DISS. ETH NO. 30457

DATA-DRIVEN AND HYBRID MODELING AND OPTIMIZATION OF CHEMICAL AND BIOLOGICAL PROCESSES

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES
(Dr. sc. ETH Zurich)

presented by

TIM FORSTER

MSc ETH Chem. Bio. Eng., ETH Zurich

born on *10.01.1994*

accepted on the recommendation of
Prof. Dr. Gonzalo Guillén Gosálbez, examiner
Dr. Antonio Del Rio Chanona, co-examiner
Prof. Dr. Paolo Arosio, second advisor

2024

To my girlfriend, friends and family
who wonder what I do the whole day

Acknowledgements

The present work would not have been possible without the support of many people from my private and academic environment. I would like to thank all of them, as their support, encouragement, and inspiration have been invaluable throughout this journey.

First and foremost, I would like to express my deepest gratitude to the head of our research group, Prof. Dr. Gonzalo Guillén Gosálbez. Your guidance, support, and encouragement made it possible for me to pursue this work within the group. Your belief in my colleagues' and my abilities and your willingness to provide me with opportunities have been indispensable on this academic path. I am truly grateful for the trust you have placed in our colleagues and me, and for the mentorship you have provided.
I am deeply indebted to Dr. Daniel Vázquez Vázquez, whose mentorship and expertise have shaped my understanding of most of the topics I touched during my work. Your patience, dedication, enthusiasm, and willingness to share your knowledge have been instrumental in my professional development, and I am truly grateful for the priceless lessons I have learned from you.
I would like to express my deepest appreciation to Dr. Antonio Del Rio Chanona, with whom I had the privilege of engaging in a research exchange at the Sargent Centre for Process Systems Engineering at the Imperial College in London. Your insights, perspectives, and diverse approach to research have broadened my horizons and enriched my understanding of the field. I am overwhelmingly thankful for the time we spent together, and I will always cherish the lessons I learned from our interactions.
I would also like to thank Dr. Michael Sokolov, Dr. Fabian Feidl, and Dr. Alessandro Butté, who, back in 2017, brought the fields of programming, data analysis, modeling, and machine learning into my life. The interactions with you and your colleagues helped me to discover the passion for these fields. Without bringing me onto this track, I would probably be not even be aware of it, and therefore I would not have been able to pursue this academic path.

To my colleagues of and around the Sustainable Process Systems Engineering group at ETH Zurich, my collaborators, the people I was lucky to meet and interact with at the Imperial College in London, and all my students I had the privilege to supervise. I am deeply thankful for your support, collaboration, and the stimulating discussions we have shared. Each of you made my time at ETH in Zurich and also in London an unforgettable experience. All of your diverse perspectives, innovative ideas, and collective expertise have challenged and inspired me to push the boundaries of my research and strive for excellence. A special thanks also goes to our group's administrative staff for their support, dedication, and willingness to help. Thank you very much for your kindness, patience, and professionalism, which made my time at ETH Zurich a truly enjoyable experience.

I also want to express my deepest gratitude to my girlfriend, Laura. Your support and understanding have been fundamental throughout this journey. Our relationship has provided me with priceless insights into how I look at my work and my entire life, and I am endlessly grateful for the profound impact you have had on shaping me into the person I am today. You are full of true inspiration, courage, and passion, and without your encouragement, love, and support, this academic journey would literally not have happened.

I also owe an immense debt of gratitude to my parents, Christina and Walter, and my sister, Nina, for their unwavering support throughout my academic and professional endeavors. Your belief in me, coupled to the opportunities you provided, has been the foundation of my success. I am endlessly grateful for your love, guidance, and sacrifices, which have brought me to where I am today.

Last but certainly not least, I would also like to extend my heartfelt thanks to my friends, whose immense support and companionship have been a source of joy and inspiration. Your presence in my life has provided me with much-needed moments of respite and laughter, allowing me to deeply enjoy the time of my academic and private life, recharge my batteries, and tackle challenges with renewed vigor.

I am deeply grateful to all those who have supported and inspired me along the way. Your contributions, whether big or small, have played an integral role in shaping my private and professional journey, and for that, I am truly thankful!

Tim Forster
Zurich, 2024

Abstract

The work presented in this thesis focuses on addressing modeling, optimization, and flexibility analysis challenges which are encountered on different conceptual scales within the chemical and biological industry. By leveraging advancements in machine learning, mathematical modeling, and optimization techniques, innovative data-driven and hybrid approaches are introduced as alternatives to existing methods. These approaches provide solutions to address challenges spanning from micro scale (i.e., kinetic modeling) to macro scale (i.e., flowsheet optimization). Through these implementations, which are created using concepts from the Process Systems Engineering field, decision-making processes are supported in a digital manner, which can improve the understanding of a system under study, and enhance its efficiency and sustainability.

Mathematical modeling helps to guide experiments more effectively, to support process monitoring and control tasks, to stabilize product quality, to increase consumer safety, or to ease specific decision-making tasks for subject matter experts. Considering kinetic systems – a conceptually microscopically small scale – the construction of accurate models can be challenging, especially with bioprocesses, due to complex metabolic mechanisms and data scarcity. Chapter 2 tackles these challenges and proposes a method for building models combining a mass balance backbone in form of a canonical kinetic representation – thereby intrinsically incorporating expert knowledge – with a scarce dataset. The final model structure and parameters that best describe the studied system are automatically and simultaneously identified by using a mathematical programming approach. Following an incremental procedure, the integration of ordinary differential equations is avoided. Numerical examples show that the proposed method performs similarly to models based on artificial neural networks, even outperforming them in some cases while providing an analytical, closed-form model. Such expressions can be more easily and efficiently optimized in existing algebraic modeling systems. The resulting hybrid models can be physically interpreted since the intrinsically incorporated

knowledge guides the structure of the model (due to the canonical formalism). In Chapter 3, the entire workflow is then extended to the use of symbolic regression approaches. This chapter particularly targets the challenge when interpretable models for bioprocesses are sought in cases where little knowledge about the system is available. Classical machine learning algorithms are gaining wide interest to simulate complex bioprocesses that are hard to describe via first principles. They often rely on *a priori* assumptions of the model structure and lead to mathematical expressions that are hard to interpret or to further integrate into other platforms. Therefore, Chapter 3 proposes an alternative approach based on symbolic regression to identify bioprocess models without assuming a pre-defined model structure. Algebraic expressions for the kinetic rates are obtained from measured data that only consists of concentration profiles. The workflow builds up on the one presented in Chapter 2, which allows avoiding the iterative integration of differential equations for the parameter estimation step. The proposed procedure was found to slightly outperform neural network benchmarks. Moreso, the obtained algebraic expressions for the rate equations facilitate the model interpretation and enable the direct application of optimization algorithms. Going from such a conceptually small scale to a macro scale, the presented thesis addresses the challenges of globally optimizing process units and flowsheets through the use of surrogate models and state-of-the-art optimization algorithms. Therefore, in Chapter 4, a strategy for the global optimization of processes is proposed. In a first step, algebraic surrogates are built from rigorous simulations via symbolic regression. The applied method provides a closed-form expression that, in a second step, can be optimized to global optimality using state-of-the-art solvers. When predicting unseen test data, the algebraic models show a similar accuracy level compared to traditional surrogates based on Gaussian processes. However, they can be more easily optimized to global optimality due to their closed-form structure, which allows the user to apply well-established global deterministic solvers. The capabilities of the proposed approach are demonstrated in several case studies, ranging from the meso scale (process units) to the macro scale (full flowsheets). Finally, by leveraging surrogate modeling techniques, Chapter 5 provides a framework that allows to analyze the flexibility of a production unit in cases when process constraints are difficult to model. Flexibility analyses are widespread in chemical engineering to quantify allowed deviations from nominal conditions. Standard approaches to perform flexibility analysis can be hard to apply if process constraints are difficult to handle. This chapter focuses on the computation of the traditional flexibility index in problems with complicating constraints. In a first step, symbolic regression is used to build algebraic expressions of the said complicating constraints. This allows, in a second step, to simplify the flexibility analysis of complex process models by enabling the application of state-of-the-art

deterministic solvers. This approach is applied to two different case studies. The performance is assessed in terms of model building time, predictive accuracy of the model, and the time required to solve the flexibility formulations. Overall, this combination of a deterministic formulation and data-driven models – a hybrid framework – provides a way to analyze the process flexibility entailing complicating constraints. Finally, Chapter 6 summarizes the main findings, where limitations of the proposed methods are discussed, and possible future research avenues are described.

The provided and discussed tools offer alternative solutions that allow to model, analyze, and optimize systems under study, which subsequently support the decision-making process for practitioners. Each of the chapters discusses how the applied surrogate and hybrid frameworks can be useful to create applications that allow advancing digitalization within the chemical and biological production industry.

Zusammenfassung

Die vorliegende Dissertation fokussiert auf Herausforderungen in der Modellierung, Optimierung und der Flexibilitätsanalyse in der chemischen und biologischen Industrie. Solche Herausforderungen können auf verschiedenen konzeptionellen bzw. physikalischen Dimensionen auftreten. Aktuelle Fortschritte in der mathematischen Modellierung und Optimierung werden genutzt, um datengetriebene und hybride Lösungsansätze zu entwickeln, die Alternativen zu bereits existierenden Möglichkeiten darstellen. Die Implementierung solcher neuen bzw. alternativen Ansätzen wird auf Konzepten der Prozesssystemtechnik aufgebaut. Die vorgeschlagenen Methoden sollen bei der Bewältigung von Fragestellungen helfen, welche auf konzeptionell und physikalisch kleinen (z.B. kinetische Modellierung) und grossen (z.B. Optimierung von gesamten Produktionsprozessen) Dimensionen auftreten. Durch die Nutzung solcher Methoden werden Entscheidungsprozesse auf digitale Weise unterstützt, das Verständnis eines untersuchten Systems verbessert und seine Effizienz und Nachhaltigkeit gesteigert.

Ein System mathematisch zu modellieren kann helfen Experimente effektiver zu planen, Prozessüberwachungen zu unterstützen, die Produktqualität zu stabilisieren, die Verbrauchersicherheit zu erhöhen und fachspezifische Entscheidungen zu erleichtern. Bei kinetischen Systemen – ein konzeptionell und physikalisch kleiner Massstab – kann die Konstruktion von akkuraten Modellen eine Herausforderung darstellen. Dies trifft insbesondere auf Bioprozesse zu, bei denen zum einen komplexe Stoffwechselmechanismen auftreten, und zum anderen die Datensätze meist begrenzt sind. Diese Problematik wird in Kapitel 2 der vorliegenden Arbeit aufgegriffen. Dabei wird eine eine Methode zur Modellgenerierung besprochen, welche eine Massenbilanz in Form einer kanonisch-kinetischen Darstellung mit einem kleinen verfügbaren Datensatz kombiniert. Expertenwissen wird somit durch den benutzten kanonischen Formalismus und die damit verbundene mathematische Grundstruktur in die Generierung des Modells miteinbezogen. Die finale Modellstruktur und die involvierten Parameter werden mit Hilfe eines mathe-

