)) \]  \hspace{1cm} (5.8)

\[ y(k) = h(x(k), u(k)) \]  \hspace{1cm} (5.9)

in which at time \( k \), the vector of state variables is \( x \), the vector of inputs is \( u \), the vector of outputs is \( y \). The discretized rate of change equations are represented by \( f \), and the output evaluations according to the current states and inputs are included in the function \( h \).

The state-space projection dynamic mapping method in this subsection is a generalization of the previously presented state-space projection for linear dynamic operability mapping. The objective of the dynamic operability mapping is generating a collection \( \{AOS(k)\} \) of achievable output sets at different time instances. To avoid the computational burden of simulating all the possible time-varying input combinations, the state-space projection generates a series of achievable state dynamic sets before constructing the achievable output funnel accordingly. From the state-space structure of the dynamic model, the achievable output sets at the following time instance depend on the range of the \( AIS \) and the bounded set of the current time achievable output set. Thus, the simplification, which is convexification for linear dynamics, of the achievable state sets at all time steps leads to the truncation of redundant input sequences. The summary of the state-space projection is illustrated in Figure 5.3.

The steps enumerated in Figure 5.3 are briefly described below.

\((1)\) Initialization: The notation of the dynamic operability set for the nonlinear ma-
(1) Initialization: 
Set $k = 0$; $AOS_x(k)$ is exhaustively generated

(2) Obtain the minimal polyhedron representation $AOS_x(k)$

(3) Merge the $AOS_x(k)$ and $AIS^1$

(4) Generate the $AOS_x(k + 1)$; $k = k + 1$

Terminate the state-space projection mapping and generate all $AOS(k)$

Figure 5.3: Nonlinear state-space projection for dynamic operability mapping

Mapping is adapted from the linear mapping notation. Let $AOS_x(k)$ be the set of all achievable state variables at the time $k$, and $AIS^1$ be the input sequence of a one-step horizon. The range of achievable inputs and the set of all possible initial conditions are collected from the process knowledge and constraints.

(2) Simplification of the current $AOS_x(k)$: In this step, simplification of the achievable state set is referred to a reorganization of the polyhedra in the hyperplane representation of (5.4), and not to the truncation of generated inputs-outputs pairs. If two neighboring polyhedra can be merged into a single convex polyhedron, then they are replaced with the union set. The procedure is repeated until no possible merges are available within the $AOS_x(k)$. At this step, the objective is to reduce the number of subregions created during the feedback mapping of step (4), and not to find the smallest number of convex polyhedra that can cover the achievable state set. The non-vertex achievable output variables generated during the feedback mapping (which are not needed for the polyhedra representation) are stored in an ordered array for reuse if needed in later steps.
(3) Generating the next mapping inputs: From the discrete-time model (5.8), the state variables at the next time step, $x(k+1)$, are dependent on both the current states, $x(k)$, and inputs, $u(k)$. Thus, the mapping inputs of $f$ in (5.8) is a concatenation of $x(k)$ and $u(k)$. For each polyhedron in the hyperplane representation of $AOS_x(k)$, the merging with the $AIS^1$ is efficiently performed by stacking the inequalities in the hyperplane representation of both $AOS_x(k)$ and $AIS^1$.

(4) Feedback Mapping: The proposed feedback mapping procedure is applied to generate the polyhedra union of the following time step, $AOS_x(k+1)$. Since the newly generated polyhedra can potentially be overlapped on top of each other, the union indication function is evaluated at all achievable outputs at the following time. If an output has the $I_p$ index higher than 1, any repetition of this output can be arbitrarily removed without compromising the overall $AOS_x(k+1)$ representation due to the convexity of each subregion.

5.3.2 Case study: Jacketed CSTR

As a proof-of-concept, the state-space projection dynamic mapping approach is demonstrated through its application to a continuous stirred-tank reactor (CSTR) for propylene glycol production with a heat exchanger jacket. The dynamic model of this process is available as educational material in the literature, which is sourced from a chemical engineering textbook[56]. The continuous time model is a system of nonlinear differential equations, and the discrete-time order is converted from the continuous time model, assuming the zero-order hold on the inputs and a numerical method for solving the differential equations between consecutive time steps.

In this type of reactor, the reactants, usually propylene oxide and water, are introduced continuously into a tank equipped with a stirrer to ensure a homogeneous mixture. The resulting mixture is heated to a specified temperature and maintained at that temperature for a certain duration to facilitate the reaction. This reaction is exothermic, generating heat during the process. To control the temperature, a cooling system is employed, such as a jacketed tank or a cooling coil. This system dissipates the heat generated during the
reaction and maintains the temperature at the desired level. The initial condition of the CSTR example is the steady state condition with the feeding rates of 36.3 kmol/hour for propylene oxide and 453.6 kmol/hour for water. The initialized steady state condition is also referred to as the nominal condition for the CSTR. The input ranges for the formulation of the AIS are assumed to be at the 10% deviations of their nominal values.

![Achievable output funnel for the jacketed CSTR](image)

Figure 5.4: Achievable output funnel for the jacketed CSTR

The dynamic funnel obtained for this system is shown in Figure 5.4. Similarly for the HYPER process case study in Chapter 4, a Monte Carlo simulation method is performed to validate the boundary of the achievable outputs given by the funnel. The random variables of the Monte Carlo simulation are the manipulated inputs to inspect the span of all possible control actions, and they are not considered for uncertainty propagation. As shown in the Figure, 1000 samples is drawn for the simulation to confirm that no input sequence bounded by the identified AIS results in outputs exceeding the regions of the dynamic funnel.

In the feedback mapping of the state-space projection case study, the maximum splitting limit of 7 is chosen, which is equal to a partition of each available input range into $2^7 = 128$ uniform intervals (this splitting limit can be increased if the hardware is capable of generating more simulations). The high-resolution discretization of the AIS and the polyhedra operations in the state-space projection leads to approximately one hour of
computational time for each time step calculation, as shown in Figure 5.5. A closer look at the computational effort shows that the most intensive step in the state-space projection is the hypervolume-based linearity test of the feedback mapping, and this challenge is recommended to be addressed in future work. Despite the time length needed for each step, the most important observation in this result is that the computational time does not scale exponentially with the prediction horizon length. As a result, nonlinear dynamic operability mappings for large-scale dynamic models using the proposed algorithms can potentially be benefited from parallel computing or more advanced hardware. This is a proof of concept that the proposed state-space projection effectively truncates the redundant input sequences, trimming the input scenario tree, for a nonlinear process. Finally, it is important to note that the state-space projection only addresses the increase in input dimensions caused by the dynamics. If more input variables are considered, further development in the feedback mapping algorithm would be needed to find the least number of simulations required to reflect the achievable output sets accurately.

![Figure 5.5: Computational time of the state-space projection mapping for the jacketed CSTR](image)

Figure 5.5: Computational time of the state-space projection mapping for the jacketed CSTR
5.4 Multiple Steady-State Identification

5.4.1 Limitation of current multiple steady-state identification methods

A pathway for systematically identifying multiple steady states was discovered when developing the dynamic mapping for the operability analysis above. A chemical process may have multiple steady states if an identical set of manipulated inputs converge to multiple stable process conditions. Identifying multiple steady states in a chemical process is a crucial aspect of process design and optimization in the chemical process industry. Such identification can offer valuable insights into the behavior of a chemical process under diverse operating conditions, thereby enhancing process control and safety. In the context of operability analysis, multiple steady-state identification is necessary to generate the full $AOS$, and a complete dynamic operability analysis requires the initializations of the process at all achievable steady-state values.

However, a systematic methodology to identify steady-state regions for general chemical processes is yet to be established. Multiple steady states in chemical processes are recognized using heuristics, such as observing the multiple heat generation and removal of a nonisothermal reactor[57]. Although this approach is reliable, it is not generalizable and applied case-by-case. Mathematical methods of identifying multiple steady states are classified into two major groups: bifurcation analysis and root-finding deflations.

Bifurcation analysis is a diverse class of algorithms, all relying on a numerical method known as the arc-length continuation. The basic idea behind arc-length continuation is to use the system solution path as a guide to vary the inputs incrementally. As the inputs change, the solution path may undergo bifurcations, such as the appearance of multiple steady states, and the regular input path integration to a new solution is no longer feasible. Arc-length continuation ensures that the solution path is followed as smoothly as possible through these bifurcations by keeping the solution on the same branch of the bifurcation diagram.

However, because the arc-length continuation method follows a solution branch, it is
Figure 5.6: Limitation of arc-length continuation for multiple steady-state identifications computationally expensive to identify when the branch extends beyond the range of the AIS. An illustration of such limitation is shown in Figure 5.6. The star marker denotes the initially obtained solution at an available input in the AIS. In order to obtain additional solutions at the same initial input value, the arc-length continuation solves for all steady states along the connecting solution path. Consequently, computational resources are expended in determining solutions that fall outside the interested available input regions.

The deflation method represents a numerical approach to determining multiple roots of a system of equations\cite{58}, \cite{59}, which is equivalent to a steady-state model in this context. In this method, the steady-state system is first solved, and then the model equations are deflated by dividing them by the linear factor corresponding to the solution. This method involves first finding one solution to the steady state system, and then the model equations are deflated by dividing them by the linear factor corresponding to the solution found. As a result, the newly augmented equation system reaches infinity at the initial solutions, and a nonlinear solver will converge to the other steady-state solutions for the same input value.

The deflation method is better suited for operability mapping due to its lower computational requirements. While the theoretical basis is sound, the practical application of this approach presents challenges. The case study of an adiabatic continuous stirred-tank reactor for propylene glycol production used in the feedback mapping example is considered to show the limitation of the deflation and to later demonstrate the novel multiple steady-state identification method. The chosen process has been shown to have the multi-
ple steady-state behavior. The propylene glycol production is formulated as a state-space nonlinear dynamic model in continuous time as follows.

\[
\dot{x} = F(x, u) \tag{5.10}
\]

in which the state variables in the vector \( x \) are also the output variables, and the vector \( u \) consists of process inputs, which can either be process design inputs or manipulated inputs for operations. For the propylene glycol case study, the state variables are the reaction conversion and temperature, and the input variables are the total feed concentration and the reactor volume. The steady-state model is obtained by setting the left-hand side of the differential equations in (5.10) to zero. For a fixed input vector \( u_0 \), let \( x_0 \) be the respective steady-state solution. The deflation method is applied to find the additional steady-state solution by augmenting the model equations.

\[
0 = F(x, u_0) \frac{1}{(x - x_0)^T S (x - x_0)} \tag{5.11}
\]

in which \( S \) is a positive definite scaling matrix for the deflation method.

(a) steady-state solutions of propylene glycol production using deflation method with validations

(b) steady-state solutions of propylene glycol production using homotopy continuation

Figure 5.7: Multiple steady-state identifications using deflation with poor \( S \) selection

While the system of equations in (5.11) yields the additional steady-state solution in
theory, the square inverse of the additional deflation term can lead to incorrect solutions if $K$ is not properly selected for the model. Thus, the solutions obtained using deflations are substituted in (5.10) to validate the steady-state convergences. An identity matrix is chosen as $K$ for the deflation of the propylene glycol production reactor, and the results are shown in Figures 5.7 and 5.8. In Figure 5.7, a few additional steady-state solutions are identified using deflation when compared to the homotopy continuation method. However, many of the deflated solutions are incorrect without the validation step, as shown in Figures 5.8. The primary reason for this undesirable behavior stems from the augmented deflation term increasing at a much faster rate towards infinity when compared to the approach of the model equations $F$ towards zero.

5.4.2 Divergent dynamic jump method for multiple steady-state identifications

In this work, a novel divergent dynamic jump method for multiple steady-state identifications is proposed. At fixed process inputs, the method manipulates the stability of the state-space dynamic process to find transients between different steady states. The
adiabatic propylene glycol reactor in the previous subsection is employed to demonstrate
the divergent dynamic jump method.