matischen Optimierungsansatzes automatisch und simultan ermittelt. Durch die entwickelte mehrstufige Methode wird die iterative Integration von Differentialgleichungen vermieden. Numerische Beispiele zeigen, dass die dadurch generierten Modelle ähnlich gute oder sogar bessere Ergebnisse erzielen können wie Modelle, die auf künstlichen neuronalen Netzwerken basieren. Im Gegensatz zu trainierten neuronalen Netzwerken liefert die vorgestellte Methode jedoch analytische und geschlossene Gleichungen. Die durch den vorgestellten Ablauf resultierenden hybriden Modelle können physikalisch interpretiert werden, da sie auf der Grundstruktur der erwähnten kinetischen Darstellung beruhen. Ein weiterer Vorteil solcher Modelle liegt darin, dass sie aufgrund der geschlossenen Formulierung in bestehenden algebraischen Modellierungssystemen leicht und effizient optimiert werden können. Der in Kapitel 2 beschriebene Ablauf wird dann in Kapitel 3 durch die Verwendung symbolischer Regressionsansätze erweitert. Dieses Kapitel befasst sich zudem damit, interpretierbare Modelle für Bioprozesse zu generieren, bei welchen wenig oder gar kein Wissen über das vorliegende System vorhanden ist. Klassische Algorithmen des maschinellen Lernens gewinnen zunehmend an Interesse, um komplexe Bioprozesse zu simulieren, da sich diese Prozesse teils nur schwer durch erste Prinzipien beschreiben lassen. Obwohl solche rein datenbasierte Methoden sehr hilfreich sein können, beruhen sie oft auf *a priori* Annahmen über die Modellstruktur und/oder führen zu mathematischen Ausdrücken, die schwer zu interpretieren oder in weitere Plattformen zu integrieren sind. Daher wird in Kapitel 3 ein auf der symbolischen Regression basierender alternativer Ansatz vorgeschlagen, um Modelle zu identifizieren, ohne dass dabei eine vordefinierte Modellstruktur angenommen werden muss. Daraus resultieren algebraische Ausdrücke für die Prozesskinetik, wobei lediglich gemessene Konzentrationsprofile benötigt werden um die Modelle zu trainieren. Das Vorgehen baut auf der in Kapitel 2 vorgestellten Methode auf und ermöglicht es wiederum, die iterative Integration von Differentialgleichungen zu vermeiden. Das vorgeschlagene Verfahren kann Modelle generieren, welche die Benchmarks (neuronale Netzwerke) im Bezug auf die Vorhersagegenauigkeit übertreffen. Darüber hinaus erleichtern die erhaltenen algebraischen Ausdrücke die Modellinterpretation und ermöglichen die direkte Anwendung von Optimierungsalgorithmen oder die Einbindung in weitere Plattformen. Kapitel 4 der vorliegenden Arbeit befasst sich mit den Herausforderungen der globalen Optimierung von Prozesseinheiten und Fliessbildern durch die Kombination von approximativen Modellen und bekannten Optimierungsalgorithmen. Zudem wird eine konzeptionell und physikalisch grössere Dimension betrachtet. Mittels symbolischer Regression werden dabei in einem ersten Schritt Surrogatmodelle aus Simulationen erstellt. Diese Regressionsmethode liefert dabei Ausdrücke in algebraischer Form. Bei der Vorhersage ungesehener Testdaten weisen diese algebraischen Modelle eine ähnliche Genauigkeit auf wie

traditionelle Surrogate, die auf Gaussschen Prozessen basieren. Ein Vorteil der identifizierten Surrogtmodelle in algebraischer Form ist jedoch, dass die Nutzung von bestehenden und etablierten globalen deterministischen Optimierungsalgorithmen ermöglicht wird. Die Anwendbarkeit der vorgeschlagenen Strategie wird in mehreren Fallstudien demonstriert, welche von der Mesoskala (Prozesseinheiten) bis zur Makroskala (Fliessschemata von ganzen Prozessen) reichen. Schliesslich wird in Kapitel 5 darauf eingegangen, wie Surrogatmodelle helfen können die Flexibilität einer Produktionseinheit in Fällen zu analysieren, in denen Prozessbeschränkungen schwer zu modellieren sind. Flexibilitätsanalysen sind in der chemischen Verfahrenstechnik weit verbreitet, um die zulässigen Abweichungen von den Nennbedingungen zu quantifizieren. Standardansätze zur Durchführung von solchen Analysen sind in der Regel schwer anwendbar, wenn die Prozessbeschränkungen mathematisch schwierig zu beschreiben oder zu handhaben sind. Kapitel 5 konzentriert sich auf die Berechnung des traditionellen Flexibilitätsindexes bei Problemen mit komplizierten Nebenbedingungen. Es werden in einem ersten Schritt algebraische Ausdrücke für die besagten Nebenbedingungen identifiziert, wodurch in einem zweiten Schritt die Flexibilitätsanalyse vereinfacht und die Anwendung deterministischer Löser ermöglicht wird. Diese Methode wird auf zwei verschiedene Fallstudien angewandt. Dabei werden die Zeit für die Modellerstellung, die Vorhersagegenauigkeit des Modells und die für die Lösung der Flexibilitätsformulierungen erforderliche Zeit genauer diskutiert. Insgesamt bietet die Kombination von datengetriebenen Modellen mit einer deterministischen Problemformulierung – demnach gesamthaft ein hybrider Ansatz – eine Möglichkeit zur Analyse der Flexibilität von Prozessen, die schwer zu beschreibende Einschränkungen beinhalten. Abschliessend werden in Kapitel 6 die Resultate dieser Arbeit zusammengefasst, wobei auch auf mögliche Themen für zukünftige Forschungsfragen eingegangen wird.

Die entwickelten und beschriebenen Applikationen bieten allesamt alternative Lösungen, welche es ermöglichen, die untersuchten Systeme zu modellieren, zu analysieren und zu optimieren. Dadurch können Entscheidungsprozesse für verschiedenste Fragestellungen in der chemischen und biologischen Produktion auf digitale Weise unterstützt werden.

List of Figures

List of Tables

Abbreviations

ALAMO	Automated learning of algebraic models for optimization
ALVEN	Algebraic learning via elastic net
ANTIGONE	Algorithms for continuous / integer global optimization of nonlinear equations
ANN	Artificial neural network
ARGONAUT	Algorithms for global optimization of constrained grey-box computational problems
BARON	Branch-and-reduce optimization navigator
BIC	Bayesian information criterion
BIDSAM	Bayesian identification of Dynamic Sparse Algebraic Models
BMS	Bayesian machine scientist
BST	Biochemical systems theory
CAPEX	Capital expenditure
CC	Construction cost
CCS	Carbon capture and storage
CE	Contingency expense
CFD	Computational fluid dynamics
COBALT	Constrained Bayesian optimization of computationally expensive grey-box models exploiting derivative information
COM	Component object model
CPU	Central processing unit
CQA	Critical quality attribute
CS	Case study
CW	Capital for working
DAC	Direct air capture
EKF	Extended Kalman filter
FD	Finite difference
FDA	Food and Drug Administration
FOC	Fixed cost of operation
GAMS	General Algebraic Modeling System
GMA	Generalized mass action

GO	Global optimization
GP	Gaussian process
HDMR	High-dimensional model representation
ICH	International Council for Harmonisation of Technical Requirements for Pharmaceuticals for Human Use
ISBL	Inside battery limits
KKT	Karush-Kuhn-Tucker
LASSO	Least absolute shrinkage and selection operator
LBF	Linear basis function
LCEN	LASSO-Clip Elastic net
LHS	Latin hypercube sampling
LSTM	Long short-term memory
MAE	Mean absolute error
MAiNGO	McCormick-based algorithm for mixed-integer nonlinear global optimization
MCMC	Markov-chain Monte Carlo
ME	Motivational example
MeLON	Machine Learning models for Optimization
MI(N)LP	Mixed-integer (non)linear programming
ML	Machine learning
(N)LP	(Non)linear programming
ODE	Ordinary differential equation
OMLT	Optimization and machine learning toolkit
OPEX	Operational expenditure
OSBL	Outside battery limits
OVP	Observed versus predicted
PDE	Partial differential equation
PINN	Physics-informed neural network
PSE	Process systems engineering
RAE	Relative absolute error
RMSE	Root mean squared error
RO	Robust optimization
SBB	Simple branch-and-bound
SG	Savitzky-Golay
SINDy	Sparse identification of nonlinear dynamics
SMR	Steam methane reforming
SP	Stochastic programming
SR	Symbolic regression
SSR	Sum of squared residuals
SST	Sum of squared total
VOC	Variable cost of operation

Table of Contents

Chapter 1

Introduction

1.1 Digitalization in the chemical and biological sector

The industrial revolution is not merely a recent phenomenon but has roots that date back to the 18th century (Petrillo et al., 2018). During these times, humanity was dealing with the daily challenge of creating better goods while considering limited resources, growing demand, and the impact on society and its environment (Beier et al., 2018; J. M. Müller et al., 2018). These main challenges have not significantly changed since then, and they remain up until today. However, the tools that we are able to apply nowadays certainly did change. After going through a second and third industrial revolution (Figure 1.1), characterized by the applications of electricity and automation steps in the industry, respectively, the term "Industry 4.0" sets another important milestone in this centuries-lasting revolutionizing process (X. Chen et al., 2021; Communication-Promoters-Group-of-the-Industry-Science, 2013). Within the fourth industrial revolution, Industry 4.0, an initiative that emerged between 2011 and 2015, focuses on the application of advanced technologies in the industrial sector (Philbeck & Davis, 2018). It is, however, not only about integrating available technologies, but it also describes an entire concept of how advanced tools can be used to create interactions and provide insights into how resources can be used for a more efficient and sustainable path to the future (Koh et al., 2019; Lasi et al., 2014). In this entire concept, digitalization has become an important part through which profound transformations are taking place, not only in the industrial sector, but also in our society (J. M. Müller et al., 2018; Rosen et al., 2015). Especially in the past decades, the enhanced global interconnection brings new challenges and drives competition in the markets, where the pace and extent of digital integration and the application of advanced technologies have accelerated exponentially (Koh et al., 2019; Zhou, 2013). Focusing specifically on the production sector of chemicals and biologics, digitalization can help to enhance efficiency and productivity by measuring, analyzing, modeling, automating, and optimizing processes and integrating many tasks (Fantke et al., 2021; Pantelides & Pereira, 2024). The digital transformation in the chemical

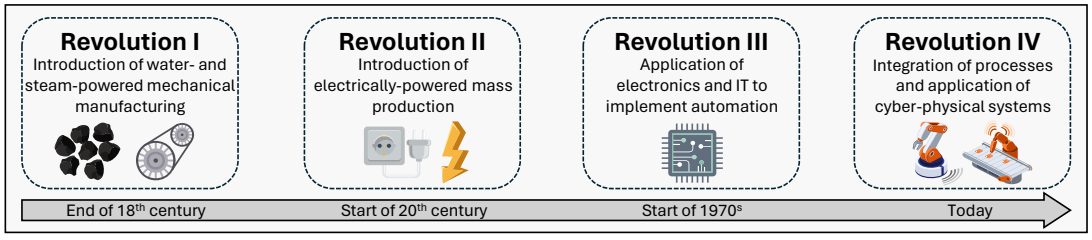

Figure 1.1. Simplified overview of the different steps of the Industrial Revolution. Adapted from Chen et al. (2021).

industry was estimated to deliver between $310 billion to $550 billion in value between 2016 and 2025 (ABB, 2018; Accenture & WEF, 2017). Furthermore, a recent report by the World Economic Forum (2022) describes the increasing importance of industrial supply chain resilience or environmental sustainability during global disruptions, such as the recent COVID-19 pandemic. It is mentioned that more than 70% of surveyed manufacturing executives considered advanced analytics methods to be more important than they have been some years earlier. The role of data, data-driven decision-making, self-optimizing systems, and, in general, digital transformations are highlighted to be key enablers for manufacturing environments to tackle highly complex future challenges. Such digital transformations might occur in wide range of applications (Figure 1.2), from a macro scale level (i.e., digital twins and optimization of entire production processes or supply chains) down to a micro scale level (i.e., modeling of kinetics). To narrow this down, this thesis will specifically focus on the modeling and optimization aspects that might support such digital transformations in the chemical industry. In such a multiscale and interdisciplinary environment, there is a clear need for tools that enable a structured and tactical decision-making process. Pistikopoulos et al. (2021) describe the scientific discipline of Process Systems Engineering (PSE) as a *"field that focuses on integrating scales and components to describe the physicochemical system via mathematical modeling, data analytics, design, optimization and control"*. It therefore *"provides the 'glue' within the field of engineering and offers a scientific basis and computational tools towards addressing future challenges such as in energy, environment, the 'industry of tomorrow' and sustainability."*. With this very short description of PSE being an enabler for improvements in industry, the reader is subsequently introduced to the role and importance of PSE in more detail.

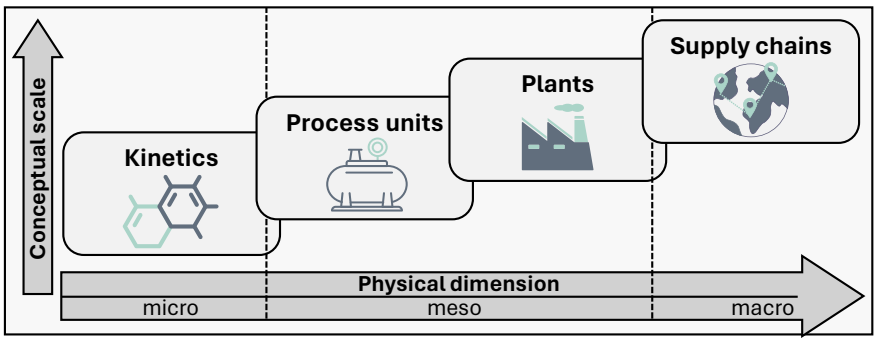

Figure 1.2. Schematically visualized conceptual scales that are targeted within this thesis. The concept was adapted from Ioannou et al. (2021).

1.2 The role of Process Systems Engineering

PSE – a term proposed in 1982 in Kyoto that combined aspects from various engineering disciplines (I. E. Grossmann & Westerberg, 2000; Pistikopoulos et al., 2021; Sargent, 2004) – plays a pivotal role in the digitalization of the chemical industry by integrating engineering principles with computational tools and data analytics (Sargent, 1967). PSE focuses on designing efficient systems, optimizing complex processes, and improving operational performance in various industrial aspects. In the context of digitalization within the chemical sector, PSE harnesses tools such as mathematical modeling, simulation, and optimization algorithms to address challenges across the chemical value chain (Klatt & Marquardt, 2009; Pistikopoulos et al., 2021), and contributes to the enhancement of efficiency, reliability, and sustainability in the chemical industry's digital transformation journey (I. E. Grossmann & Harjunkoski, 2019). Sargent (2004) and Stephanopoulos and Reklaitis (2011) provide a deeper insight into the history of PSE, whereas Pistikopoulos et al. (2021) describe the perspective of the PSE field in the future.

Sargent (1983) describes possible approaches for solving problems in the field of process engineering, specifically within the context of modeling, design and control: One may start formulating a considered problem mathematically and then use the mathematical structure to create algorithms, or one may try to use engineering knowledge and insights in order to solve the problem with physical intuition. In either of these ways described by Sargent (1983), one special and pivotal role in such a problem-solving approach is the use of modeling and optimization techniques (Figure 1.3). Pistikopoulos et al. (2021) summarize this description by Sargent by mentioning the example of chemical and biochemical engineers who work with unit operations for the production and purification of a specific product. There, practitioners might use several PSE techniques to design and synthesise (Westerberg, 2004), operate (Venkatasubramanian, Rengaswamy, & Ka, 2003; Venkatasubramanian, Rengaswamy, Yin, & Ka, 2003), analyze (Wan et al., 2005), and optimize (Biegler & Grossmann, 2004) such a process. Since Sargent's statements in the 1980s, the toolbox to tackle problems in these fields has been growing continuously, which was greatly supported by advancements in modeling and data analytics (Bhosekar & Ierapetritou, 2018; Qin, 2014; Reis & Saraiva, 2021), optimization (Alcántara Avila et al., 2021; I. E. Grossmann, 2021), and machine learning (J. H. Lee et al., 2018; Perera et al., 2019). The gained knowledge over those decades allows the creation of solutions for optimal tactical decision-making in fields like scheduling (Badejo & Ierapetritou, 2022; Díaz-Madroñero et al., 2014), enterprise-wide optimization (I. Grossmann, 2005; I. E.

Grossmann, 2012, 2014; Van De Berg et al., 2023; van de Berg, Petsagkourakis, et al., 2023; van de Berg, Shah, & Del Rio Chanona, 2023; Q. Zhang & Grossmann, 2016), or supply chain management and resource allocation (Barbosa-Póvoa, 2012; Berning et al., 2004; Garcia & You, 2015; Guillén-Gosálbez, Mele, et al., 2006; Lasschuit & Thijssen, 2004; Nikolopoulou & Ierapetritou, 2012; Papageorgiou et al., 2001; Susarla & Karimi, 2011), even in the case of disruptions or general uncertainties (Applequist et al., 2000; Guillén-Gosálbez, Badell, et al., 2006; Z. Li & Ierapetritou, 2008; Z. Li & Ierapetritou, 2008; Lin et al., 2004; Pistikopoulos, 1995).

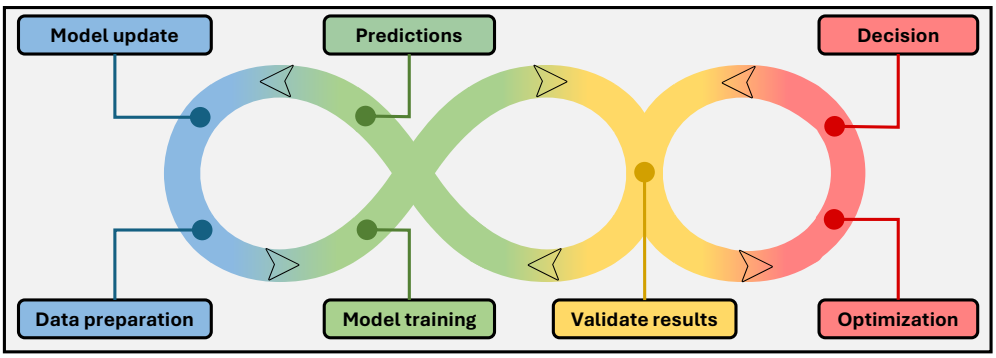

Figure 1.3. Visualization of the interaction of modeling and optimization for decision-making. The interconnections between the different steps should not be regarded as conclusive, where of course also other intermediate steps and shortcuts might be possible. The figure should give a general overview of how modeling and optimization might be linked. The concept was adapted from Van Den Heuvel and Tamburri (2020).

1.3 State-of-the-art, challenges and research gaps

As our society develops, the demand for more sustainable and efficient processes and products from a greater variety is growing. Industry therefore faces challenges on many technical and conceptual scales, reaching from understanding a material to efficiently design experiments (Scotti et al., 2023) or intensify cell culture processes (Del Rio Chanona et al., 2017; Kyriakopoulos et al., 2018; Potvin et al., 2012; D. Zhang, Dechatiwongse, del Rio-Chanona, et al., 2015) up to improve process units and entire production processes (Biegler et al., 1997; Edgar et al., 1988; I. E. Grossmann, 2021; R. Smith, 2005, 2016) or supply chains (Garcia & You, 2015). To address these holistic challenges, mathematical modeling and optimization techniques might be used as key enabling tools. Therefore, the reader will subsequently be introduced to those topics in more detail.

1.3.1 Mathematical modeling to understand chemical and biological process systems

Mathematically representing and simulating a material or a chemical reaction is an important step in building industrial digital applications to support the understanding and enable the optimization of production processes. Mathematical modeling can bring insights into the material behavior and can help to understand the relationships between the observed performance and the physical properties (Olson, 1997). Understanding a material in detail subsequently enables the prediction of its behavior in different conditions, which consequently allows the optimization of a process to enhance its performance (Scotti et al., 2023). Having a model of a system at hand also significantly contributes to accelerating the process development by reducing the need for expensive and time-consuming experimental steps (Scotti et al., 2023). These savings are certainly important in high-value production processes such as in the pharmaceutical industry, where resources are scarce and expensive (Kroll, Hofer, Ulonska, et al., 2017; Mercier et al., 2014; Von Stosch et al., 2021; Walsh, 2018), and where a vast number of new products is rapidly developed (Narayanan et al., 2023). Several recent works highlight the importance of modeling approaches across a wide range of applications within the chemical and bioprocess industry: Filho et al. (2020), for example, describe the development of a process model which supports the scale-up for an aerosol-assisted chemical vapour deposition process. Another important aspect of modeling in industrial environments is given by Elnashaie and Elshishini (2022). The authors discuss reactor models and how they can be used in the design, operation and optimization of industrial reactors. They describe the concept of reactor modeling as a method to perform predictions and analyses based on available data and information. Of course, these examples do not give an exhaustive list where mathematical modeling is useful, but they should rather give an overview of the importance of modeling as a tool that can support and enhance the digital transformation in the chemical and bioprocess industry.

To classify a mathematical model, different definitions have been proposed (Sansana et al., 2021). In this work, models are conceptually classified to belong to one of three possible groups (Bonvin et al., 2016). On the one hand, a model might be built up from knowledge. Such a model might be described by a mechanistic, or white-box model. On the other hand, a model might be generated by using data-driven approaches, also described by black-box models. The two classes might be mixed, leading to hybrid models, which are sometimes also described by the term grey-box models (Glassey & Von Stosch, 2018; von Stosch et al., 2014).

Mechanistic modeling An example of a well-known mechanistic model with long-lasting academic and industrial relevance was the description of enzymatic reactions by Michaelis and Menten (1913). This work allowed studying and understanding biocatalytic reactions in depth in the subsequent decades, up until today (Cornish-Bowden, 2015; Johnson & Goody, 2011). In literature, many other works explore mathematical biochemical models, for example, to analyze and understand intracellular signalling pathways (Klipp & Liebermeister, 2006) and metabolic mechanisms that drive the behavior of the organism (Guillén-Gosálbez et al., 2013; Mercier et al., 2014; Petsagkourakis, Sandoval, et al., 2020; D. Zhang et al., 2020), to describe the time-dynamic behavior of a biochemical system (Voit, 2013), to study the cascade activities of proteins (Schoeberl et al., 2002), or to even deepen the understanding in how diseases such as Alzheimer progress (Götz et al., 2018) and how they could be treated (Scearce-Levie et al., 2020; Van Dam & De Deyn, 2011). Of course, such modeling methods are not limited to biological systems like the ones mentioned above, but they can also certainly be used to study procedures like freeze drying (Srisuma et al., 2024) or chromatographic steps (Hahn et al., 2023), to describe the production of viral particles for pharmaceutical applications (Canova et al., 2023), or to model the kinetics of ammonia (Chehade & Dincer, 2021) and methanol (Nestler et al., 2020) production units. Such methods often involve the mechanistic formulation of a model where the involved practitioner is required to specify the behavior of the system and to define some parameters, which are subsequently estimated by using available experimental data. Sha et al. (2018), Gosálbez et al. (2013) and Voit (2000) describe the steps and methods that might be involved in setting up such mechanistic models for a biological system. The parameter estimation steps – often also described by the term inverse problem or model calibration (J. Sun et al., 2012) – might not always be a straightforward task, as the modeler might face challenges like missing or noisy experimental data (Lillacci & Khammash, 2010), multi-modal and nonlinear models (Rodriguez-Fernandez et al., 2006) or lack of robustness in the estimation of parameters (Gábor & Banga, 2015; J. Sun et al., 2012). De Carvalho Servia and Del Rio Chanona (2023b) describe several metrics that might be considered when performing such a parameter estimation step, where they also analyse the impact of the noise and the sampling frequency. According to Michalik et al. (2009), there are mainly two approaches that can be used to estimate parameters in mechanistic models, namely a sequential and a simultaneous approach. Both approaches often require a given mathematical structure of the model, where the involved model parameters are estimated by using an appropriate loss function. Both approaches also show challenges that need to be overcome: In the sequential approach a modeler might encounter the possibility of high computational efforts or the occurrence of stiff ordinary differen-

tial equations (ODEs), whereas, in the simultaneous approach, one might need to use reformulation techniques such as orthogonal collocation (Carey & Finlayson, 1975) to transform the dynamic equations into a large system of equations (Esposito & Floudas, 2000; Miró et al., 2012).