Recall that the input-output pair \( \{u_0, x_0\} \) is the initial steady state condition of
the given process. Using the first steady state, the nonlinear dynamic model (5.10) is modified
as follows:

\[
\dot{x} = F(x, u_0) + \frac{1}{1 + e^{-k_s ((x-x_0)^T (x-x_0) - r_d^2)}} \times \frac{A_d (x-x_0)}{\sqrt{(x-x_0)^T A_d^T A_d (x-x_0)}}
\]  

(5.12)
in which \( k_s \) is a smoothing constant, \( r_d \) is termed as the divergent radius, and \( A_d \) is a
positive definite matrix. The additional term on the existing dynamic model is denoted
as the divergent source function, and it is a vector-valued function. The first component
of the divergent source is an indicator function, which is similar to the one previously
introduced in (5.6), and it limits the effects of the divergent source to an \( n \)-ball centered
at the initial steady state \( x_0 \) with radius \( r_d \). While the first component cannot mathematically equates to zero, it asymptotically approaches zero at an exponential rate. Since
multiple steady-state analysis is often carried out in computational code packages, the
proposed indicator function can become sufficiently small for the tolerance of nonlinear
solvers to accept the results as steady-state solutions. However, the above formulation of
the indicator function includes a user-specified smoothing constant, \( k_s \), which promotes
a better transient between the divergent source and the actual dynamics of the process.
The second component is a divergent field that pushes the augmented dynamic model
toward the nearest steady-state solutions.

An important criterion of selecting \( A_d \) is the requirement of shaping the divergent
source to a matching form of the linear stability at the first steady-state solution. In
other words, if the initially obtained stable steady state is a local spiral sink, the divergent
source must be a local spiral source. This selection ensures that the dynamic solutions of
(5.12) will not fall into a Hopf bifurcation. A systematic approach of constructing \( A_d \) is
as follows. At the initial steady-state solution, obtain the gradient of the state-space rate
of change: \( \nabla_x F(x_0, u_0) \). The divergent directions are the inverse of the local stability of
the first solution while maintaining the direction: \( A_d = -\nabla_x F(x_0, u_0) \).
This section presents a comprehensive description of the divergent dynamic jump method for identifying multiple steady states, accompanied by a case study application to facilitate a clearer understanding of the methodology. The illustration of the proposed method for the CSTR is shown in Figure 5.9. In the figure, the first identified stable steady state $x_0$ is denoted with the red circle marker. The second steady state is an unstable saddle-node equilibrium, denoted by the green circle marker. The final stable unidentified steady state is denoted with the blue circle marker. The vector field in the figure corresponds to the time derivatives of the state variables. To initialize the steady-state identification process, two random feasible solutions are chosen to show that the proposed method is independent of the initial conditions, and they are denoted as the cyan and the orange pentagon markers. Additionally, the trends from the initial feasible solutions with respective colors represent the solution of the dynamic system at the two initial conditions.

From an initial condition, the closest stable steady state is identified by setting the rates of change in (5.10) to zeros and solving the system of nonlinear equations. An alternative method is simulating the dynamic process for an extended period until a steady state is met, as shown in Figure 5.9a. After the first steady state is obtained, the dynamics are augmented according to (5.12), and simulated until an augmented steady state is reached. It is important to emphasize that the augmented steady state is not a real steady state of the process. Because between two neighboring stable steady states, there must exist at least one stable steady state, the divergent jump aims to push the dynamic process in the opposing direction of the closest unstable steady state by incrementally increasing the divergent radius. Because the dynamic process is continuous, the augmented divergent source and the natural vector field divergence around the unstable steady state can only result in one of the three scenarios: an augmented stable steady state, a saddle point, or a limit cycle. The event in which the augmented dynamic system converges to a saddle point is extremely unlikely because it requires the perfect alignment of the divergent source, the unstable steady state, and their respective local dynamics. If the scenario is true, a small positive definite perturbation of the $A_d$ matrix will eliminate this issue. Additionally, the
Figure 5.9: Divergent dynamic jump method for multiple steady-state identifications at fixed inputs for propylene glycol production example

(a) First stable steady-state solution identification

(b) Divergent source added at the first steady-state solution

(c) Divergent radius passing through the saddle point steady-state solution

(d) Identification of second stable solution

proposed selection of the matrix $A_d$ ensures that a limit cycle will not occur. Thus, the augmented dynamic system will converge to a stable steady state. 

Because the effects of the divergent source are steadily increased via the control of the divergent radius and the smoothing constant $k_s$, the augmented steady-state solutions move along the transient path between the initial steady state and the unstable steady state. Figures 5.9b, 5.9c, and 5.9d are the augmented dynamic process with increasing values of the divergent radius. At the augmented steady state at each step, the solution is substituted by the process model to recognize the unstable steady state. An optional step of initializing the nonlinear equation solver with the achieved augmented steady states is implemented to find an increase in the efficiency of the divergent jump. As observed in
Figures 5.9c and 5.9d, once the augmented steady state passes the unstable steady state of the system, the dynamic process will immediately converge to the next stable steady state. At the second stable steady state, an additional divergent source can be added on top of the existing augmented dynamic model, and the next steady state can be identified.

Given the process input ranges, all steady-state solutions of the adiabatic propylene glycol reactor are found by employing the divergent jump technique. The reactor volume is limited to a range of 20 to 40 $ft^3$, and the total feed concentration is varied between 20 to 80 $lbmol/ft^3$. The steady states are plotted in Figure 5.10. Compared to the multiple steady-state identification results in Figure 5.7, which are obtained using the deflation method, the proposed divergent jump method was able to find the unstable steady-state solutions. Since the smoothing constant $k_s = 0.01$ is chosen to be a small value, the transient between the artificial divergent source and the saddle-point equilibrium is incrementally transformed. This slow transition allowed the augmented dynamics to pass through the unstable steady-state solutions.
Figure 5.10: Multiple steady-state identifications for adiabatic CSTR example
Chapter 6

Dynamic Discrepancy Reduced Order Modeling

6.1 Introduction

The state-space projection mapping method in Chapters 4 and 5 are based on the state-space structure of the dynamic model formulations. The transient dynamics of the outputs are uniquely defined by the transient behaviors of the states, and the preceding predicted states are dependent on the current states and manipulated input ranges. These characteristics are not only beneficial for simplification purposes without loss of accuracy for dynamic operability mapping, but also applicable for the dynamic discrepancy model reduction, which is proposed in the current chapter.

In the recent developments in advanced process control, nonlinear model predictive control (NMPC) is an increasingly popular control strategy used in the chemical process industry and advanced manufacturing plants. Unlike linear MPC, which is designed for linear dynamic systems, NMPC handles nonlinearities in the system, such as due to saturation effects, nonlinear dynamics, and convoluted input and output constraints. In NMPC, the nonlinear system model predicts the system future behavior and generates control actions that optimize a performance criterion over a specified prediction horizon. NMPC is often formulated as a dynamic optimization problem in which a mathemati-
The mathematical model is used as an equality constraint set, and the objective is to find the optimal control actions. A well-known challenge of implementing NMPC online is its heavy computational load, as the optimization problem must be solved in a limited time for real-time applications with fast sampling rates.

Several approaches have been proposed to address this challenge, including model reduction, algorithmic improvements, and hardware acceleration[60]. While each approach has its own advantages and disadvantages, there is no universally applicable solution. The primary approach to be considered in this work is dynamic model reduction. Specifically, this work focuses on balancing the computational complexity and model accuracy via a grey-box modeling framework. A more comprehensive high-fidelity model for control purposes results in a small plant-model mismatch, which leads to better closed-loop performance. However, such a detailed model increases the optimization problem complexity of the NMPC, and a longer time period is needed for the nonlinear programming solver to converge to an optimal solution.

The two most common mathematical model formulations for NMPC are the first-principles modeling approach and the black-box modeling approach. The first-principles modeling approach is based on a detailed understanding of the underlying physics and mechanisms of the system. This approach involves developing a mathematical model of the system based on fundamental principles, such as conservation laws, thermodynamics, and reaction kinetics. The obtained model is typically described by a set of differential equations, which can be solved numerically to predict the system behavior. In contrast, black-box modeling relies on empirical data and does not require a detailed understanding of the underlying physics or mechanisms. This approach involves developing a mathematical model of the system based on input-output data, using techniques such as system identification or machine learning. The resulting model is typically described by equations or empirical relationships, which can be used to predict the performance of the system.

The first-principles modeling and the black-box modeling approaches both have their own advantages and disadvantages. A first-principle model is generally more accurate, and it can be extrapolated to predict system behavior under different conditions, whereas
black-box models tend to have poor extrapolation capabilities. Alternatively, black-box models have superior computational efficiency, and their simplifications are more systematic due to their fixed equation structures. In the current work, the grey-box model refers to a hybrid model in which a high-fidelity model (HFM) is simplified to a reduced-order model (ROM). In the model reduction, some key characteristics of the HFM are retained based on the process understanding, and the discrepancies between the HFM and the ROM are compensated by strategically placed black-box functions.

In the current literature, discrepancy functions are often augmented to the time-varying outputs[31]. However, these output discrepancy functions are difficult to calibrate for a dynamic process because they require functional calibration inputs. To address this challenge, a dynamic discrepancy reduced-order model (DD-ROM) is developed in this work to reduce the model complexity for online control implementation. A DD-ROM structure was initially proposed to propagate uncertainty in a multiscale model[61]. While formulating a DD-ROM is typically done on a case-by-case basis due to a diverse collection of model reduction methods, a set of heuristics for constructing a DD-ROM to model predictive control is provided in this work as a contribution.

The dynamic discrepancy functions correspond to black-box functions, and the identification of the given functions is computed based on a Bayesian calibration framework. A special class of Gaussian process with Bayesian smoothing spline Analysis of Variance (BSS-ANOVA) covariance function is selected for the dynamic discrepancy because it can capture complex, nonlinear relationships among the system variables[62]. Using Karhunen–Loéve expansion, the selected Gaussian process is decomposed into a sum of basis functions for model selection to avoid overfitting. Another contribution of the proposed framework is a moving horizon estimation adaptation during data collection, which allows the dynamic discrepancy function to take the form of Bayesian linear regression for more efficient calibration and model selection while maintaining the nonlinearities in the basis functions.

Bayesian inference and modern control theory are the two fundamental aspects of the DD-ROM application to advanced control. Due to the common underlying mathematical
framework, certain nomenclature is employed in both areas despite having distinct definitions. More specifically, the ambiguity of the terminology "inputs" and "outputs" is removed as follows. In the context of control systems, manipulated inputs are the variables or parameters that are directly modified by the controller to regulate the behavior of the system. These inputs can be physical quantities such as flow rates, pressure, or valve positions. In Bayesian calibration literature, the calibration inputs, which are sometimes referred to as control inputs, are the domains of the data-driven model. In a DD-ROM, the calibration inputs are the inputs of the dynamic discrepancy functions, which are a combination of the current state variables and the manipulated inputs. In a similar manner, the controlled outputs in this work are the variables that the controller is designed to regulate in order to achieve a desired objective, and the calibration outputs are the values of the discrepancy functions that aim to counterbalance the mismatch between the ROM and the HFM.

The proposed DD-ROM formulation is applied to formulate a simplified dynamic model of a Fischer-Tropsch reactor. The Fischer-Tropsch process is a chemical reaction that converts a mixture of carbon monoxide and hydrogen, called synthesis gas or syngas, into hydrocarbon waxes such as for synthetic diesel, gasoline, and lubricants. Large-scale Fischer-Tropsch synthesis production is often carried out in a slurry bubble column reactor[63], which provides a large contact area between the catalyst pellets and the syngas, thereby increasing the conversion rate and overall efficiency of the process.