Data-driven modeling Instead of using mechanistically designed models, data-driven approaches offer a direct and efficient way to model a process if the available knowledge is limited (Taylor et al., 2021). Recent advances the field of machine learning (ML) made it possible to create data-driven models for a wide range of applications. Helleckes et al. (2023) and Lee et al. (2018) provide an overview of how ML models might be useful for the bioprocess development or the general PSE field, respectively. Data-driven models were already successfully applied several years ago, where, for example, Willis et al. (1995) used artificial neural networks (ANNs) to model the relationships between the measured data and the biomass concentration. Similarly, but more recent, Gaussian processes (GP) were used to describe the growth of microbes (Tonner et al., 2017). In an even more recent work by Del Rio Chanona et al. (2019), the authors leveraged the predictive capabilities of GPs to model a wastewater treatment procedure with algae and bacteria. Some researchers also used ML-based surrogate models to approximate mechanistic models: Renardy et al. (2018), for example, replaced mechanistic simulations with a polynomial surrogate model to improve the computational efficiency of a given modeling approach (Gherman et al., 2023). A similar work was published by Wang et al. (2019), who used a long short-term memory (LSTM) network to model transduction pathways in cell-cycle progressions, where their proposed framework was able to enhance the computational efficiency by orders of magnitude compared to mechanistic modeling approaches. Further, Mowbray and colleagues (2021) offer a deep dive into the different kinds of models that might be used as a black-box to model the system under study. Of course, data-driven modeling strategies are not only of high interest in the bio-related industry. Bishnu et al. (2023) recently published an overview about applications with data-driven modeling in process systems. In there, studies that leverage data-driven strategies are showcased for the petrochemical production optimization (Min et al., 2019), the prediction of catalyst saturation levels (Steurtewagen & Van Den Poel, 2020), and the modeling of hydrocracking processes (Elkamel et al., 1999).

Hybrid modeling With increasing popularity, hybrid modeling approaches are used to bridge the gaps between such white and black-box models by combining mechanistic and surrogate components (von Stosch et al., 2014). Mechanistic knowledge about the system under study is complemented by the data-driven part that leverages available process data. Tsopanoglou and Jiménez Del Val (Tsopanoglou & Jiménez Del Val, 2021)

describe hybrid modeling strategies as an emerging key tool in the era of biopharma 4.0, hence playing a pivotal role in advancing the digitalization within this industry. They further provide an overview of the advantages and challenges of coupling mechanistic and data-driven approaches, specifically within the pharmaceutical industry. Such hybrid approaches were used to solve a wide range of problems, like the modeling of specific kinetics (S. Zhang et al., 2013) or entire biochemical process systems (Gnoth et al., 2007, 2008b; M. R. Mowbray et al., 2023). Hybrid models were also successfully applied in the field of computational fluid dynamics (CFD): Mosavi et al. (2019) introduced a combination of a network-based model with a CFD model to enhance the accuracy of flow characteristics predictions in bioreactors. In the field of biohydrogen production, a recent publication by Pal et al. (2024) introduced a model based on a Bayesian neural network (BNN) that allowed to accurately model the concentration evolution of involved species in a photobioreactor. In another very recent work, De-Luca et al. (2024) developed an ANN-based hybrid model to support the design and testing of operating conditions during adeno-associated virus particle production, a field that has gained significant interest over the past few years (C. Li & Samulski, 2020). In many works published in this area, ANN models are very commonly used as data-driven components, where Mahanty (2023) and Agharafeie and colleagues (2023) provide an overview and a systematic literature review about applications of such hybrid neural models for biological process systems. Of course, hybrid models are not solely used to model systems tinged with biological elements. Additional examples where hybrid models were successfully used are given in the work by Azarpour et al. (2017), where the authors combined mechanistic first principle models with ANNs to develop generic frameworks for modeling industrial catalytic reactors that consider catalyst deactivation. A recent study by Lastrucci et al. (2024) investigated the applicability of hybrid models based on physics-informed neural networks (PINNs) for the design of industrial catalytic reactors. Using PINNs for solving ODEs, the authors achieved significant speed improvements compared to classical ODE solvers, while maintaining high levels of the system modeling accuracy. Further, in the work by Chen and Ierapetritou (2020), a framework based on hybrid models was developed that captures plant-model-mismatches, where the authors showcase the advantages of these modeling approaches in the context of pharmaceutical unit operations. Furthermore, Quaghebeur and colleagues (2022) created a framework that introduces data-driven components into existing mechanistic models of a water resource facility, thereby being able to replace parts of the system that are hard to model. One important aspect of hybrid models is that they can also be developed in a flexible manner, allowing to customize the degree of hybridization. Narayanan and colleagues (2021), for example, showcase this aspect of different hybridization degrees (Figure 1.4) by com-

paring the performance of different hybrid models applied to a chromatographic case study. The flexibility, transferability from one case study to another, the combination of process knowledge with powerful data-driven components, and many more reasons are making hybrid modeling approaches suitable for industrial applications, where several companies mention their active development for production (Datahow AG, 2024; Merck KGaA, 2024; Sartorius AG, 2024).

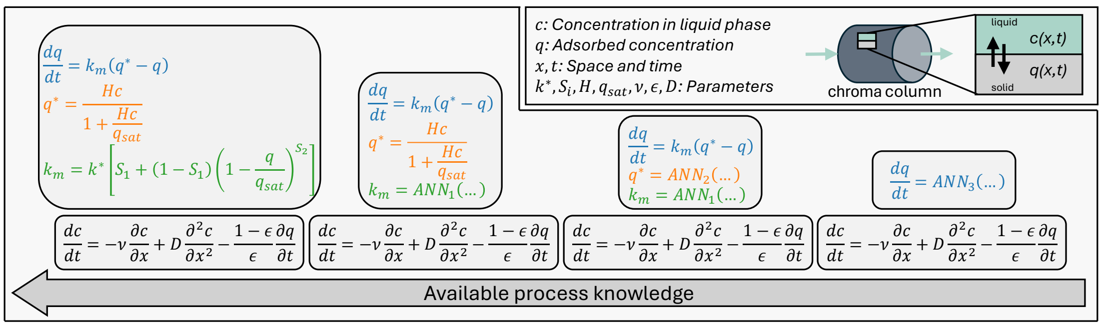

Figure 1.4. Possible hybridization levels in the application of a hybrid model for the modeling of a chromatographic process. The concept and figure were adapted from Narayanan et al. (2021). In addition to the mass balance of the species in solution, different parts of the model are replaced by data-driven components, leading to hybrid models with a different degree of hybridization.

No matter which kind of modeling approach – mechanistic, data-driven, or hybrid – is chosen to identify a suitable representation for the system under study, two important key steps need to be executed during the model identification process. First, some kind of model structure needs to be defined or identified, and second, the involved parameters need to be estimated, which brings the modeler back to either the simultaneous or sequential parameter estimation approach discussed before. Ideally, the model structure and its parameters should be simultaneously determined since the choice of a specific model structure limits the accuracy of the model and vice-versa. In the past few years, many algorithms and approaches have been presented in the scientific community that allow modeling a system without expert knowledge (so just relying on available data), while returning a closed-form algebraic description that is interpretable (J. Lee et al., 2024): Among others, the automated learning of algebraic models for optimization (ALAMO) approach (Cozad et al., 2014; Wilson & Sahinidis, 2017), the sparse identification of nonlinear dynamics (SINDy) approach (Brunton et al., 2016) with its recent combination with an extended Kalman filter (EKF-SINDy) (Rosafalco et al., 2024), the algebraic learning via elastic net (ALVEN), the recently published LASSO-Clip-EN (LCEN) by Seber and Braatz (2024), or the recently published Bayesian identification of

dynamic sparse algebraic models (BIDSAM) approach by Adeyemo and Bhattacharyya (2024) were developed and successfully deployed in a variety of case studies. Without going into too many details, the algorithms seek an appropriate surrogate model, where model identification and a parameter estimation step are combined, so the outputs of the algorithms are closed-form mathematical equations. Since those methods rely on some chosen sets of basis functions or mathematical expressions, and they usually do not assume any knowledge about the process, they might lead to models that are hard to interpret or may return structures that do not have a direct physical interpretation. A slightly different approach to search for a closed-form model for a process system is based on the principles of genetic programming (Keane et al., 1993; Koza, 1994), where mathematical equations are created as symbolic expression trees (Cozad and Sahinidis, 2018). Employing a defined search procedure – which is mainly stochastic – such a symbolic regression (SR) approach simultaneously identifies the mathematical structure of the model while it also estimates the involved parameters. Due to this, such expression-tree-based approaches only require a pool of allowed mathematical operators and some measured data points. SR has been successfully applied in various fields, for example, to model a vacuum distillation column and a chemical reactor system (McKay et al., 1997), to predict energy outputs from wind farms (Vladislavleva et al., 2013), to discover physical relationships in an available dataset (Schmidt & Lipson, 2009), to discover new catalysts (Weng et al., 2020), to model kinetics in catalytic systems (De Carvalho Servia et al., 2024), to enhance the feasibility in which thermodynamic models can be applied reliably (Kay et al., 2024), or to even recover psychological models for human information processing (Hewson et al., 2023). SR was also included in commercial (i.e., Eureqa (Schmidt & Lipson, 2009) or TuringBot (2023)) or open-source (i.e., PySR (Cranmer, 2020), the Bayesian machine scientist (Guimerà et al., 2020)) software. Although there are many advanced modeling algorithms available, such as ANNs or GPs (M. Mowbray et al., 2021), having a compact, closed-form mathematical model at hand might be advantageous, depending on the framework the model should be deployed in. One aspect is that derivative-based, state-of-the-art and off-the-shelf solvers can be used, which, if global solvers are applied, can even guarantee global model-based optimality compared to heuristics or stochastic solvers.

1.3.2 Mathematical optimization to enhance process performance

Besides mathematical modeling, optimization can serve as a powerful tool for a practitioner to enhance process performance across various domains. At its core, mathematical optimization entails the formulation of models that are solved to find the best possible solution to a given problem within specified constraints. In the realm of chemical pro-

cesses, optimization techniques play a pivotal role in maximizing production efficiency, minimizing costs or emissions, and ensuring product quality.

Practitioners might make use of a wide range of methods to optimize a target objective when an optimization problem is encountered. Usually, such an optimization problem can be described by expression (1.1), where $f(x, y, \theta)$ is the defined objective function that depends on some variables x (continuous in $\mathbb{R}$, n-dimensional) and y (integer, in $\mathbb{Z}$, m-dimensional), and some given parameters θ. Additionally, it is common that optimization problems require the inclusion of constraints in the form of equalities ($h(x, y, \theta)$ of set I) or inequalities ($g(x, y, \theta)$ of set J), respectively, which might represent for example process or safety limitations.

$$\begin{aligned} \min_{x,y} \quad & f(x, y, \theta) \\ \text{s.t.} \quad & h_i(x, y, \theta) = 0, \quad \forall i \in I \\ & g_j(x, y, \theta) \leq 0, \quad \forall j \in J \\ & x \in \mathbb{R}^n, y \in \mathbb{Z}^m \end{aligned} \tag{1.1}$$

Such optimization problems can be solved using mathematical programming approaches. Depending on the nature of the objective and the constraints, a practitioner might use – but is not limited to – linear programming (LP), nonlinear programming (NLP), mixed-integer linear programming (MILP), or mixed-integer nonlinear programming (MINLP) methods. Grossmann (2021), Cavazzuti et al. (2013), and Hillier and Lieberman (2010) guide through these approaches from the scientific, PSE and operations research perspectives, respectively. To give some specific examples across the entire physical dimensions where such optimization approaches might be applied, the reader is referred to the publication of Esposito and Floudas (2000), which discusses the application of global optimization to estimate parameters in differential algebraic systems, to the work of Grossmann and Halemane (1982) who describe the design of flexible chemical plants, or the one of Duran and Grossmann (1986a) for insights into heat integration and optimization of chemical processes. Other interesting works that apply similar tools were published by Liu and colleagues (1996), who tackle the problem of selecting processes and planning expansions of a chemical complex, or by Esposito and Floudas (2002), who investigate an isothermal reactor network synthesis using mathematical programming. In a more recent publication by Lasschuit and Thijssen (2004), the authors use an MINLP model to support the decision-making process for supply chain optimization questions.