Overall, the main objective of this work is to propose a framework to construct dynamic discrepancy functions for reduced-order models employed in model predictive controllers. To achieve the objective, three major aims of the framework are: augmenting the selected ROM with discrepancy functions, generating the data set that captures the differences between the ROM and the HFM, and calibrating the discrepancy functions.
Chapter 6

Section 6.2

6.2 Background

6.2.1 System preliminaries

For the high-fidelity model, a first-principles dynamic model is assumed with sufficient knowledge of the simulated system. This model is mathematically formulated as a system of nonlinear differential equations in the following form:

\[
\dot{x}(t) = F(x(t), u(t), d(t)), \quad x(0) = x_0 \tag{6.1a}
\]

\[
y(t) = H(x(t), u(t), d(t)) \tag{6.1b}
\]

in which at any given time \( t \), \( x(t) \in \mathbb{R}^{n_x} \) is the state vector, \( \dot{x}(t) \in \mathbb{R}^{n_x} \) is the vector of time derivatives of the states, \( u(t) \in \mathbb{R}^{n_u} \) is the manipulated/input vector, \( d(t) \in \mathbb{R}^{n_d} \) is the disturbance vector, and \( y(t) \in \mathbb{R}^{n_y} \) is the controlled/output vector. The dynamics of the high-fidelity model are described with the rate of change equations \( F : \mathbb{R}^{n_x+n_u+n_d} \rightarrow \mathbb{R}^{n_x} \). The mapping \( H : \mathbb{R}^{n_x+n_u+n_d} \rightarrow \mathbb{R}^{n_y} \) are an injective function that uniquely defines the output vector according to the state vector, the manipulated vector, and the disturbance vector. It is important to note that equations (6.1a) and (6.1b) are not parts of a unified system of differential-algebraic equations, and the time evolution of the dynamic model is solved by integrating equation (6.1a).

Dynamic models can be constructed as large-scale flowsheet models to represent entire manufacturing processes or as atomic-scale computational dynamics fluid simulations to represent multi-body particle systems. Since the focus of this work is model reduction for NMPC in distributed control systems, dynamic models strictly refer to unit operation models in the following discussion. Furthermore, the considered model reduction methods can be classified into two categories: heuristic-based methods and mathematical methods. Mathematical model reduction methods are well-structured and are easily applied because they follow specified algorithms. However, these methods often ignore the model structure which leads to the loss in the physical meaning of variables[64]. On the contrary, heuristic-based methods retain the physical relationships in the model structure, but they are often
selected on a case-by-case basis.

The considered reduced-order dynamic model is a simplification of the high-fidelity model given above, and a more detailed discussion on the model reduction methods is provided in the literature[24]. A state-space representation of the reduced-order model considered is given by the following system of nonlinear differential equations:

\[
\begin{align*}
\dot{\hat{x}}(t) &= \hat{F}(\hat{x}(t), u(t), d(t)), \quad \hat{x}(0) = \hat{x}_0 \quad (6.2a) \\
\hat{y}(t) &= \hat{H}(\hat{x}(t), u(t), d(t)) \quad (6.2b)
\end{align*}
\]

in which at any given time \(t\), \(\hat{x}(t) \in \mathbb{R}^{n_{\hat{x}}}\) is the reduced-order state vector \((n_{\hat{x}} < n_x)\), \(\dot{\hat{x}}(t) \in \mathbb{R}^{n_{\hat{x}}}\) is the vector of time derivatives of the reduced-order states given by the rate of change equations \(\hat{F}\). Since the reduced-order model is intended to replace the high-fidelity model in the NMPC formulation, it admits the same input vector and disturbance vector, \(u(t)\) and \(d(t)\), respectively from equations (6.1a) and (6.1b). Additionally, the reduced-order model output vector, \(\hat{y}(t) \in \mathbb{R}^{n_y}\), must contain the same variables as the output vector, \(y(t)\), of the high-fidelity model. Similarly to the high-fidelity model, the time evolution of the reduced-order model is solved by integrating equation (6.2a), and the output \(\hat{y}(t)\) is obtained via an injective mapping \(\hat{H} : \mathbb{R}^{n_{\hat{x}} + n_u + n_d} \to \mathbb{R}^{n_y}\) after a solution of \(\hat{x}(t)\) is acquired.

The dynamic systems defined in (6.1) and (6.2) are assumed to be Lipschitz continuous. A Lipschitz continuous system has a unique solution from a fixed initial condition, and the dynamic discrepancy formulation takes advantage of this characteristic. This assumption is especially important for the dynamic discrepancy reduced-order model to be able to approximate the high-fidelity model in the following section. Furthermore, Lipschitz continuity is a reasonable assumption for physics-derived models as the state variables rarely experienced extremely stiff transitions.

95
6.2.2 Nonlinear model predictive control formulation

Modern distributed control systems have multiple layers to maximize profit and minimize risks, such as environmental impacts or equipment damage. Typically, the scheduling and real-time optimization layers determine the optimal operation strategy of the whole process every hour and provide the setpoints to the controllers in the regulatory layers. The main objective of the regulatory controllers is then to bring the subprocesses or units to the given setpoints by updating the control actions every few seconds or minutes. In this work, the high-fidelity model is used to represent the actual process, and the derived reduced-order model serves as the equality constraint of the dynamic optimization of NMPC. The closed-loop system is simulated by consecutively solving the high-fidelity model within fixed-time intervals while holding the control actions constant, and this task is carried out by a differential equation solver. In practice, NMPC is paired with a state estimator to determine the current state variables from the process measurements. Since state estimation is not the main focus of this work, full state information is assumed to be available to initialize the reduced-order model from the output measurements of the high-fidelity model at every time step. The schematic of the considered closed-loop system is illustrated in the following figure (6.1).

![Diagram of closed-loop control with reduced-order model framework](image)

Figure 6.1: Closed-loop control with reduced-order model framework

While the dynamics of the first-principles models are most accurately described in continuous time, process instrumentation and control equipment operate in the discrete-time domain, and thus the conversion between the two domains is necessary and is expressed
as follows:

\[ t_k = \Delta t \times k \quad \forall k \in \mathbb{N} \quad (6.3) \]

in which \( t \) is the continuous-time index, \( t_k \) is the discrete-time representation, and \( \delta t \) is the discretized time step. As a dynamic model can be rescaled appropriately by multiplying the rate of change equations (6.1a) and (6.2a) with a global time scaler, without loss of generality, the discretized time step is assumed to be \( \Delta t = 1 \) in the following NMPC formulation for ease of notations. This unifies the discretized time index at the \( t_k \)th time-step with the continuous-time representation at the moment \( t_k \).

For every \( t_k \), the considered NMPC is a setpoint tracking controller, and is formulated as the following constrained optimization problem:

\[
\begin{align*}
\min_{x} & \sum_{k=1}^{N} \left\{ \| \hat{y}(k) - y^{sp} \|_Q^2 + \| u(k) - u(k - 1) \|_R^2 \right\} + \| \hat{y}(N) - y^{sp} \|_P^2 \\
\text{such that} & \quad \hat{x}(k + 1) = \hat{f}(\hat{x}(k), u(k)) \\
& \quad \hat{y}(k) = \hat{h}(\hat{x}(k), u(k)) \\
& \quad \hat{x}(0) = L(x(t_k)) \\
& \quad c(x(k), u(k)) \leq 0
\end{align*}
\]  

\[ (6.4) \]

in which \( \| s \|_H^2 \) denotes the weighted sum of squares, \( s^t H s \); \( Q, R \) and \( P \) in the objective function (6.4) are positive definite matrices that respectively represent the weights of the predictive setpoint offsets, the predictive manipulated input variations and the terminal cost. The vector of setpoints given by the real-time optimizer, \( y^{sp} \), is obtained by solving a steady-state economic optimization subjected to the high-fidelity model in (6.1). Zero-order hold is applied to the manipulated inputs, \( u(k) \), in between discretized time steps, and the discrete-time equivalent of the reduced-order model (6.2) is embedded in (6.5a) and (6.5b). In the literature, there is a diverse collection of methods to explicitly discretized a system of nonlinear differential equations, such as the explicit Runge-Kutta
methods. In the studies of dynamic optimization for NMPC applications, the discretization can be simultaneously solved with the optimal control actions by combining collocations or implicit Runge-Kutta methods into the constraints of the NMPC. Thus, the formulation of the discrepancy functions of the reduced-order model must be compatible with both discretization approaches. The equality constraint (6.5c) serves as the feedback component of a closed-loop system, in which the mapping $L$ is the state estimation layer from the full state information, $x$, at the current time $t_k$. The manipulated input, state, and output constraints are incorporated into the set of inequality constraints $c$ in (6.5d).

6.3 Dynamic Discrepancy Model Reduction Framework for Advanced Process Control

Figure 6.2: Dynamic discrepancy reduced-order modeling framework

6.3.1 Dynamic discrepancy

The formulation of a reduced-order model from a high-fidelity model inevitably removes or simplifies some components of the process model representation. Thus, discrepancy functions are often added to the reduced-order model to compensate for the differences from the high-fidelity model. For a dynamic process, a universal approach is appending
the outputs from equation (6.2b) with a black-box function as follows:

\[
\begin{align*}
\dot{\hat{x}}(t) &= \hat{F}(\hat{x}(t), u(t), d(t)), \quad \hat{x}(0) = \hat{x}_0 \\
\hat{y}(t) &= \hat{H}(\hat{x}(t), u(t), d(t)) + \delta(\hat{y}(t-1), \hat{y}(t-2), ..., u(t), u(t-1), ...; \beta)
\end{align*}
\]

in which the discrepancy function, \(\delta\), in equation (6.6b) is a function of the past output measurements and the manipulated input sequence up to the current time step. Also, the vector of hyperparameters, \(\beta\), is calibrated to match the transient behaviors of \(\hat{y}(t)\) and \(y(t)\).

The first challenge of the output discrepancy formulation in (6.6b) arises from the mismatch in the time-domain inputs of \(\delta\) and \(\dot{x}\). By taking into account the dynamic behaviors of the system in the time interval preceding the current time step, the discrepancy function in (6.6b) reflects more accurately the mismatch of the reduced-order model than a discrepancy function that only considers the outputs and manipulated inputs at the exact moment. However, if \(\delta\) depends on every value of the manipulated input and the measured output sequences of the immediately preceding time intervals, then the discrepancy function is a functional instead of a function. In that case, the identification of \(\delta\) is a search in a functional space, which is more computationally expensive than a parametric calibration of the hyperparameter \(\beta\).

A solution for this challenge is considered here as a fixed and dense partition of the time interval preceding the current moment for the inputs of the discrepancy function. As a result, the rates of change in equation (6.6a) are modeled in a continuous-time domain, and the inputs of the function \(\delta\) in equation (6.6b) are in a discrete-time domain. However, this approach leads to another difficulty when applying the reduced-order model to an NMPC. Since the embedded dynamic system in (6.5) employs \(\hat{f}\) and \(\hat{h}\), which are the discretizations of \(\hat{F}\) and \(\hat{H}\) in (6.6), a transformation of \(\delta\) is needed before applying it to a closed-loop control system. Furthermore, dynamic optimization approaches to expedite the convergence of (6.5) to the optimal solutions can be inhibited by the offline transformation of \(\delta\), and the NMPC is limited to the sequential discretizing before optimizing solution methods.
An additional disadvantage of the output discrepancy formulation proposed in (6.6b) is its convoluted calibration of the hyperparameters, $\beta$. Because the discrepancy function takes the form similar to a nonlinear autoregressive exogenous (NARX) model and the number of NARX’s inputs increases with the length of the considered input horizon, a large data set is required to reflect the transient dynamic behavior represented by the discrepancy terms accurately. Additionally, since the discretization time, $\Delta t$, in (6.3) uniquely defines the calibration result of $\delta$, tuning of the NMPC to a different update frequency of the manipulated inputs can require an updated data set and a new hyperparameter’s calibration.

The dynamic discrepancy reduced-order model is formulated to address the above challenges associated with the output discrepancy dynamic model. In particular, for a Lipschitz continuous dynamic system, the Picard–Lindelöf theorem guarantees the existence and uniqueness of the solution to every initial condition and a fixed path. From this crucial property, a dynamic discrepancy reduced-order model matches the transient behaviors of a high-fidelity model by imitating the rate of changes of the state variables. Since the outputs are the projections of the state variables, the convergence of the reduced-order states leads to the convergence of the time-varying outputs to those of the high-fidelity model. Mathematically, a reduced-order model with dynamic discrepancy terms can thus be defined as follows:

$$\dot{x}(t) = \hat{F}(\hat{x}(t), u(t), d(t), \delta(\hat{x}(t), u(t); \beta)), \quad \hat{x}(0) = \hat{x}_0 \quad (6.7a)$$

$$\dot{y}(t) = \hat{H}(\hat{x}(t), u(t), d(t)) \quad (6.7b)$$

Because the dynamic discrepancy function, $\delta$, in (6.7a) only requires the current states and manipulated inputs instead of the past variables, the dimension of its domain is significantly lower than that of the output discrepancy function in (6.6b). Consequently, if both the discrepancy functions are constructed from the same model architecture, the dynamic state discrepancy function is significantly more straightforward to obtain than the output discrepancy, and it thus has a lower potential to be overfitted when using the
same data set. Additionally, the augmentation of the dynamic discrepancy function in the ROM is not limited to an external addition such as the output discrepancy, but it can also be an internal enhancement of the dynamic model.

Since there are many different methods of obtaining the reduced-order model, the construction of the discrepancy function is considered here on an ad-hoc basis. While a unified routine or algorithm to formulate $\delta$ is difficult to generalize for all dynamic processes, the following fundamental criteria are proposed as heuristics for obtaining DD-ROM of an NMPC:

1. The augmented dynamic system portion $\dot{\hat{x}}(t) = \delta(\hat{x}(t), u(t); \beta)$ must not converge to a steady-state when the manipulated inputs are fixed at all admissible values.

2. The initial conditions of the DD-ROM and the HFM must be equivalent in nature.

3. Physical and thermodynamics laws are not to be violated in the DD-ROM.

The first criterion is necessary for the DD-ROM and the HFM to achieve similar steady-state outputs from the same inputs. Because the considered NMPC is a regulatory controller, if the DD-ROM is unable to solve for the steady-state inputs associated with the given setpoint, the outputs of the closed-loop system will not converge to the desired targets. However, if the above-augmented dynamics, which admit the dynamic discrepancy functions as the rates of changes, consistently reach steady-state values for all admissible manipulated inputs, then the values of the discrepancy function will be zeros at any steady state. In other words, a self-stabilizing dynamic discrepancy function cannot compensate for the steady-state errors between the HFM and the ROM, and it only improves the transient dynamics of ROM. A particularly problematic characteristic of this system is its integrated errors of the plant-model mismatch, and the DD-ROM model requires frequent recalibration to correct these errors. For these reasons, any dynamic discrepancy functions that satisfy the above condition must be avoided. From the expressions of equations (6.1a), (6.2a) and (6.7a), it is easily mistaken that the discrepancy terms, $\delta$, equate to the differences between $F$ and $\hat{F}$. Because the HFM and the ROM are dynamically stable, any linear combinations of their rates of change will result
in a dynamic discrepancy function that directly violates the first criterion and should be avoided.

The second criterion is recommended to reduce the offsets between the DD-ROM in the controller and the HFM plant. Since the dynamics of the systems are assumed to follow state-space structures that are modeled as initial value problems in differential equations, if the initial conditions of the DD-ROM and the HFM are not the same, the dynamic performances of the DD-ROM may be precise but not accurate in the best-case scenario. The dynamic discrepancy function assists the steering of the ROM dynamic states toward the solutions of the HFM, but it does not account for the past manipulated inputs or state variables. Thus, the DD-ROM is incapable of recorrecting itself once the initial states are misplaced, and identifying a proper initial condition is as crucial as calibrating the discrepancy function. For a closed-loop system, the criterion on initial conditions is applied for the state estimation layer, \( L \), in (6.5c) using a similar rationale.

While the first two criteria focus on the numerical aspects that make DD-ROM compatible for NMPC implementation, the third criterion seeks to preserve the advantage of the physical system via protecting the physics-derived properties of the ROM. For example, mass balances can be retained by incorporating the dynamic discrepancy into the kinetic parameters of reactions instead of directly adding the dynamic discrepancy to the mass accumulation terms. While augmenting an energy balance, it is advisable to enhance thermal conductivity or heat transfer coefficient with the dynamic discrepancy functions, rather than changing the heat of reaction or specific enthalpy. Generally, the dynamic discrepancy function is prioritized to represent conversions over generations to ensure compliance with physics conservation laws.

### 6.3.2 Data collection and choice of basis functions

Data collection is critical in calibrating the dynamic discrepancy functions because it provides the necessary information to adjust the hyperparameters to fit the modeled system. The DD-ROM may not accurately represent the system without accurate and relevant data from the HFM, leading to inaccurate predictions or suboptimal solutions.
of the NMPC. The illustration of the data collection procedure is proposed in Figure 6.3, which is also suitable when an actual plant is used in place of the HFM. Because the discrepancy function, $\delta$, has the domains containing $\dot{x}(t)$ and $u(t)$, the calibration inputs are the states and the manipulated inputs of the ROM. The calibration outputs are the values of $\delta$ that minimize the discrepancy between the ROM’s predictions and the observed data.

![Figure 6.3: Calibration data collection method for dynamic discrepancy functions](image)

A pseudorandom number generator, such as a pseudorandom binary sequence (PRBS), is utilized to achieve a set of manipulated inputs, and system identification techniques can be applied to the design of these input signals to excite the dynamic process properly. If the time to reach new steady states of the dynamic process is available, the holding period of the random manipulated input signal is chosen to be longer than this time. Because the dynamic discrepancy function cannot distinguish between a steady state and a transient state, choosing the period as such ensures the collected data characterize both scenarios. Additionally, the repetition of steady states in a calibration data set can act as a bias weight, meaning that it can skew the estimation of model parameters towards such particular operating points. This generated bias can be advantageous for NMPC applications as accurately reaching the setpoints lead to the desired outputs, which is more important than finding the optimal transient path.
During model calibration, the HFM and ROM are simulated with the same manipulated input sequence, and the discrepancy between their outputs is evaluated. By using the same manipulated input sequence for both models, the calibration process can effectively account for any differences in the system’s response that may arise due to a mismatch in model structure or complexity. The DD-ROM hyperparameters are then adjusted to minimize the discrepancy using an optimization algorithm such as in parameter estimation. A comprehensive review of the current state-of-the-art state and parameter estimation algorithms is provided in the literature [65], [66]. After fixing the manipulated input sequences, a moving horizon parameter estimation algorithm is employed to obtain the discrepancy values for the calibration output set and the reduced-order state variables for the calibration input set. Since the HFM model is independent of the calibration process, its output sequence resulting from the previously generated random signal is recorded as a reference for the moving horizon data generation for obtaining the DD-ROM. For each value of the manipulated input, the following dynamic programming problem is solved for each data point.

\[
(\hat{x}(t_k + \Delta t), \delta(t_k)) = \arg\min_{\delta} \int_{t_k}^{t_k + M} \|y(\tau) - \hat{y}(\tau)\|^2 d\tau \tag{6.8}
\]

such that

\[
\dot{\hat{x}}(\tau) = \hat{F}(\hat{x}(\tau), u(\tau), d(\tau), \delta(\tau)) \tag{6.9a}
\]
\[
y(\tau) = \hat{H}(\hat{x}(\tau), u(\tau), d(\tau)) \tag{6.9b}
\]
\[
\hat{c}(\hat{x}(\tau), u(\tau), \delta(\tau)) \leq 0 \tag{6.9c}
\]

In the parameter estimation problem formulated in (6.8) and (6.9), the dynamic discrepancy terms are considered as independent variables instead of functions as in (6.7a). Thus, their optimal values in the above estimation problem are used as calibration inputs of the data set for the following step. The initial values of the DD-ROM are fixed as provided in (6.7a), and included as a hard constraint in the first estimation problem associated with the first data point. Since the dynamic discrepancy function is added to the reduced-order
model rate of changes, the reduced state variables at the following discretized time step, \( \hat{x}(t_k + \Delta t) \), depends on the value of the \( \delta \) at the considered time \( t_k \). Thus, the solution of the state vector is used as an initial condition constraint in the following estimation problem, and they are also listed as calibration inputs.

The estimation horizon, \( M \), in (6.8) is selected to be approximately the same as the time to reach a new steady state of the true dynamic process because any external perturbation to the state variables beyond this horizon has negligible effects on the states and the discrepancy terms. Furthermore, the selection of the horizon length is not affected by the size of the data set, thus maintaining the scale of the optimization problem in (6.8) and (6.9) even for large data sets, which are required by high-dimensional calibration of input-output systems.

The inequality constraints in (6.9) include the physical constraints such as the non-negative conditions for molar concentration and choked flow limits for gases. Thus, formulating the calibration data collection as a nonlinear programming problem provides an additional benefit of filtering impractical data points. These constraints also prevent the dynamic process from operating in multiple steady-state regions. Because the dynamic discrepancy function is formulated as an explicit instead of an implicit function, each calibration input set only generates one set of calibration outputs. Output multiplicity behaviors, in which one set of calibration inputs can lead to multiple calibration outputs, leads to the precision of the calibration being significantly diminished.

The moving horizon parameter estimation in (6.8) and (6.9) is defined in the continuous-time domain. However, this formulation does not invalidate the conversion of the DD-ROM to a discrete-time model while solving the nonlinear programming problem. Instead, it emphasizes the significance of transforming the dynamic discrepancy functions. If a dynamic optimization algorithm with embedded discretization, such as in the collocation method, is used to solve the estimation problem, its optimal solutions immediately contain the values of the calibration inputs. If a time discretization is performed explicitly on (6.8) and (6.9) before a nonlinear optimizer solves the estimation problem, the optimal solutions would require an inverse transformation to achieve the correct dynamic
discrepancy values in continuous-time for the calibration inputs.

A special case of the ROM formulation is truncation, in which the key characteristics portion of the HFM is retained based on actual knowledge of the process. The remaining portion is removed or simplified based on justifiable assumptions. In this case, the optimal solution of the discrepancy terms and the reduced-order state in the above estimation problem equal their respective values evaluated using the curtailed portion of the HFM. In more general cases that do not employ truncations, the proposed data collection method using a moving horizon estimation is compatible with all other model reduction methods.

When it comes to the formulation of the dynamic discrepancy functions, there are various methodologies to do so as long as the following requirements are fulfilled. Firstly, the chosen black-box function must span the solution space of the original HFM or the actual plant, which means the function can approximate any reasonably smooth operation. Secondly, the identification or calibration of such functions for the dynamic discrepancy terms within the DD-ROM must also not be hindered as a result of their inherent characteristics. There is a direct impact of the selection of the data-driven functions on the accuracy, computational efficiency, and calibration of the DD-ROM. Some common choices of black-box models are polynomials, Fourier series, neural networks, and Gaussian processes. While many of these models have the potential to be chosen for the DD-ROM, each class of model has its advantages and disadvantages to be considered. For example, a polynomial basis model can theoretically approximate any smooth function, and its expressions are simple enough to decrease the computational time of the NMPC. However, high-order polynomials approach infinity rapidly, making them numerically indistinguishable, so the calibration of polynomial basis functions is less accurate with increasing terms.