By leveraging such optimization techniques, production processes can be streamlined, energy consumption minimized, resource utilization maximized, and overall productivity enhanced. This ultimately leads to improved profitability and competitiveness in the dynamic landscape of the chemical industry (Dutta, 2016). Within this landscape, a major challenge for practitioners is located in the operational optimization of existing processes. Originally, process optimization relied on models that are based on engineering knowledge, where such models provide closed-form descriptions that enable the application of off-the-shelf deterministic optimization algorithms (Bongartz & Mitsos, 2019; Haydary, 2019). However, as discussed above, developing such mechanistically designed models might not be an easy task, depending on the problem structure. At this point, some of the above-mentioned data-driven model-building approaches (i.e., the ALVEN (W. Sun & Braatz, 2020) or ALAMO (Cozad & Sahinidis, 2018) approach) might be very useful tools that can be considered. Such approaches can be able to identify closed-form approximations for the process by taking observed data into account, and therefore allow to bypass the development of mechanistic descriptions from scratch. Hüllen et al. (2020), for example, successfully demonstrate how such a procedure can be used to solve an optimization problem related to direct air capture (DAC). The authors develop an optimization problem that would require algebraic equations and the formulation of a large nonlinear programming problem. Instead, they combine polynomial approximations with observed data and incorporate the model into the original optimization problem. The main advantage of those tools compared to the mechanistic description of the process is the possibility to develop a model in a relatively short time, which can then also more easily be incorporated in an existing optimization framework. However, these methods might constrain the model structure by relying on predetermined monomials and transformations of the input variables, likely leading to less accurate approximations and therefore representing a significant drawback, depending on the system under study.

As discussed in Section 1.3.1, data-driven models emerged to deal with modeling tasks in which the underlying phenomena cannot be easily described. Within the optimization landscape, data-driven methods have been used to build surrogates of mechanistic models that are hard to identify and/or optimize. For example, Jones et al. (1998) used a response surface methodology for expensive multimodal functions and applied Bayesian optimization to find the optimum solution. Other and more recent examples were given by Quirante et al. (2015, 2018) and Quirante and Caballero (2016), where the authors used surrogate models to optimize distillation columns and other units. Taking these outstanding works as examples, the advantage of data-driven models is very well observable.

Although showing great promises, what remains challenging is the fact that – depending on the system under study – incorporating such sophisticated modeling methods into optimization frameworks might not be straightforward due to their intrinsic complexity and nonlinearities. Nevertheless, several algorithmic frameworks were developed that allow the incorporation of advanced machine learning models into an optimization formulation. The team around Mitsos and colleagues (2009; 2019; 2020) developed MeLON (Machine Learning models for Optimization) and MAiNGO (McCormick-based Algorithm for mixed-integer Nonlinear Global Optimization), tailored algorithms that – in a nutshell – allow to perform deterministic global optimization with data-driven models. A similar approach was developed by Ceccon et al. (2022), who presented OMLT (optimization and machine learning toolkit), which allows to optimize trained ANNs or gradient-boosted trees using the Python-based framework Pyomo (Bynum et al., 2021; Hart et al., 2011). Boukouvala et al. (2017) introduced the algorithmic framework ARGONAUT that allows to globally optimize general constrained grey-box problems using the ANTIGONE (Misener & Floudas, 2014) solver. In a recent work, Paulson and Lu (2022) proposed the COBALT (constrained Bayesian optimization of computationally expensive grey-box models exploiting derivative information) algorithm for constrained grey-box optimization problems, combining GP models with state-of-the-art optimizers (2021).

In contrast to deploying advanced machine learning models for subsequent optimization tasks, closed-form expressions – as the resulting models of the above-mentioned SR approaches – have the advantage that they easily allow the use of deterministic optimization algorithms (Androulakis et al., 1995; I. E. Grossmann, 1996; Ryoo & Sahinidis, 1995; E. M. B. Smith & Pantelides, 1997; Tawarmalani & Sahinidis, 2002; Zamora & Grossmann, 1999). This is especially promising in the field of global optimization (GO), where, in case the objective function or constraints are unavailable, deterministic optimization is not possible since the algebraic expression of the function and therefore its corresponding derivatives are not available. In such cases, other methods – for example, derivative-free optimization algorithms and heuristic methods like the Nelder-Mead method (Nelder & Mead, 1965), Bayesian optimization (Močkus, 1975; Shahriari et al., 2016), differential evolution (Price, 2013), genetic algorithm (Holland, 1992; Mirjalili, 2019), particle swarm optimization (Bonyadi & Michalewicz, 2017; Kennedy & Eberhart, 2006), or simulated annealing (Hwang, 1988; Van Laarhoven & Aarts, 1987) – methods might be applied (Bradford et al., 2018). The disadvantage of such methods is, however, that they are not guaranteed to identify the global optimum, whereas deterministic GO methods are guaranteed to identify the global solution – within a given ϵ-tolerance – in

a finite number of iterations (Androulakis et al., 1995; Horst & Tuy, 1996).

In the above-mentioned introduction to the field of optimization, one very important aspect was not yet mentioned, which is the presence of uncertainty. Mostly, the real world is affected by some kind of stochasticity, which means that the outcome of a process is influenced by some uncertainty. In such cases, practitioners might make use of approaches such as stochastic programming (SP) (Birge & Louveaux, 2011; Ierapetritou & Pistikopoulos, 1994; Z. Li & Ierapetritou, 2012; Marti & Kall, 1995; Prekopa, 1995; Shapiro et al., 2021) or robust optimization (RO) (Ben-Tal et al., 2009; Ben-Tal & Nemirovski, 2002; Z. Li & Ierapetritou, 2008; Z. Li & Ierapetritou, 2008; Lin et al., 2004) to account for uncertainties in a decision-making process. These methods are building up on and leveraging the above-mentioned programming methods and enable the development of strategies that can withstand variations in input parameters, market conditions, or external factors. By incorporating uncertainty considerations into the optimization framework, practitioners can make informed decisions that balance risk and reward, leading to more sustainable and resilient operations in the chemical industry (Sahinidis, 2004).

1.3.3 Consideration of uncertainty for flexible processes

Uncertainties pervade various stages, posing significant challenges for practitioners in different industry-related fields. These uncertainties can arise from diverse sources, including variations in raw material properties (Sharifian et al., 2021), inherent complexity in biochemical reactions (Mišković & Hatzimanikatis, 2011), or in fluctuations of environmental conditions and product demands (Gabrielli et al., 2019). These variations or disturbances can then propagate and be revealed in the process design and operation phase (Pistikopoulos, 1995) or even in conceptually different stages like supply chain and scheduling activities (Ehrenstein et al., 2019; Ovalle et al., 2024). Managing uncertainties – also considering different scales – is crucial to ensure safety, process robustness, and reliability (**sarkis.etal_2023**; I. E. Grossmann et al., 1983). If uncertainties are not taken into account, designing and optimizing processes can lead to suboptimal or infeasible solutions (I. E. Grossmann et al., 1983). Shimoni et al. (2014), for example, describe the edge of failure in a pharmaceutical application. It describes a point, which, if exceeded, could result in deviations that might have impact on the quality of the product. Additional examples and an overview about how uncertainty might affect chemical processes in design, optimization, control, and fault detection is given by Sharifian et al. (Sharifian et al., 2021).

When a practitioner seeks an optimal design or operating solution, the identified realiza-

tion usually needs to meet a variety of requirements, where unconsidered perturbations might significantly impact the operability of the process. However, several challenges in considering uncertainties in optimization problems are faced by practitioners, which make the development of industrial applications difficult. The lack of information on the uncertainty of the data might be described as one of the major challenges in this field (I. E. Grossmann et al., 2016). Other barriers are certainly the difficulty in determining the nature of the uncertainties (exogenous vs. endogenous), the choice of strategy to prepare process systems to withstand variations due to uncertainty (robust/chance-constrained optimization vs. stochastic programming), and also the immense computational power required to consider uncertainty in calculations (often orders of magnitude larger than deterministic models) (I. E. Grossmann et al., 2016). Practitioners might make use of different approaches to consider uncertainties in their optimization problems, where the work published by Grossmann et al. (2017) offers a great overview of recent advances and possible applications of both, robust optimization and stochastic programming, within the landscape of the chemical industry. Sahinidis (2004) and Diwekar (2020) offer a more general overview of the application of stochastic programming, robust optimization, and many other methods such as decomposition methods (Dantzig & Wolfe, 1960), sampling-based approaches (Diwekar, 2020), fuzzy mathematical programming (Bellman et al., 1967; Zimmermann, 2001), or stochastic dynamic programming (Bellman, 2010). In the published work by Sahinidis (2004), different applications are given to which these mentioned approaches were applied.

Uncertainty also comes into play when a process design should be implemented as flexibly as possible (design stage, Figure 1.5), or when a fixed design should be assessed for its flexibility (operational stage, Chapter 5). The flexibility of a process system describes the ability of the system to adapt to changes in operating conditions or external factors (I. E. Grossmann et al., 1983). Based on concepts that were usually mainly applied during the process design phase (Pistikopoulos, 1995), a flexibility index was designed by the PSE community – with pioneering works by Grossmann et al. (1983), Halemane and Grossmann (1983), and Swaney and Grossmann (1985a, 1985b) – that allows to assess the ability of a design to remain feasible against variations in the parameter values during the operation. Alternatively, other metrics to quantify process flexibility were described in the literature, where the resilience index (Morari et al., 1985), stochastic design reliability (Kubic & Stein, 1988), or stochastic flexibility index (Pistikopoulos & Mazzuchi, 1990; Straub & Grossmann, 1990, 1993) are well-known. A more detailed overview of the flexibility index, its applications in the chemical industry, and how it can be assessed will be given in Chapter 5.1. Although the description of process

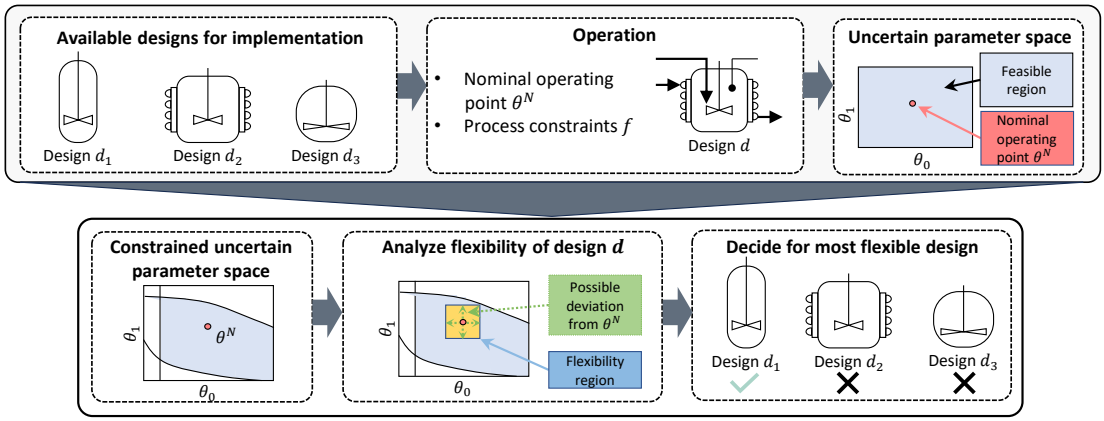

Figure 1.5. Schematically represented procedure to implement the most flexible design of a process equipment.

flexibility was already established in the 1980s, there are still remaining challenges that need to be addressed. One of the main issues is the computation of the flexibility index when the process dynamics are complex (Ding & Ierapetritou, 2021), when the constraints are hard to model, or when the only knowledge about the system consists of observations of input and output data due to limited process understanding (Boukouvala & Ierapetritou, 2012). In such cases, the flexibility index cannot be computed using deterministic mathematical models, as the constraints are not available in a closed-form algebraic manner. Some works were published where the authors showed an alternative possibility to assess the flexibility of the process by estimating the feasible region. Mostly, this is achieved by creating a function that evaluates the feasibility of the model for given values of the decision variables and the parameters (Boukouvala & Ierapetritou, 2012; Boukouvala et al., 2011; Metta et al., 2021), or by sampling approaches (Sachio et al., 2023, 2024). These methods that model the feasible region, however, are conceptually different approaches compared to the described assessment of the originally described calculation of the flexibility index (Halemane & Grossmann, 1983; Swaney & Grossmann, 1985a, 1985b). In other words, they approximate the feasibility function with a surrogate and they do not rely on the original deterministic flexibility index, but rather they use alternative flexibility metrices. Therefore, the challenge in computing the originally described flexibility index remains unsolved in case of complex process dynamics or hard-to-model constraints. One possibility to link the originally developed flexibility index formulation with such complicating constraints is to use approaches that result in closed-form algebraic models, which, as described above in Section 1.3.2, can be straightforwardly incorporated into deterministic optimization problems, such as the original flexibility index problem.