In the following section, the dynamic discrepancy is assumed to be a Gaussian process with a Bayesian Smoothing Spline Analysis of Variance covariance function. The BSS-ANOVA framework has been shown to offer several advantages over the traditional Gaussian Process approach[67]. These advantages include the ability to handle categorical parameters in a flexible manner, as well as the capacity to model correlated outputs. Additionally, the BSS-ANOVA framework exhibits a linear computational complexity in the
number of observations made by the simulator, thus further enhancing its computational efficiency[68]. Although the BSS-ANOVA is a non-parametric Gaussian process, it can be decomposed into orthogonal basis functions using Karhunen–Loéve expansion[61]. Let $\epsilon$ be the uncertainty introduced by the parameter estimation during the data collection process; the dynamic discrepancy function is an infinite sum of basis functions given by the expansion as follows:

$$\delta(s) = \beta_0 + \sum_{j=1}^{L_3} \beta_j \phi_j(s) + \sum_{j=1}^{L_3} \sum_{k=j+1}^{L_3} \beta_{j,k} \phi_j(s) \phi_k(s) + \ldots + \epsilon$$

(6.10)
in which $\beta_j$ and $\beta_{j,k}$ are respectively the scalar coefficients of the main effect functions and the two-way interaction functions. The calibration input, $s$, denotes the concatenation of the vector of reduced-order state variables, $\hat{x}$, and the vector of manipulated inputs, $u$. Since the dynamic discrepancy values only depend on the calibration input at the same time instance, the data at each discretized time step is a pair of calibration inputs-outputs, and the time notation is dropped in the black-box model identification. While three-way or higher-order interactions can be included in the expression (6.10), their effects are assumed negligible due to their infinitesimal numerical values. In the Karhunen–Loéve expansion, the domain of each basis function is fixed to the interval $[0, 1]$ to avoid bias, and the calibration inputs are scaled to match this range with a bijective linear mapping. The covariance function of the BSS-ANOVA is additive in nature, in which the first two kernels are the first and second-order Bernoulli polynomials. The third kernel is a zero mean Gaussian process with the kernel: $K(s, t) = -B_4(|s - t|)/4!$, so the number of non-degenerate eigenvalue-eigenfunctions is infinite. However, it is possible to estimate the eigenpairs by discretizing the domain of the covariance function and determining the associated eigenvalues and eigenvectors. The resulting approximations of the third and higher basis functions are illustrated in Figure 6.4.

While the proof that the Karhunen–Loéve expansion of the BSS-ANOVA on the chosen interval is equal to the summation of trigonometric eigenfunctions is beyond the scope of this paper, the simulation results of the first 100 basis functions agreed with the conjecture.
Additionally, the high-order interactions between the basis functions are shown to be the product of individual main effect eigenfunctions[61]. For the decomposed expression of $\delta(s)$ in 6.10 to converge pointwise to a non-parametric Gaussian process, the number of considered basis, $L_\delta$, must approach infinity. Because the eigenfunctions are observed to be a set of trigonometric functions with increasing frequency and decreasing amplitude, the high-order basis functions are negligible, and $L_\delta$ is cautiously chosen as a sufficiently large number. Since the model selection procedure does not distinguish between the main effects and cross-interaction basis functions, a change of index is introduced to simplify the notation by considering all admissible basis as a tuple. The expression in (6.10) is reformulated as follows:

$$\delta(s) = \beta_0 \phi_0(s) + \beta_1 \phi_1(s) + \beta_2 \phi_2(s) + \ldots + \epsilon$$  \hspace{1cm} (6.11)$$

The discrepancy model calibration in the following section is a Bayesian inference technique. Although this method can flexibly find the posterior distributions of the hyperparameters without the proposed moving horizon estimation, it requires a very intricate likelihood function. In practice, a closed-form expression of the likelihood function of a nonlinear model is unlikely to exist, and it is commonly approximated by a Monte-
Carlo simulation\cite{69}, an Approximate Bayesian computation (ABC)\cite{70}, or a kernel function\cite{62}. Monte-Carlo simulations and ABC methods are computationally expensive, so they may not be suitable for large-scale systems. The kernel function typically requires an additional layer of Bayesian inference on its hyperparameters, which makes the calibration of the posterior of $\beta$ more complex. However, the kernel is fundamentally a way of quantifying the differences between the data and the expected behaviors, such as the integrated square error in the objective (6.8) of the proposed moving horizon estimation. Thus, the proposed approach is a hybrid that benefits from the robustness of numerical optimizations in the data collection while preserving the flexibility of Bayesian inference in identifying hyperparameters and model selection. Additionally, the estimation layer separates the vector of dynamic discrepancy values into independent calibration problems. The correlations between different dynamic discrepancy values and the DD-ROM are indirectly embedded in the optimality of the estimation problem. Therefore, instead of calibrating all discrepancy functions under the same posterior function, the calibration of each term can be done independently and in parallel with each other.

6.4 Calibrating the Dynamic Discrepancy Terms of the Reduced-Order Model

6.4.1 Bayesian model selection

Bayesian calibration and least-squares estimation are two different approaches used to estimate model parameters from observed data. While neither approach is inherently better than the other, Bayesian calibration can offer certain advantages over least-squares estimation in certain situations. Bayesian calibration has the advantage of incorporating prior knowledge or beliefs about parameters into the estimation of parameters, thereby helping to reduce uncertainty. This is particularly useful when the available data is limited or noisy, as the prior information can provide additional constraints on the parameter values. By computing the posterior distributions of the parameters, Bayesian calibration
can also establish a framework by which the uncertainty associated with parameter estimates can be quantified. As a result, it is possible to select a mode with a greater degree of confidence, as a range of parameter values and their associated probabilities can be provided. The Bayes’ Theorem is stated as follows:

\[ \pi(A \mid B) = \frac{\pi(B \mid A)\pi(A)}{\pi(B)} \]  

(6.12)

in which \( \pi(A \mid B) \) denotes the probability density function of set of variables \( A \) given fixed values of a set of variables \( B \) and vice versa. The right-hand side of (6.12) is often referred to as the posterior density function. The likelihood function is the conditional density function \( \pi(B \mid A) \), and the prior is the probability density \( \pi(B) \). Since the posterior density function is required to be integrated to one, the denominator \( \pi(B) \) is the normalization constant of the numerator over the domain of the set \( B \). In a Markov chain Monte Carlo method, which is later used to sample the probability of discrepancy model over the given calibration data set, only the product of prior and the likelihood is required. Thus, an expression of the posterior density function that is not normalized is sufficient to calibrate a black-box model.

Aside from the advantages discussed above, Bayes’ Theorem is powerful when inverting the quantification of conditional events. If the probability of the outputs given a predetermined model is available, the probability of a model given the calibration data set can be evaluated. From a process systems engineering perspective, the posterior density for models is interpreted as a measure that simultaneously quantifies the model fitness to data and its structural simplicity. Furthermore, Bayes’ Theorem can be applied in a nested hierarchical structure to provide better statistical characteristics of a model. This multilevel Bayesian inference allows the assessment of the nested conditional statements, such as when the expected calibration output depends on the hyperparameters and the hyperparameters depend on the selected basis functions.

From the data generated using the procedure in the previous section, let \( n \) be the number of data points, \( m \) be the number of basis functions considered for each calibration input, and \( r \) be the number of calibration inputs. Consequently, the \( i^{th} \) calibration inputs
of the $j^{th}$ data point is denoted $s_{i,j}$. The matrix of calibration inputs, $X$, is constructed as follows.

\[
X = \begin{bmatrix}
1 & \phi_1(s_{1,1}) & \phi_2(s_{1,1}) & \cdots & \phi_m(s_{1,1}) & \phi_1(s_{2,1}) & \phi_2(s_{2,1}) & \cdots & \phi_m(s_{r,1}) \\
1 & \phi_1(s_{1,2}) & \phi_2(s_{1,2}) & \cdots & \phi_m(s_{1,2}) & \phi_1(s_{2,2}) & \phi_2(s_{2,2}) & \cdots & \phi_m(s_{r,2}) \\
1 & \phi_1(s_{1,3}) & \phi_2(s_{1,3}) & \cdots & \phi_m(s_{1,3}) & \phi_1(s_{2,3}) & \phi_2(s_{2,3}) & \cdots & \phi_m(s_{r,3}) \\
\vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\
1 & \phi_1(s_{1,n}) & \phi_2(s_{1,n}) & \cdots & \phi_L(s_{1,n}) & \phi_1(s_{2,n}) & \phi_2(s_{2,n}) & \cdots & \phi_m(s_{r,n})
\end{bmatrix}
\]

(6.13)

in which the first column of ones is added to the calibration inputs to represent the bias of the chosen black-box model and the generated data. Since each basis function of each calibration input can either be added to or removed from the dynamic discrepancy function, the number of possible models for calibration is $2^{m \times r + 1}$. An effective method of enumerating every possible model is using a vector of indicators $\gamma = [\hat{\gamma}_0 \hat{\gamma}_1 \hat{\gamma}_2 \ldots \hat{\gamma}_{m \times r}]^T$, in which each $\hat{\gamma}_i \in \{0, 1\}$. Additionally, let the number of selected basis in a model $\gamma$ be $n_\gamma$, which is also the sum of all indicators in $\gamma$. The dynamic discrepancy function is reformulated as the following expression:

\[
\delta = \hat{\gamma}_0 \beta_0 X_0 + \hat{\gamma}_1 \beta_1 X_1 + \hat{\gamma}_2 \beta_2 X_2 + \cdots + \hat{\gamma}_{m \times r} \beta_{m \times r} X_{m \times r} + \epsilon
\]

(6.14)

If, and only if, all the components in $\hat{\gamma}_i = 1$, then the basis function of the calibration input corresponding to the $i^{th}$ column of the matrix of calibration inputs, $X$, is included in the model associated with $\gamma$. For each uniquely defined model $\gamma$, the respective vector of hyperparameters $\beta_\gamma$ is a subdivision of $\beta$ that only contains the $i^{th}$ components, $\beta_i$, if $\hat{\gamma}_i = 1$. Similarly, the truncated matrix of calibration inputs, $X_\gamma$, is the matrix that only contains the $i^{th}$ column, $X_i$, if $\hat{\gamma}_i = 1$. The calibration error is assumed to be a zero-mean independent and identically distributed Gaussian random vector with variance $\epsilon^2$. Let $N(\mu, \Sigma)$ be the Gaussian distribution with a mean vector $\mu$ and a covariance positive definite matrix $\Sigma$, and let $I$ denote an identity matrix with appropriate dimensions.

The objective in this subsection is to construct the overall posterior probability,
\( \pi(\gamma | X, \delta) \), of any model \( \gamma \) given calibration data \( X \) and \( \delta \). From a process systems engineering perspective, the posterior probability of a model can be interpreted as a measure of the extent to which the dynamic discrepancy function fits the performance of the HFM instead of a statistical probability. Furthermore, the probability of a model is also closely related to the concept of model complexity, which measures the number of basis functions or degrees of freedom in the model. By balancing the complexity of the model with the fit to the data, the probability of a model provides a way to avoid overfitting and identifies accurate and computationally inexpensive models. This balance is achieved via constructing the probability of the model using a Bayesian framework.