1.4 Motivation and objectives

The motivation for this thesis stems from the need to address the modeling and optimization challenges faced in the chemical and biological sector, which were introduced above. Herein, surrogate and hybrid approaches are discussed that leverage advancements in several areas of research, including machine learning, mathematical modeling, and optimization. A general goal of this thesis is the development of solution approaches for a wide range of modeling and optimization applications, where the provided frameworks should serve as alternative methods to existing ones that target the support of the design and optimization of chemical and biological processes. The following objectives should be addressed within this work:

(1) The development of purely mechanistic models for chemical and biological systems is a significant challenge, not only due to a possible lack of knowledge about the system. Even if a model can be developed, estimating the parameters of such a representation might pose challenging tasks, as discussed previously. Furthermore, pure data-driven models might face issues in interpretability and extrapolation. The first objective of the thesis is therefore to design models that are based on expert knowledge or available statistical prior information. However, instead of using a fixed structure of the model, it should be identified in parallel to the parameter estimation step. One advantage of such a procedure is that the resulting expressions can be more easily interpreted due to the conceptually incorporated knowledge. Further, such resulting models can be optimized in existing algebraic modeling systems due to the closed-form expressions.

(2) The second objective of the thesis is to develop a framework that allows to perform a deterministic optimization of processes using closed-form surrogate models. The framework should be able to build algebraic surrogates purely from data, which can subsequently be optimized using well-established and off-the-shelf deterministic solvers. The advantage of such a procedure is that the modeling and optimization steps can be decoupled, where the resulting models can be optimized in existing algebraic modeling systems due to the obtained closed-form expressions.

(3) Lastly, this thesis focuses on the development of a framework that allows to analyze the flexibility of processes using surrogate models. Algebraic surrogates – purely generated from data – should be incorporated into an optimization problem that can assess the flexibility of an existing process. The advantage of such a hybrid framework is that the modeling and optimization steps can be decoupled, while the flexibility analysis can be performed with existing well-known deterministic solvers.

1.5 Research contributions and outline

The thesis is divided into several chapters which guide through various conceptual and physical scales within the chemical and biological process systems value chain, from modeling of reactions up to the optimization of production processes. The chapters will highlight how mathematical tools can be used to contribute to the enhancement of efficiency, sustainability, and flexibility in the chemical industry while targeting the above-mentioned objectives (Figure 1.6).

As discussed above, the accurate model construction might be not straightforward, especially with bioprocesses, due to complex metabolic mechanisms and data scarcity. In Chapter 2 of this thesis, a modeling approach is discussed that uses a mechanistic mass balance backbone in the form of a canonical kinetic representation, which is well-known from the field of biochemical systems theory (BST). Together with observed data, an incremental approach is used to perform a parameter estimation step while avoiding complex integrations of ODEs. The obtained expressions can subsequently be more easily interpreted and optimized in existing algebraic modeling systems than for example purely data-driven models. This chapter therefore tackles the challenge of accurate model construction, data scarcity and the need for interpretable models. Building on Chapter 2, Chapter 3 of the thesis focuses on identifying suitable kinetic models for bioprocesses. Compared to the second chapter, however, here, an approach is used that identifies models without assuming a pre-defined model structure. With this method, kinetic rates are directly obtained from data and a two-step approach for the parameter estimation step. The obtained algebraic expressions for the rate equations simplify the model interpretation and subsequently allow the application of optimization algorithms. Chapter 4 tackles the challenge of combining surrogate models with global optimization of processes by building algebraic surrogates from data, which can subsequently be op-

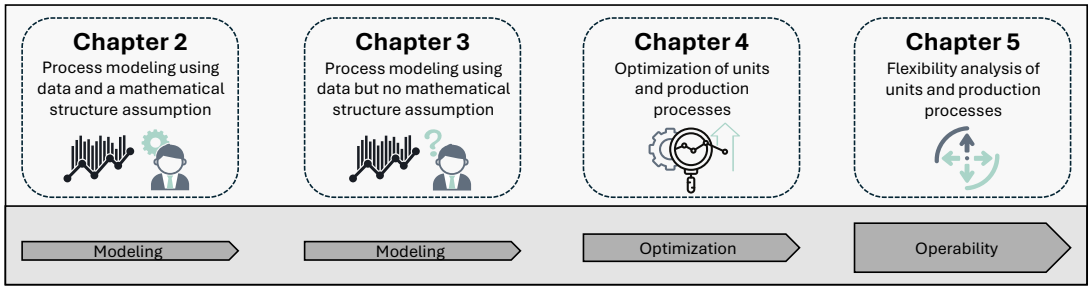

Figure 1.6. A schematic structure of the chapters of this thesis. The arrows below the blocks represent the main topic touched throughout the corresponding chapter.

timized using deterministic solvers. Deterministically optimizing a given system usually requires algebraic descriptions of a process. The advantage is that these models can easily be optimized due to their closed-form structure. On the other hand, it might not be straightforward to write sophisticated surrogate models like ANNs or GPs in a closed-form manner. After having optimized an existing production process for its operating conditions, it could be analyzed for its flexibility. There might be many reasons why standard approaches for analyzing the flexibility can be challenging to apply. In Chapter 5 of this thesis, the traditional flexibility index is calculated using a hybrid approach, in which complicating parts of the optimization model are replaced by algebraic surrogate models. This simplifies the flexibility analysis of complex process models since, although surrogate models are used, the application of deterministic solvers becomes possible. This approach provides an alternative way to existing methods to analyze the flexibility of processes entailing complicating constraints. Lastly, in Chapter 6, the discussions of the previous chapters are summarized. The main findings are highlighted, and the potential of the developed models and frameworks is discussed. The chapter concludes with an outlook on future research directions and potential applications of the developed models and frameworks.

Chapter 2

Modeling via MINLP-based symbolic regression of S-system formalisms

This chapter is based on the following publication: Forster T., Vazquez D., Cruz-Bournazou M. N., Butté A., Guillén-Gosálbez G. (2023). Modeling via MINLP-based symbolic regression of S-system formalisms. *Computers and Chemical Engineering*, 170, 108108.

Nomenclature for Chapter 2

Sets

E	$\{e \mid e$ is a batch experiment$\}$
I	$\{i, j \mid i$ and j are species/components$\}$
M	$\{m \mid m$ is a node in the hidden layer of the neural network$\}$
R	$\{r \mid r$ is a chemical reaction$\}$
U	$\{u \mid u$ is a discrete/sampled time point$\}$
$\mathbb{R}$	Real numbers

Parameters

$f_{j,r}$	Reaction rate order of species j in reaction r (used in the GMA)
$g_{i,j}$	Reaction rate order of species j influencing species i (production term in S-system)
$h_{i,j}$	Reaction rate order of species j influencing species i (depletion term in S-system)
k_r	Reaction rate constant of reaction r (used in GMA)
NP	User-defined maximum number of non-zero parameters in the S-system
$Q_{i,in}$ and $Q_{i,out}$	Material flow into/out of a reactor
t_0, t_f	Initial and final time
$X_{0,i}$	Initial concentration of metabolite/species i
μ	Mean of a particular property
σ^2	Variance of a particular property
w and b	Weights and biases of the ANN
α_i	Production reaction rate constant of reaction r (used in S-system)
β_i	Depletion reaction rate constant of reaction r (used in S-system)
$\gamma_{i,r}$	Stoichiometric coefficient of species i in reaction r

Variables

X_i	Concentration of metabolite/species i (used as continuous variables in ODE expressions)
$X_{i,u}$	Concentration of metabolite/species i at time t_u
$\hat{X}_{i,u}$	Model predictions of the concentration of metabolite/species i at time t_u
$\dot{X}_{i,u}$	Derivatives/slopes of metabolite/species i at time t_u
$\hat{\dot{X}}_{i,u}$	Model predictions of the derivatives/slopes of metabolite/species i at time t_u
$\overline{X}_(i, u)$	Mean of the experimental data points of species i at time t_u
t and t_u	Time and sampled time point

2.1 Introduction

In recent years, model-based methods have gained increasing attention in the chemical and pharmaceutical industries (Narayanan et al., 2019), finding applications in process development, design, optimization, monitoring, and control. Among other advantages, mathematical modeling techniques and simulations help guide experiments more effectively and monitor/control processes in a predictive manner, thereby reducing the associated development or process costs. These savings are particularly relevant in the biopharmaceutical industry, where resources are scarce and expensive (Kroll, Hofer, Ulonska, et al., 2017; Mercier et al., 2014). The modeling of biotechnological processes is particularly challenging due to the difficulty of precisely describing the underlying metabolic mechanisms dictating the microorganisms' behavior (Guillén-Gosálbez et al., 2013; Mercier et al., 2014; Petsagkourakis, Sandoval, et al., 2020; D. Zhang et al., 2020). Since these pathways are nevertheless the critical enabler for producing therapeutic compounds, they need to be modeled accurately so the associated process can be optimized effectively (Narayanan et al., 2019). However, since parameter estimation requires a model to which the parameters are fit, the fact that the behavior of the bioprocesses is not well understood directly affects the difficulty of the parameter estimation step.

Ideally, first-principles models combining well-established equations with available data (semi-empirical or deterministic) could support efficient and cost-effective development for therapeutic new drugs (von Stosch et al., 2014). The model-building journey in bioprocesses often encompasses several steps, as described by Gosalbez et al. (2013), and Voit (2000), from identifying the mass flow structure (stoichiometry) through selecting a kinetic representation and, finally, estimating the intrinsic parameters. Here, choosing a suitable kinetic formalism and calibrating it with experimental points are particularly challenging tasks involving solving nonconvex dynamic optimization problems.

Parameter estimation for model building in dynamic systems can be solved either sequentially or simultaneously (Michalik et al., 2009). Both approaches, which often assume a given fixed mathematical structure, attempt to minimize a loss

function, such as the squared residuals. In the sequential approach, the objective function is calculated by integrating the differential model and comparing the predictions to the observed values. The intrinsic model parameters are adjusted, and the procedure is performed iteratively until convergence. Examples of such a procedure can be seen in Bellman et al. (1967) and Kim et al. (1991). The drawback of this approach is its high computational effort due to the need to integrate ordinary differential equations (ODE) in every iteration.