In the following Bayesian model selection, non-informative priors are selected to avoid subjective bias and to place an equal chance for all possible models before the calibration data is incorporated. In particular, Zeller’s non-informative G-prior[71] is selected to introduce a penalty term that encourages simpler models. The hyperparameters are represented by a Gaussian distribution with a mean of zero and independent and identically distributed variances that are proportional to the inverse of the sample size, multiplied by a constant called the g-prior. The g-prior is denoted as \( c \) in (6.15) and is set to a value between 0 and 1, with smaller values indicating less informative priors. The Jeffrey prior is used in (6.16) for the variances of moving horizon estimation errors since it remains unchanged when the model parameters are transformed[72].

\[
\begin{align*}
\beta_\gamma | \gamma, \varepsilon^2, X, c & \sim N \left( 0_{\gamma}, c\varepsilon^2 \left( X_\gamma X_\gamma^T \right)^{-1} \right) \quad (6.15) \\
\pi(\varepsilon^2) & = \pi(\varepsilon^2 | \gamma, X, c) \propto \varepsilon^{-2} \quad (6.16)
\end{align*}
\]

A diffuse prior on \( c \) is introduced in (6.17), in which \( \mathbb{I}_{N^+} \) is an indicator function of the set of positive natural numbers, and a uniform prior (6.18) is imposed on all possible model \( \gamma \). Because the random variables \( \varepsilon^2, c, \) and \( \gamma \) are independent of all other parameters of the discrepancy function, their conditional probabilities are equal to their respective marginal
probabilities as shown in (6.16)-(6.18).

\[
\pi(c) = \pi(c \mid \gamma, X) \propto c^{-1}I_{\mathbb{N}^+}(c) 
\]

(6.17)

\[
\pi(\gamma) = \pi(\gamma \mid X, c) = 2^{-m \times r - 1} 
\]

(6.18)

From the construction of \(\delta\) in (6.14), the conditional probability of the dynamic discrepancy term at a fixed model structure and other parameters can be reconstructed from (6.15). Specifically, it is a linear transformation of the \(\beta_\gamma\) via the mapping \(x_\gamma\), which results in the Gaussian distribution in (6.19), and adding with a zero-mean Gaussian of errors \(\epsilon \sim N(0, \epsilon^2)\) to achieve (6.20).

\[
X_\gamma \beta_\gamma \mid \gamma, \epsilon^2, X, c \sim N\left(0, \epsilon^2 X_\gamma \left(X_\gamma^\top X_\gamma\right)^{-1} X_\gamma^\top\right) 
\]

(6.19)

\[
\delta \mid \gamma, \epsilon^2, X, c \sim N\left(0, \epsilon^2 \left(I + cX_\gamma \left(X_\gamma^\top X_\gamma\right)^{-1} X_\gamma^\top\right)\right) 
\]

(6.20)

The dependency of the conditional probability of \(\delta\) on the error variance \(\epsilon^2\) can sum up to a marginal conditional probability by integrating the density function over the domain of \(\epsilon^2\). The closed-form expression of the probability density functions of \(\pi(\delta \mid \gamma, \epsilon^2, X, c)\) and \(\pi(\epsilon^2 \mid \gamma, X, c)\) are respectively given by the Gaussian prior (6.20) and the Jeffrey’s prior (6.16). Since the search for the most probable model in the current work is a Markov chain Monte Carlo procedure, a proportion of the model evidence is sufficient for a sampling algorithm. Therefore, the constant factor of the likelihood function in (6.21) is simplified for more efficient computation.

\[
\pi(\delta \mid \gamma, X, c) = \int_0^\infty \pi(\delta \mid \gamma, \epsilon^2, X, c) \pi(\epsilon^2 \mid \gamma, X, c) \, d\epsilon^2
\]

\[
= \int_0^\infty \frac{(\epsilon^2)^{-\frac{n}{2} - 1}}{(2\pi)^{\frac{n}{2}} (c + 1)^{-\frac{n\gamma + 1}{2}}} \exp \left( -\frac{1}{2\epsilon^2} y^\top \left(I + cX_\gamma \left(X_\gamma^\top X_\gamma\right)^{-1} X_\gamma^\top\right)^{-1} y \right) \, d\epsilon^2
\]

(6.21)

\[
= \pi^{-\frac{n}{2}} \Gamma \left(\frac{n}{2}\right) (c + 1)^{-\frac{n\gamma + 1}{2}} \left(y^\top \left(I - \frac{c}{c + 1}X_\gamma \left(X_\gamma^\top X_\gamma\right)^{-1} X_\gamma^\top\right) y\right)^{-\frac{n}{2}}
\]

\[
\propto (c + 1)^{-\frac{n\gamma + 1}{2}} \left(y^\top \left(I - \frac{c}{c + 1}X_\gamma \left(X_\gamma^\top X_\gamma\right)^{-1} X_\gamma^\top\right) y\right)^{-\frac{n}{2}}
\]
In the integration over the g-prior to achieve the marginal probability of an output set given a fixed set of basis functions and calibration inputs, the distribution of the prior in (6.18) and the obtained conditional distribution in (6.21) are employed. The indicator function $I_{N^+}$ convert the Lebesgue integral in (6.22) to a summation.

$$
\pi(\delta \mid \gamma, X) = \int_0^\infty \pi(\delta \mid \gamma, X, c)\pi(c \mid \gamma, X)dc = \sum_{c=1}^\infty \pi(\delta \mid \gamma, X, c)c^{-1}
\propto \sum_{c=1}^\infty c^{-1}(c + 1)^{-\frac{n_{\gamma} + 1}{2}} \left( y^T \left( I - \frac{c}{c + 1}X_{\gamma}X_{\gamma}^\top X_{\gamma}^\top \right) y \right)^{-\frac{n}{2}} \tag{6.22}
$$

The probability of the model given the calibration data is obtained from the probability of the generated data given a fixed model by applying Bayes’ theorem. As a result of the uniform prior of the basis structure, the posterior probability of the model is proportional to the likelihood of achieving dynamic discrepancy values.

$$
\pi(\gamma \mid \delta, X) \propto \pi(\delta \mid \gamma, X)\pi(\gamma \mid X) \propto \pi(\delta \mid \gamma, X) \tag{6.23}
$$

The expansion of the summation in (6.22) is truncated to the first $L_{\gamma}$ terms, and the choice of $L_{\gamma}$ is recommended to be a large number, such as 50. While the proof of convergence is outside the focus of the current work, computational experiments performed confirm the property with scaled data of the BSS-ANOVA Gaussian process for $L_{\gamma}$ greater than or equal to 50. The approximation of the unnormalized model evidence, $\hat{\pi}(\gamma \mid \delta, X)$, given in (6.25) is employed in the following Gibbs sampling model selection algorithm.

$$
\pi(\gamma \mid \delta, X) \propto \hat{\pi}(\gamma \mid \delta, X) \tag{6.24}
$$

$$
\hat{\pi}(\gamma \mid \delta, X) = \sum_{c=1}^{L_{\gamma}} c^{-1}(c + 1)^{-\frac{n_{\gamma} + 1}{2}} \left( y^T \left( I - \frac{c}{c + 1}X_{\gamma}X_{\gamma}^\top X_{\gamma}^\top \right) y \right)^{-\frac{n}{2}} \tag{6.25}
$$

As the number of calibration data points, $n$, increases, the numerical values of the posterior model probability decrease due to the negative exponent $n/2$. While the Bayesian inference is unaffected since the relative quantities between different probabilities are
consistent, floating point error may influence the computation of their values. To avoid this issue, the evaluation of (6.25) is recommended to be used in conjunction with a log transformation to identify the basis and the orders of magnitude.

### 6.4.2 Gibbs sampling procedure

The unnormalized posterior probability of a model, $\hat{\pi}(\gamma \mid \delta, X)$, is a combined measure of the data fitness and the model simplicity. In fact, a least-squares regression formula will appear if the base of the negative exponent $n/2$ is expanded. The negative exponent $\frac{n_s+1}{2}$ penalizes the model with more basis functions if an identical calibration output performance is obtained from two different models. The overall hierarchy of the model evidence strives to achieve different balances between the given calibration data uncertainty and span of the chosen basis functions and between the prior beliefs and the provided data.

If the model evidence is calculated for each dynamic discrepancy model $\gamma$, the model with the highest value of posterior probability model is the model that contains a sufficient amount of basis functions with the least chance of being overfitted. However, the number of possible models grows exponentially with the number of basis functions and calibration inputs. Some potential solutions for identifying the best subset of basis are using the posterior probability in (6.25) as the objective function for a mixed-integer nonlinear programming problem with the vector of independent variables being $\gamma$, or a derivative-free mixed-integer solver such as a genetic algorithm to find the most probable model. By exploiting the probabilistic nature of the model, a Gibbs sampling algorithm is utilized here to iteratively generate a sequence of candidate models, which converge rapidly in distribution to the model evidence.

The Markov chain Monte Carlo method is a class of algorithms used to generate samples from complex probability distributions that are difficult to evaluate analytically or sample easily using other techniques. Markov chain Monte Carlo involves constructing a Markov chain with the desired probability distribution as its equilibrium distribution. If the Markov chain is simulated for a sufficiently large sample size, it will eventually converge to its equilibrium distribution, and samples obtained from the chain can be used
to approximate the distribution desired. Gibbs sampling is a specific type of Markov chain Monte Carlo algorithm that is particularly efficient for sampling high-dimensional distributions. In Gibbs sampling, each variable in the target distribution is updated sequentially, conditioned on the current values of the other variables. This conditional update can be based on either the joint distribution of all variables or just a subset. By iteratively updating each variable, the Markov chain generated by Gibbs sampling converges to the desired complex and high-dimensional distribution by generating samples in low-dimensional space.

In Gibbs sampling, an initial sample is selected from the target distribution, and the rate of convergence of the algorithm is dependent on the choice of the initial sample. In the case of a poorly chosen initial sample, the Gibbs sampler may not be able to reach a stationary distribution without a significant number of iterations. Conversely, if the initial sample is selected well, it may be possible for the Gibbs sampler to converge more quickly. There are several ways to select an initial sample for the Gibbs sampler. One common approach is to use a random sample from the prior distribution. This approach is straightforward but may not always lead to an acceptable initial sample. Another approach is to use a preliminary estimation of the posterior distribution, such as the maximum likelihood estimate or the method of moments. This approach can provide an adequate initial sample but may require additional computation. It is generally good practice to try several different initial samples and check their convergence properties to ensure the algorithm converges efficiently.

A detailed description of the steps involved in the Gibbs sampling procedure can be found in Algorithm 1. The inputs of this algorithm are the matrix of calibration inputs provided by (6.13), the calibration outputs $\delta$ generated in the previous subsection, the number of samples $N_{\text{Sample}}$ generated using the Monte Carlo simulation, and the burn-in period length $N_{\text{Burn}}$. The output of the Gibbs sampling is a sequence of models $\{\gamma_0, \gamma_1, \gamma_2, \ldots, \gamma_{N_{\text{Sample}}}\}$ that are drawn from the probability $\hat{\pi}(\gamma | \delta, X)$. For each model, $\gamma_i = [\gamma_{i,0} \gamma_{i,1} \gamma_{i,2} \cdots \gamma_{i,m \times r + 1}]^T$, in the sampling sequence, the indicator $\gamma_{i,j} \in 0, 1$ denotes the inclusion of the $j^{th}$ basis function in the dynamic discrepancy function associated with
Algorithm 1 Model Selection via Gibbs Sampling

1: Initialize $\gamma_0 = [1 \ 1 \ldots 1]^T$
2: for $i = 1$ to $N_{\text{Sample}}$ do
3: \[ \hat{\gamma} = [\gamma_0 \ \hat{\gamma}_1 \ \hat{\gamma}_2 \ldots \hat{\gamma}_{m \times r}]^T = \gamma_{i-1} \]
4: \[ \text{for } k = 0 \text{ to } m \times r \text{ do} \]
5: \[ \tilde{\gamma}_0 = [\hat{\gamma}_0 \ \hat{\gamma}_1 \ \hat{\gamma}_2 \ldots \hat{\gamma}_{k-1} \ 0 \ \hat{\gamma}_{k+1} \ldots \hat{\gamma}_{m \times r}]^T \]
6: \[ \tilde{\gamma}_1 = [\hat{\gamma}_0 \ \hat{\gamma}_1 \ \hat{\gamma}_2 \ldots \hat{\gamma}_{k-1} \ 1 \ \hat{\gamma}_{k+1} \ldots \hat{\gamma}_{m \times r}]^T \]
7: Evaluate $\hat{\pi}(\hat{\gamma}_0 | \delta, X)$ and $\hat{\pi}(\hat{\gamma}_1 | \delta, X)$ using equation (6.25)
8: Calculate the acceptance probability $A = \frac{\hat{\pi}(\tilde{\gamma}_1 | \delta, X)}{\hat{\pi}(\tilde{\gamma}_0 | \delta, X) + \hat{\pi}(\tilde{\gamma}_1 | \delta, X)}$
9: Generate a uniform random number $u \in [0, 1]$
10: if $u \leq A$ then
11: \[ \text{Update } \hat{\gamma}_k = 1 \]
12: else
13: \[ \text{Update } \hat{\gamma}_k = 0 \]
14: end if
15: end for
16: $\gamma_i = [\gamma_{i,0} \ \gamma_{i,1} \ \gamma_{i,2} \ldots \gamma_{i,m \times r}]^T = \hat{\gamma}$
17: end for

Using Gibbs sampling, the most likely model is the one that appears with the highest frequency within a sequence of samples generated by the algorithm. To determine the most probable model from the sequence of samples generated, an additional counting algorithm is required, which can be computationally intensive and time-consuming. An alternative solution is computing the probability of inclusion $P(\hat{\gamma}_k | \delta, X)$, which is the chance of the $k^{th}$ basis function, $\hat{\gamma}_k$, in the most probable model. Mathematically, the probability of inclusion is defined as follows:

$$P(\hat{\gamma}_k | \delta, X) = \frac{\sum_{j=N_{\text{Burn}}}^{N_{\text{Sample}}} \gamma_{j,k}}{N_{\text{Sample}} - N_{\text{Burn}}}$$

Some common choices of the threshold for the probability of inclusion are the mean and the median of the probability of inclusion, in which a basis function is considered not important and excluded from the model if its probability of inclusion is below a certain threshold. Furthermore, the probability of inclusion indirectly reflects the quality of the calibration data set as well as the span of the considered basis functions. For instance, if both the calibration data and the considered basis functions are sufficient to recreate the expected performances of the dynamic discrepancy function, there is a clear distinction.
between the probability of inclusion associated with the basis that should be included and excluded from the most probable model. In contrast, if all probability of inclusions are consistently lower than 50%, the Gibbs sampling sequence is not sufficiently long or the basis functions cannot mimic the dynamic behaviors reflected in the data set.

6.5 Case Study: Fischer-Tropsch Synthesis Slurry Bubble Column Reactor

6.5.1 High-fidelity dynamic model development

The slurry bubble column reactor is assumed to be operated in the churn-turbulent regime to achieve optimal conversion for Fischer Tropsch synthesis[73]. In this regime, small bubbles combine into large bubbles and rise up the column in a plug-flow manner at high superficial velocities (in the range of 1-2 m/s). The large bubbles also churn up the slurry phase, allowing the well-mixed assumption to be applicable. The slurry phase contains catalyst particles and small bubbles (1-7 mm in diameter) submerged in paraffin oil, which can be recycled from the product stream of the Fischer-Tropsch synthesis. Vapor-liquid equilibrium is assumed to occur between the liquid paraffin and the gas bubbles, regardless of size. In order to maintain the optimal temperature for the Fischer-Tropsch synthesis reaction, cooling fins are placed internally in the column. The schematic of the developed model is illustrated in Figure 6.5.

The dynamic model of the slurry bubble column reactor is divided into three phases, and each phase has a separate set of equations to represent its mass balances, energy balances and thermodynamic properties. The first phase represents the large bubbles rising from the injectors at the bottom of the column. Since these bubbles have a significantly higher rise velocity when compared to the other phases, the large bubble phase is modeled as a plug flow. Furthermore, since the large bubbles also agitate the slurry and the small bubbles, they act as a stirrer and the remaining two phases are assumed to be well-mixed. More specifically, the slurry phase and the small bubble phases are
Figure 6.5: Schematic of the slurry bubble column for the Fischer-Tropsch synthesis modeled as two continuous stirred tank reactors. A diagram of the modelling approach of the Fischer-Tropsch reactor is shown in Figure 6.6.

The catalyst is assumed to be a cobalt base with an alumina (Al₂O₃) promoter support. This catalyst increases the reaction rate and selectivity of hydrogenation while reduces the water gas shift reaction to negligible levels. The kinetic of syngas gas conversion is assumed to be Langmuir-Hinshelwood type and the mass transfer effect is ignored according to the literature[74]:

\[-R_{CO+H_2} = \frac{a \times p_{H_2} \times p_{CO}}{(1 + b \times p_{CO})^2}\] (6.27)

\[a = 8.8533 \times 10^3 \times \exp \left[4494.31 \left(\frac{1}{493.15} - \frac{1}{T}\right)\right]\] (6.28)

\[b = 2.226 \times \exp \left[-8236 \left(\frac{1}{493.15} - \frac{1}{T}\right)\right]\] (6.29)

in which \(R_{CO+H_2}\) is the syngas conversion rate, \(a\) and \(b\) are the kinetic parameters, \(p_{CO}\) and \(p_{H_2}\) are respectively the partial pressure of carbon monoxide and hydrogen, and \(T\) is the temperature. The hydrodynamics of the bubbles in the column are obtained through experimental data fitting for the churn-turbulent regime[75]. The relationships between the gas holdup, \(\epsilon\), large bubble holdup, \(\epsilon_b\), small bubble holdup, \(\epsilon_{df}\), liquid holdup, \(\epsilon_L\),
Figure 6.6: High-fidelity modeling approach for the Fischer-Tropsch synthesis slurry bubble column reactor

and catalyst holdup, $\epsilon_s$, can be determined as follows:

$$\epsilon_L + \epsilon + \epsilon_s = 1 \quad (6.30)$$

$$\epsilon = \epsilon_b + (1 - \epsilon_b) \epsilon_{df} \quad (6.31)$$

The large bubble holdup is affected by the column diameter, $D_T$, and the superficial gas velocity through the large bubbles, $(U - U_{df})$, where $U$ is the overall superficial gas velocity and $U_{df}$ is the superficial velocity through the small bubbles[75]:

$$\epsilon_b = 0.3 \times D_T^{-0.18} \times (U - U_{df})^{0.58} \quad (6.32)$$
Chapter 6

Section 6.5

The small bubbles holdup is a function of gas density, \( \rho_g \), and the catalyst holdup\[73\]:

\[
\epsilon_{df} = 0.27 \times \left( \frac{\rho_g}{1.2 \, \text{kg/m}^3} \right)^{0.48} \times (1 - 2.592 \times \epsilon_s) \quad (6.33)
\]

The increase in catalyst concentration also leads to an increase in small bubble rise velocity, \( V_{small} \), and an increase in the superficial velocity through the small bubbles\[73\]:

\[
V_{small} = \left( \frac{0.095 \, \text{m}}{s} \right) \times (1 + 8.421 \times \epsilon_s) \quad (6.34)
\]

\[
U_{df} = \epsilon_{df} \times V_{small} \quad (6.35)
\]

In the dynamic simulation, the column diameter is assumed to be 7.5\( m \), the overall superficial gas velocity is the manipulated input variable, the catalyst holdup is assumed to be 0.36, and the gas density is calculated using a Peng-Robinson thermodynamic package, with equations (6.30)-(6.35) solved simultaneously for \( \epsilon, \epsilon_{df}, \epsilon_L, \epsilon_b, V_{small}, U_{df} \).

For each species \( c \) \( \in \{ \text{H}_2, \text{CO}, \text{H}_2\text{O}, \text{C}_n\text{H}_{2n}, \text{C}_n\text{H}_{2n+2} \} \), the volumetric mass transfer coefficient between the large bubble and the slurry liquid, \( (k_{L, a})_{c,\text{large}} \), is determined by:

\[
\frac{(k_{L, a})_{c,\text{large}}}{\epsilon_b} = (0.5 \, \text{s}^{-1}) \times \sqrt{\frac{D_{c,L}}{2 \times 10^{-9} \, \text{m}^2 \text{s}^{-1}}} \quad (6.36)
\]

The volumetric mass transfer coefficients of species \( c \) between the small bubble and the slurry liquid, \( (k_{L, a})_{c,\text{small}} \), is determined by the following expression.

\[
\frac{(k_{L, a})_{c,\text{small}}}{\epsilon_b} = (1.0 \, \text{s}^{-1}) \times \sqrt{\frac{D_{c,L}}{2 \times 10^{-9} \, \text{m}^2 \text{s}^{-1}}} \quad (6.37)
\]

The diffusivity of the species \( c \) in the liquid phase, \( D_{c,L} \), are calculated using Peng-Robinson thermodynamic method assuming the paraffin oil is C\(_{16}\)H\(_{34}\). Let \( A \) be the cross-sectional area of the bubble column, the dispersion height \( H = 30 \, m \), the slurry’s superficial velocity \( U_L = 0.01 \, m/s \), \( \xi_c \) denotes the extent of reaction of species \( c \), \( C_{G,\text{large}} \) indicates the concentration of species \( c \) in the large bubbles, \( C_{G,\text{small}} \) represents the concentration of species \( c \) in the small bubbles, \( C_{L,c} \) stands for the concentration of species \( c \) in the liquid slurry, \( m_c \) is the Henry constant of species \( c \). The mass balances at the differential
reactor height, \( dz \), are represented according to the following equations:

\[
\frac{d}{dt} (\epsilon_b \times C_{G,c \text{ large}}) = - \frac{d}{dz} \left( (U - U_{df}) C_{G,c \text{ large}} \right) - (k_{La})_{c, \text{ large}} \left( \frac{C_{G,c \text{ large}}}{m_c} - C_{L,c} \right) \tag{6.38}
\]

\[
\frac{d}{dt} (H \times \epsilon_{df} \times C_{G,c \text{ small}}) = U_{df} \times \left( C_{G,c \text{ small}}^{r} - C_{G,c \text{ small}} \right) - (k_{La})_{c, \text{ small}} \left( \frac{C_{G,c \text{ small}}}{m_c} - C_{L,c} \right) H \tag{6.39}
\]

\[
\frac{d}{dt} (H \times \epsilon_L \times C_{L,c}) = \int_0^H (k_{La})_{c, \text{ large}} \left( \frac{C_{G,c \text{ large}}}{m_c} - C_{L,c} \right) dz + H (k_{La})_{c, \text{ small}} \left( \frac{C_{G,c \text{ small}}}{m_c} - C_{L,c} \right) \tag{6.40}
\]

\[ - U_L C_{L,c} + H \times \epsilon_L \times R_{CO+H2} \times \xi_c \]

The separation of the three phases is a modeling simplification concept, as the phases are entangled with each other in the actual system. Assuming the heat capacities of the gas phases are negligible when compared to the slurry phase, the energy balance for the reactor is given by:

\[
HA \epsilon_L \rho_s C_{P,s} \frac{dT}{dt} = \dot{m}_{H_2O} C_{P,H_2O} \left[ 1 - \exp \left( - \frac{\alpha_{eff} A_{hx}}{\dot{m}_{H_2O} C_{P,H_2O}} \right) \right] + HA \epsilon_L R_{CO+H2} (\Delta H_{rx}) - U_L A C_{P,S} (T - T_0) \tag{6.41}
\]

The heat exchanging area, \( A_{hx} \), is assumed to be \( 6.0 \times 10^4 \text{m}^2 \), the slurry inlet temperature, \( T_0 \), is assumed to be \( 200\degree C \). The heat of reaction, \( \Delta H_{rx} \), is calculated by the differences between the specific enthalpy of the reactants and the products. The overall heat transfer coefficient, \( \alpha_{eff} \), the slurry density, \( \rho_s \), the slurry heat capacity, \( C_{P,s} \), are calculated according to the literature [76], and respectively have the values of \( 22.23 \text{Wm}^{-2} \text{K}^{-1} \), \( 810 \text{ kg m}^{-3} \) and \( 796 \text{ J kg}^{-1} \text{K}^{-1} \). The coolant flowrate \( \dot{m}_{H_2O} \) is a manipulated variable.