Additionally, this approach can lead to systems that are hard to solve due to the resulting stiff ODEs (Tjoa & Biegler, 1991). In the simultaneous approach, the parameter estimation task can be reformulated as a nonlinear programming (NLP) problem. This reformulation can be performed by applying orthogonal collocation methods (Esposito & Floudas, 2000), as shown in the case of bioprocesses in the works of Gosálbez et al. (2013), Miró et al. (2012), Willis and von Stosch (2017), and Burnak et al. (2020). This approach is also discussed in Bansal et al. (2003), which covers formulations and algorithms for solving dynamic optimization problems. In summary, the simultaneous approach avoids the computational cost of integrating ODEs at every iteration. However, this strategy requires solving complex NLPs, for which finding a good starting point might be challenging. Other alternative approaches were proposed for model building of dynamic systems, like the one by Brendel et al. (2006) based on a unifying incremental identification concept for the stepwise identification of structured submodels in complex reaction systems that aims to reduce the computational burden. We stress that algorithms for model building that also optimize the model structure in addition to its parameters are scarce, although there is increasing interest in developing such approaches (Guillén-Gosálbez et al., 2013; Henriques et al., 2015; Wilson & Sahinidis, 2019).

The mentioned methods often rely on mechanistic models that are hard to develop. Notably, the exact equations describing the phenomena one wishes to model, particularly those dictating the kinetic behavior of a bioreactor, are often unknown. As an alternative to mechanistic models, data-driven strategies allow studying the system's behavior without relying on expert knowledge (Kahrs & Marquardt, 2007; Taylor et al., 2021). For example, Willis, Montague, and Peel (Willis et al., 1995) used an artificial neural network (ANN) to model the relationships between the measured online data and biomass concentration. More recently, mathematical programming methods were applied to build surrogate models, like the auto-

mated learning of algebraic models for optimization (ALAMO) approach (Wilson & Sahinidis, 2017). This algorithm seeks an optimal surrogate model by solving a mixed-integer linear programming (MILP) model. In a similar spirit, symbolic regression (SR) uses expression trees to simultaneously identify the model structure and the values of its parameters (Cozad & Sahinidis, 2018; Neumann et al., 2020). For dynamic systems, Brunton et al. (2016) proposed the sparse identification of nonlinear dynamics (SINDy) algorithm, which was successfully applied to different systems, ranging from simple canonical systems to the fluid vortex shedding behind an obstacle. This work was later extended by Rosafalco and colleagues (2024), who combined with an extended Kalman filter with the SINDy (EKF-SINDy) approach. This led to an easy-to-impelement and computationally efficient system for identifying nonlinear systems. In recent work, Cozad and Sahinidis (Cozad & Sahinidis, 2018) introduced a mixed-integer nonlinear programming (MINLP) formulation to solve SR problems to global optimality. Sun and Braatz (2020) developed an algorithm that combines nonlinear feature generation followed by sparse regression to learn interpretable nonlinear models, called algebraic learning via elastic net (ALVEN). These process models have the advantage of only requiring data and, therefore, can be set up without any deep knowledge of the system. However, they are hard to interpret and may return values that are not consistent with the physical meaning of the data due to the few mechanistic constraints imposed during training.

As a bridge between the discussed deterministic and data-driven methods, hybrid modeling (also referred to as grey-box modeling) has recently gained popularity as an appealing strategy to exploit the complementary strengths of both modeling paradigms. In essence, hybrid models combine a mechanistic backbone with a surrogate component. The mechanistic backbone of the model is determined *a priori* on the basis of the knowledge available about the process. It reveals a physical or empirical interpretation depending on the level of knowledge that is included. On the contrary, the surrogate component is determined from the data available. Hybrid modeling combines both extremes (von Stosch et al., 2014). Hybrid approaches were used to solve a wide range of problems. Zhang, Androulakis and Ierapetritou (2013) proposed a hybrid kinetic mechanism where quasi-steady-state species are separated from the kinetic ODEs and described by a set of nonlinear algebraic equations. Boukouvala and Floudas (2017) presented ARGONAUT, a framework for the global optimization of general nonlinear con-

strained hybrid/grey-box models designed for problems with an objective function and/or constraints without an explicit analytical expression, which are adequately approximated by the algorithm.

With regards to hybrid models applied to chemical/biochemical systems, Azevedo et al. (2019), Psichogios (1992), and Gnoth et al. (2007, 2008a, 2010) integrated ANNs as a non-parametric component in kinetic models. Notably, complex ANN architectures without regularizations relying on small datasets were shown to lead to less accurate predictions in these applications (Pasupa & Sunhem, 2016). It should be noted that, so far, there is no universal recipe for building hybrid models for bioprocesses, which may differ in the level of hybridization attained, i.e., the amount and type of first principles and data-driving components they combine (Narayanan, Luna, et al., 2021).

One way to improve the hybrid modeling of (bio)chemical reactions consists of using canonical mathematical formalisms flexible enough to represent a wide range of systems. The idea here is to replace black-box surrogates with semi-empirical models based on general foundations. Following this approach, some existing general knowledge (even if limited) is added to constrain the output of the model to "reasonable" results. A canonical formalism is a general mathematical representation adaptable to many specific systems – via model calibration – that provides a sound basis to build semi-empirical models based on a general theoretical framework. In a seminal work, Savageau introduced one such formalism, the power-law representation (Savageau, 1969a, 1969b, 1970), which was later extended by Sorribas et al. (2007) and applied to model biochemical networks. Despite their simplicity and versatility, canonical formalisms, particularly the S-system approach, have often been used to model biochemical systems at the molecular level, while their application to model bioprocesses at a larger scale is yet to be explored.

This work introduces a method based on the S-system canonical formalism to build dynamic models of bioprocesses, determining both the model structure and its parameters and focusing on problems where only scarce data are available. In essence, our approach combines a first-principles backbone based on mass balances with a canonical kinetic S-system formalism, whose structure and model parameters are automatically identified by solving an MINLP. This approach significantly tightens the search space by applying "rational" constraints to the

structure of the model. The model training is performed following a two-stage approach that simplifies the calculations by avoiding the iterative integration of differential equations. Numerical examples show that our method performs similarly to models based on ANNs, while leading to a trained model based on a compact, analytical canonical expression. The latter can be used for monitoring, control, and optimization, among others.

The remainder of the paper is organized as follows: First, the problem statement is detailed, followed by the methodology. Afterward, the case studies are introduced, and the results are discussed. Finally, the conclusions of the work are drawn.

2.2 Problem statement

Here we shall consider a typical reactor implementing a given reaction system, as depicted in Figure 2.1.

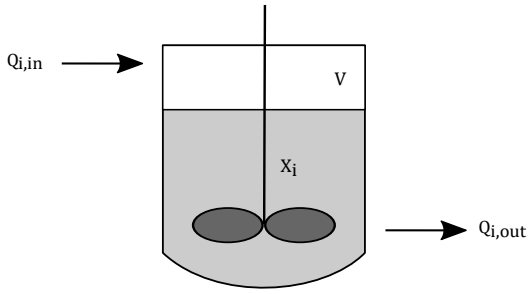

Figure 2.1. Schematic representation of a generic reactor with input/output streams $Q_{i,in}/Q_{i,out}$, considering a system volume V, and a concentration X_i of species i.

For a generic, ideally mixed, homogeneous, and isothermally operated reactor with constant volume, the overall mass balance for species $i \in I$ is shown in equation (2.1):

$$\frac{dX_i}{dt} = \dot{X} = \frac{1}{V}(Q_{i,in} - Q_{i,out}) + Rxn_i(X), \forall i \in I \tag{2.1}$$

where $Q_{i,in}$, $Q_{i,out}$ and $Rxn_i(X)$ denote the mass flows in and out of the reactor and the generation/depletion term, respectively, and $\frac{dX_i}{dt}$ (or $\dot{X}$) refers to the accumulation term. For a batch process, where no inlet nor outlet are considered,

the mass balance reduces to the accumulation and generation terms:

$$\frac{dX_i}{dt} = \dot{X} = Rxn_i(X), \forall i \in I \tag{2.2}$$

Where the $Rxn_i(X)$ term is an unknown expression that depends on the state variables X (e.g., species concentration). In this work, we approximate $Rxn_i(X)$ using ANNs and a canonical formalism tailored to the particular system.

Given a set of experimental observations X, the goal of the analysis is to find $Rxn_i(X)$ in equation (2.2) such that the mismatch between the model predictions $\hat{X}$ and the experimental observations is minimized. Note that, unlike the standard parameter estimation problem where the model structure is known, we assume that it is unknown but follows a given formalism. Therefore, we aim to find both the rate expression and its parameters simultaneously instead of solving a standard inverse problem with a defined structure, as discussed by Voit and Almeida (2004) and Brendel et al. (2006). The section that follows introduces our approach.

2.3 Methodology

We describe two alternative approaches to tackle the problem above, inspired by the modeling ideas in Guillén-Gosálbez et al. (2013), the Ph.D. thesis of Miró (2014), the work of Sorribas et al. (2010), and the decomposition methods in Voit and Almeida (2004), Michalik et al. (2009), and Brendel et al. (2006). Finally, we study the identified model performance and compare it to a benchmark model.

2.3.1 Modeling framework

In essence, our modeling approach relies on a backbone based on mass balances coupled with a canonical component that predicts the reaction rate (generation/depletion term in the mass balance) from the state variables, whose values vary over time. It is worth highlighting that this approach can provide a general modeling structure, where the canonical part can be replaced by any kind of formalism (i.e., an ANN, a GP, a Generalized Mass Action (GMA) expression, etc.). Here we focus on a regular widespread ANN and the S-system representation. Other surrogates, e.g., GPs or deep learning methods, could also be used

but are out of the scope of this work. Our method is based on an incremental approach for model building, as shown in Figure 2.2, where the ODE integration is avoided by fitting the slopes determined experimentally, as explained later in the article.

First, the slopes are calculated for each profile. The polynomial approach described in the manuscript Section 2.4.3 was applied to fit these slopes. However, another approach might also be suitable. Second, a training dataset is set up, where the input data consists of the experimental states, and the target data consists of the calculated slopes. With this data at hand, both the ANN and the S-system can be trained. The exact equations used for this training are described below. Third, the trained model is incorporated into the ODE, which can then be solved using specific initial conditions.

Backbone based on mass balances

The backbone of the model describes the mass balance equations for a generic chemical reaction in a batch reactor, as given by equation (2.2). The generation/depletion term $Rxn_i(X)$ is herein modelled following two approaches: ANN and S-system.

Data-driven component based on neural network approximation

In the first approach, the $Rxn_i(X)$ term is approximated using a shallow feed-forward ANN as follows:

$$Rxn_i(X) \approx ANN_i(X), \quad \forall i \in I \tag{2.3}$$

Where $ANN_i(X)$ denotes the neural network that uses some independent variables (i.e., the state variable/concentration of the species X) as input features. The limitation of this approach, as already mentioned, is that it is entirely data-driven and, thus, can lead to poor estimates, particularly when attempting extrapolation outside of the training range. Furthermore, an additional limitation is that the ANN is hard to interpret given its black-box nature. Hence, the following section describes how, alternatively, a tailored canonical kinetic formalism can be used to improve interpretability.