The product distribution is assumed to follow the Anderson-Schulz-Flory distribution with the chain growth probability \( \alpha_{rx} \). The chain growth probability is used as a controlled variable to manipulate the product selectivity. The formula of \( \alpha_{rx} \) is assumed to be affected by the termination to olefin rate, \( T_{\text{olefin}} \), and the termination to paraffin rate,
\( T_{\text{paraffin}} \), which are described in the following equations\[77\]:

\[
T_{\text{ol}lefin} = 6.1686 \times P_{H_2}^{-0.5} \tag{6.42}
\]

\[
T_{\text{paraffin}} = 13.8 \times P_{H_2}^{-0.47} \times P_{CO}^{-0.43} \tag{6.43}
\]

\[
\alpha_{rx} = \frac{T_{\text{paraffin}}}{1 + T_{\text{paraffin}} + T_{\text{ol}lefin}} \tag{6.44}
\]

### 6.5.2 DD-ROM calibration and closed-loop simulation results

The considered reduced-order model of the Fischer-Tropsch synthesis bubble column reactor is selected to be the continuous stirred tank reactor that corresponds to the slurry phase of the full model. Since the reactions occur in the slurry phase and on the surface of the catalyst submerged, an accurate submodel of the slurry phase is sufficient to capture the reaction kinetics. Furthermore, it is assumed that the cooling fins only establish contact with the slurry during the computation of the heat exchange area, so the slurry phase adequately captures the temperature regulation of the model. The mass balances in (6.38)-(6.40) are simplified to the following expression (6.45). The dynamic discrepancy function that depends on the states of the slurry phases and the manipulated inputs is introduced in (6.45):

\[
\frac{d}{dt} (H \times \epsilon_L \times C_{L,c}) = \delta(C_{L,c}, U_L, \dot{m}_{H_2O}) \\
- U_L C_{L,c} + H \times \epsilon_L \times R_{CO+H_2} \times \xi_c \tag{6.45}
\]

In the HFM of the Fischer-Tropsch synthesis process, the plug-flow characteristic of the large bubble phase is modeled by a discretization along the height of the slurry bubble column using the method of lines. Therefore, each species in the large bubble phase is represented by a vector of state variables that correspond to a specific location in the reactor, and the number of differential equations increases proportionally with the number of considered species. Because the DD-ROM only accounts for the slurry phase, which is assumed to be well-mixed due to the rise of the large bubbles, each species in this phase only requires one differential equation to represent these dynamics. Additionally, instead
of considering a full set of species \( c \in \{ \text{H}_2, \text{CO}, \text{H}_2\text{O}, \text{C}_n\text{H}_{2n}, \text{C}_n\text{H}_{2n+2} \} \) with the length of the hydrocarbons between 1 and 30, the DD-ROM admits a pseudo-component to replace the paraffin and olefin presented in the model. The product specification is recreated using the Anderson-Schulz-Flory distribution and the concentration of the pseudo-component.

The calibration data generation is performed as described in subsection 6.3.2. Pseudo-random sequences are employed to generate the excitation signals corresponding to the manipulated inputs, namely the coolant flow rate and the syngas feeding rate. The amplitudes of the sequences match the upper and lower limits of the manipulated inputs, and the frequency is chosen to be 5 min for the reactor to reach new steady states. Two data sets are generated to be used as calibration data and validation data sets. Each data set contains 10,000 data points with a fixed 10 sec interval between two consecutive data points. While the length of the interval does not affect the calibration results because the dynamic discrepancy reflects the instantaneous mismatch in the rates of change, it is important to select a sufficiently short interval to capture the transient behaviors of the Fischer-Tropsch synthesis reactor.

The set of basis functions for the dynamic discrepancy in (6.45) are the basis of the BSS-ANOVA Gaussian Process obtained in subsection 6.3.2. For each calibration input, a set of BSS-ANOVA basis functions up to order 4 are generated, and cross interactions between basis functions of different calibration inputs are also considered. In total, there are 210 basis functions, and the number of possible models is \( 2^{211} \approx 3.3 \times 10^{63} \), which is computationally expensive to estimate. Thus, Algorithm 1 is applied with the Markov chain length of 100,000 and a burn-in length of 5,000. The most probable model is the model that only contains the basis functions with probabilities of inclusion higher than 50%. The validation data set confirms the fitness of the obtained DD-ROM, and a fraction of the validation result is shown in Figure 6.7. An important property observed from the validation data set is that the calibration errors do not accumulate over time because the formulation of the dynamic discrepancy function satisfies the proposed criteria in 6.3.1.

A closed-loop system of the Fischer-Tropsch reactor is simulated using the HFM as the plant, and the DD-ROM as the equality constraints of the NMPC. The controller
Figure 6.7: Validation of the DD-ROM for the Fischer-Tropsch reactor

objectives are setpoint tracking and disturbance rejection. The predictive horizon of the NMPC is selected to be 20 min, with the update frequency of the manipulated variables every 30 sec. The optimal solutions of the online nonlinear programming problem of the NMPC are successfully solved within the 30-sec limit, proving a potential application of the proposed NMPC to an actual process. Various setpoint combinations are given to test the robustness of the NMPC, and the controller is able to bring the plant model to the desired outputs successfully for every case. Although the output discrepancy is not explicitly computed, the dynamic discrepancy formulation has the capability of indirectly determining the appropriate manipulated inputs associated with the setpoints by means of matching the state variables. Additionally, the data generation collects the dynamic performances every 10 sec while the NMPC is solved with a 30-sec hold, and the convergence of the DD-ROM is not affected. The proposed framework with DD-ROM formulation remains unaffected by the sampling time, thereby offering increased flexibility in both data collection and NMPC design.
Figure 6.8: Closed-loop performance of an NMPC with dynamic discrepancy reduced-order model
Chapter 7

Conclusions

In this dissertation, the dynamic operability concepts were extended, starting from the classical operability concepts. The distinctions between steady-state operability and dynamic operability led to different implementations to find feasible and operable design regions. While steady-state and dynamic operability were inherent properties of the system and thus independent of the formulated control laws, adaptations for the operability measure and mapping were proposed to evaluate the closed-loop performance of different designs. The developed framework was applied to a modular DMA-MR for a gas processing application in the Marcellus Shale. In addition to the theoretical operability contributions provided, this work proposed a novel dynamic model of the DMA-MR and a control structure that guaranteed setpoint reachability regardless of the realizations of the disturbance. While dynamic operability was shown to be an effective tool for design and control assessment, the obtained dynamic mappings relied on a number of simulations that increased exponentially with the number of time steps. The main reason for the computational challenges when scaling the dynamic operability mapping procedure was the fact that each manipulated input could take a different value at a different moment in time. Thus, the longer the time horizon was in the dynamic operability mapping, the larger the number of input combinations was needed to generate all available input sequences.

A state-space projection approach was proposed for linear and nonlinear dynamic mapping to address the computational challenge of dynamic operability mapping. For linear
dynamic systems, a novel dynamic operability mapping was proposed in a two-step framework that allows the majority of the computational effort to be performed offline. The achievable output sets at different predictive times were formulated as a set of inequality constraints that were updated online according to the current full-state information and uncertainty propagation. For the nonlinear operability mapping, feedback mapping was proposed in combination with state-space projection.

A multiple steady-state identification framework was proposed with a proof of concept case study. The identification method was based on the observations of the steady-state stability during the construction of the dynamic operability mapping. A novel dynamic divergent jump method was introduced to stably move between different steady states at a fixed input combination.

Finally, a novel framework for dynamic discrepancy reduced-order formulation was introduced for advanced control applications. The framework consisted of three steps that can be independently improved in future work. The first step was formulating the dynamic discrepancy term in the rate of change equations, and three important criteria were proposed as heuristics for implementations. The second step was generating the calibration data, which was proposed as a moving horizon estimation. The last step was a Bayesian interference to simultaneously estimate the hyperparameters and select a model structure according to the Occam’s Razor principle. Although the model reduction technique was not the primary focus of this work, the dynamic discrepancy function introduced significantly improves the reduced-order model obtained by compensating for the differences in the rates of change instead of the time-varying outputs of the ROM. During the formulation, the study presented a series of heuristics as criteria for constructing a DD-ROM that was suitable for advanced control applications. Moreover, a moving horizon data collection procedure was proposed that enhances the flexibility of placing the discrepancy functions in the ROM, while simplifying the calibration process to a linear Bayesian model selection. The approach considered the construction of the posterior of the model given the calibration data and employed a Gibbs sampling procedure to search for the most probable model when the number of possible models was extensive.
due to a high number of potential basis functions. The framework was successfully applied to a closed-loop control of the Fischer-Tropsch slurry bubble column reactor, and the results obtained through the application of DD-ROM showed that it fits for practical implementation.
Chapter 8

Future Recommendations

8.1 Computational Methods for Operability

While the state-space projection and feedback mapping are shown to prevent the computational time of dynamic operability mapping to increase exponentially, the components of these methods can potentially be improved while preserving the individual functionality. The simplification procedure in state-space projection is performed in an ad-hoc manner at the moment, and thus an improved procedure based on topological optimization can potentially increase computational efficiency. The linearization test in the feedback mapping is currently the bottleneck of the computational efforts, and it is recommended to be reformulated from a hypervolume-based to a path integration-based technique.

With the recent discovery of multiple steady-state identifications, the code base of the current Python operability app needs significant restructuring to be compatible with input-output multiplicity pairings. Additionally, an adaptation of the divergent jump method can potentially identify input multiplicities in the inversed mapping of operability analyses. In this adaptation, since the objective is mapping from the output space to the input space, the state variables are the domain of the mapping, and the process inputs are the images of the mapping. Thus, an augmented inverted dynamic system is recommended for divergent jump alteration. The augmented dynamic system should mimic the stability of the original process model. Furthermore, the future development of the divergent jump method for multiple steady-state identifications should replace the
current fixed-length divergent radius increment with an adaptive radius increased similarly to a line search method in nonlinear optimization. This approach will minimize the chance of overstepping a steady state, and a faster convergent rate to a new steady state may be observed. Additionally, the time derivatives should be included in the indicator function of the divergent source. This inclusion can assist in the event that the initially chosen jump bypasses a nearby steady-state solution.

In this work, a novel fast operability set intersection for Gaussian disturbance is proposed in Theorem 1 of Chapter 4.2. This approach is also applicable to steady-state processes with Gaussian disturbances. More importantly, since any bounded convex set in an n-dimension Euclidean space can be bijectively transformed to an n-sphere, the approach can potentially be generalized for any disturbances with an additive effect on the achievable output sets. If the disturbances also transform the achievable output set, a convex envelope can be proposed to approximate the disturbance effects, and the proposed Theorem can be generalized to be applied.

8.2 Dynamic Discrepancy Expansion

The proposed dynamic discrepancy reduced-order modeling framework is constructed from three components that can be individually updated: the formulation of the discrepancy function in the reduced-order models, the construction of the black-box function for the dynamic discrepancy, and the calibration of the black-box function. One of the key aspects of dynamic discrepancy that is yet to be addressed is balancing between steady-state and transient behaviors. In future work, two sets of discrepancy functions are recommended to flexibly match the plant dynamic model at both operating modes. To switch between steady-state and dynamic operations, an indication function is recommended to switch when the rates of change approach zero. Additionally, the indication function in (5.6) has the flexibility to cover any arbitrary convex or nonconvex region, so it is suitable for constructing a multi-model discrepancy function that behaves differently at different operations.

Online calibration is a feasible alternative direction for the development of dynamic
discrepancy. Using the Bayesian inference, a potential online update for the model selection and parameter calibration can be derived in a similar manner to the Kalman Filter approach for state estimation.

8.3 Control Applications of Dynamic Operability

Control applications of dynamic operability funnels can be considered a major future development. Chance constraints can potentially be constructed from the disturbance intersections of achievable output sets, and transient constraints can be directly applied using the hyperplane representation of the operability funnel. In the chance constraint formulation, the shrinkage of the achievable output set is the relaxation magnitude for the joint chance constraints. Transient constraints are best used in conjunction with risk assessments and safety protocols, and countermeasures are recommended to be investigated in the event that the control structure is unable to compensate for the disturbance effects. Additionally, the proposed dynamic operability framework can be potentially adapted to address the observability of a dynamic system. The set of observable inputs can be formulated as the desired input set, and the intersection of the desired input sets at different disturbances would reveal if a unique solution of the full-state information could be derived from the given measurements.
References


Chapter 8

Section 8.3