Canonical component based on an S-system

In the second approach, we rely on the biochemical systems theory (BST) to find a suitable rate expression. For a very detailed review of BST, the reader is

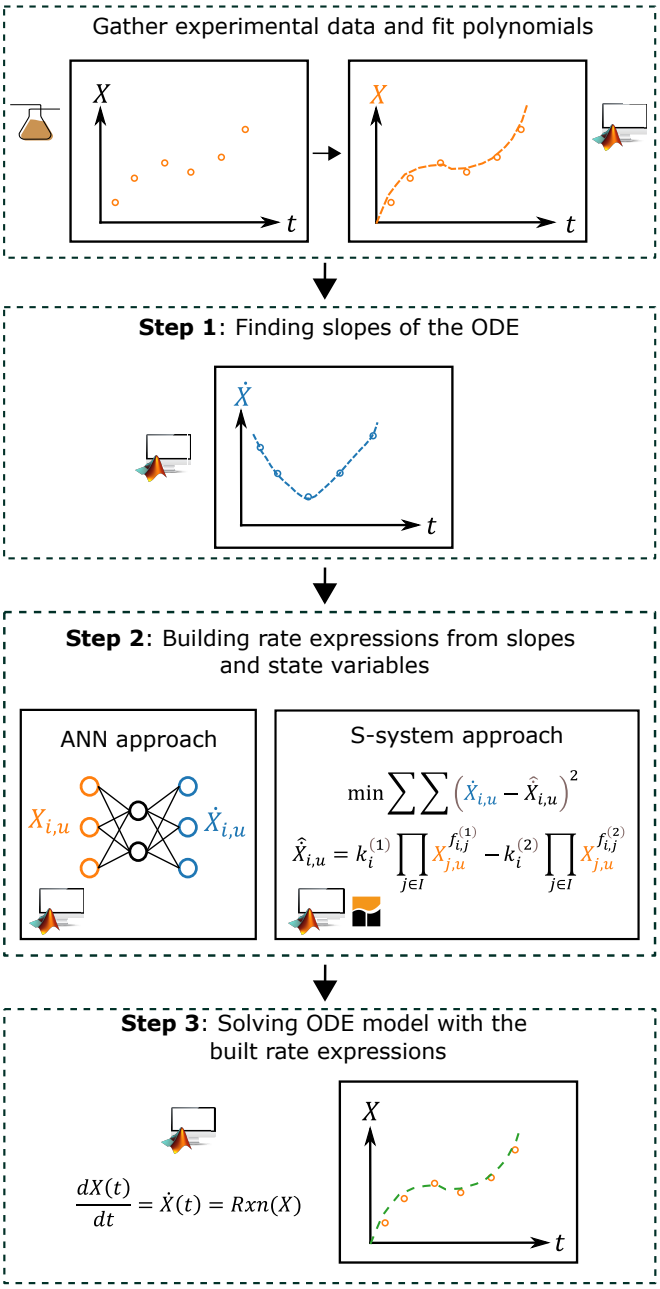

Figure 2.2. Overview of the incremental approach for model building. After gathering experimental data, a polynomial is fit from data points (one polynomial for each species and experimental run). Slopes are obtained from this polynomial via differentiation and then evaluated at the experimental time points (step 1). To train the models (ANN or S-system), the collected states in red and the calculated slopes in green are used (step 2). After training the models to predict the slopes from input states, they are incorporated into an ODE which can be solved with appropriate initial conditions (step 3).

referred to the work by Savageau (1969a, 1969b, 1970) and Voit (2013). BST uses canonical models, which means that the model-building procedure follows well-defined rules. Such models are built, for example, by applying approximation theory (Voit, 2013). One such representation is the power-law approximation, first presented by (1969a, 1969b, 1970). Following this representation, each of the participating reactions involved in the generation term $Rxn_i(X)$ in equation (2.2) are described with an individual reaction rate ν_r. Furthermore, each ν_r depends on the metabolite concentrations X_i of species i. The formalism is derived using an approximation of a general kinetic function of a reaction r in logarithmic coordinates, leading to the following expression:

$$\nu_r(X_1, X_2, \ldots, X_{|I|}) \approx k_r \prod_{j \in I} X_j^{f_{j,r}}, \quad \forall r \in R \tag{2.4}$$

Where k_r represents the rate constant of reaction $r \in R$, and $f_{j,r}$ represents the reaction order of species j influencing species i in reaction r. By summing over the $|R|$ reactions and considering the stoichiometry $\gamma_{i,r}$ of species i in reaction r, we can state the GMA model as:

$$\dot{X} = \sum_{r \in R} \gamma_{i,r} \nu_r = \sum_{r \in R} \left(\gamma_{i,r} k_r \prod_{j \in I} X_j^{f_{j,r}} \right), \quad \forall i \in I \tag{2.5}$$

By considering only two lumped terms, one for the species generation and one for the species depletion, we obtain the so-called S-system representation, which was first described in the mid-1980s (Savageau, 1985; Voit and Savageau, 1985):

$$\dot{X}_i = \alpha_i \prod_{j \in I} X_j^{g_{i,j}} - \beta_i \prod_{j \in I} X_j^{h_{i,j}}, \quad \forall i \in I \tag{2.6}$$

According to equation (2.6), the maximum possible number of parameters in the S-system only depends on the number of species in the system. Hence, it is defined *a priori* for the system under study. Where α_i and β_i represent the production and depletion rate constants for species i. The exponents $g_{i,j}$ and $h_{i,j}$ represent the reaction order of species j influencing species i for the production and depletion reactions, respectively. Following the GMA or S-system formalism, a given variable X_j will have a positive/enhancing effect on X_i if the exponent $f_{i,j}$ is positive. Conversely, it will have a negative/inhibiting effect on X_i if the exponent is negative. In case the exponent equals zero, variable X_j will not influence X_i.

This approach captures the evolution of the concentration of the metabolites and how the metabolites interact among themselves. Hence, the generation term in equation (2.2) is approximated using the aforementioned S-system, such that the relationship shown in equation (2.6) can be specified as follows:

$$Rxn_i(X) = \alpha_i \prod_{j \in I} X_j^{g_{i,j}} - \beta_i \prod_{j \in I} X_j^{h_{i,j}}, \quad \forall i \in I \tag{2.7}$$

Compared to the ANN, the S-system approach presents the advantage of providing interpretable models based on a canonical formalism, where all the coefficients have some physical meaning. Note that including many potential interactions between metabolites may lead to overfitted models with poor extrapolation capabilities. Hence, the challenge is finding a regulatory scheme capturing the main interactions and providing good estimates, even during extrapolation. As discussed in the next section, we pose this task as an MINLP problem, which is solved iteratively to provide candidate models with alternative regulatory interactions, i.e., with alternative values of the S-system parameters.

2.3.2 Incremental approach for model building

It is worth mentioning that until now, the state variables X_i were considered to be continuous in time (X did not depend on set U). For the subsequently proposed incremental approach, the concentration data for the different species $i \in I$ are collected by sampling from an ongoing reaction for each time point $u \in U$ resulting in an $|U|$-dimensional array of samples for each species i for the different time points $X_{i,u}$:

$$X_{i,u} = \left[X_{i,0}, X_{i,1}, X_{i,2}, \ldots, X_{i,|U|}\right], \quad \forall i \in I$$

We then seek the rate expression that is able to predict the time-dependent evolution by just using the state variables at a given initial time t_0. This rate expression will be subsequently used to construct the overall model.
The model-building task for the S-system can first be posed as a general dynamic optimization problem, where the sum of squared residuals (SSR) is minimized. The problem can therefore be formulated as follows:

$$
\begin{aligned}
\min_{\beta} SSR = & \sum_{i \in I} \sum_{u \in U} \left(X_{i,u} - \hat{X}_{i,u} \right)^2 \\
\text{s.t.} & \dot{\hat{X}}_{i,u} = Rxn_i(X_{j,u}, \beta), \quad \forall i \in I, j \in I, u \in U \\
& \hat{X}_{i,0} = X_{i,0}, \quad \forall i \in I \\
& \hat{X} \in \mathbb{R}^+, \dot{\hat{X}} \in \mathbb{R}^+
\end{aligned}
\tag{2.8}
$$

Where $\dot{\hat{X}}_{i,u}$ represents the predicted slope variable of species i at time point t_u and $\hat{X}_{i,0}$ the initial conditions. The intrinsic model parameters β (weights/biases of an ANN or reaction rate constants/exponents in the S-system) of the untrained expression $Rxn_i(X, \beta)$ are represented as trainable decision variables. Solving the dynamic problem requires integrating the ODEs to obtain the objective function value, which can be accomplished by following a sequential or simultaneous approach. To simplify the calculations, we follow the approach proposed by Michalik et al. (2009) and Voit and Almeida (2004) to solve the dynamic model above, which formulates the problem in the slopes space instead of the state variables space. In essence, the idea here is to obtain the slopes from the dynamic profiles of the state variables and then use them to calibrate the kinetic model. This strategy avoids integrating the dynamic system, as the slopes are experimentally determined rather than predicted *in-silico* from the kinetic equations. The choice of a specific rate expression $Rxn_i(X, \beta)$ dictates the model type, leading to the following two alternative problems. The first includes the ANN as given in equation (2.3):

$$
\begin{aligned}
\min_{\omega, b} SSR = & \sum_{i \in I} \sum_{u \in U} \left(\dot{X}_{i,u} - \dot{\hat{X}}_{i,u} \right)^2 \\
\text{s.t.} & \left[\dot{\hat{X}}_i, \ldots, \dot{\hat{X}}_{|I|} \right] = ANN \left(\left[X_i, \ldots, X_{|I|} \right]_u, \omega, b \right) \\
& \hat{X} \in \mathbb{R}^+, \dot{\hat{X}} \in \mathbb{R}^+
\end{aligned}
\tag{2.9}
$$

The second problem formulation includes the S-system, as given in equation (2.7):

$$
\begin{aligned}
\min_{\alpha,\beta,g,h} SSR = & \sum_{i\in I}\sum_{u\in U}\left(\dot{X}_{i,u} - \hat{\dot{X}}_{i,u}\right)^2 \\
\text{s.t.} \hat{\dot{X}}_{i,u} = & \alpha_i \prod_{j\in I} X_{j,u}^{g_{i,j}} - \beta_i \prod_{j\in I} X_{j,u}^{h_{i,j}}, \quad \forall i \in I, u \in U \\
& \underline{\alpha_i} \leq \alpha_i \leq \overline{\alpha_i}, \quad \forall i \in I \\
& \underline{\beta_i} \leq \beta_i \leq \overline{\beta_i}, \quad \forall i \in I \\
& \underline{g_{i,j}} \leq g_{i,j} \leq \overline{g_{i,j}}, \quad \forall i,j \in I \\
& \underline{h_{i,j}} \leq h_{i,j} \leq \overline{h_{i,j}}, \quad \forall i,j \in I \\
& \alpha \in \mathbb{R}^+, \beta \in \mathbb{R}^+, g \in \mathbb{R}, h \in \mathbb{R} \\
& \hat{X} \in \mathbb{R}^+, \hat{\dot{X}} \in \mathbb{R}^+
\end{aligned}
\tag{2.10}
$$

Parameters α_i, β_i and $g_{i,j}$, $h_{i,j}$ of the S-system are decision variables in the NLP. Upper and lower bounds on the parameters α_i,β_i and $g_{i,j}$, $h_{i,j}$ ($\underline{\alpha_i}$, $\underline{\beta_i}$, $\overline{\alpha_i}$, and $\overline{\beta_i}$, $\underline{g_{i,j}}$, $\underline{h_{i,j}}$, and $\overline{g_{i,j}}$, $\overline{h_{i,j}}$) are imposed. In contrast to the GMA, in the S-system, the reaction rate constants α_i and β_i are defined to be positive. The exponents $g_{i,j}$, $h_{i,j}$ are considered to be real numbers between defined upper and lower bounds.

A critical issue in equation (2.10) concerns the selection of the regulatory scheme, defined by the set of non-zero parameters α_i, β_i and $g_{i,j}$, $h_{i,j}$. Here, we formulate the task of finding the most suitable regulatory structure as an MINLP, given by (2.11).

$$
\begin{aligned}
\min_{\substack{\alpha,\beta,g,h,\\ y\alpha,y\beta,yg,yh}} \quad & SSR = \sum_{i\in I}\sum_{u\in U}\left(\dot{X}_{i,u} - \hat{\dot{X}}_{i,u}\right)^2 \\
& \text{s.t.} \hat{\dot{X}}_{i,u} = \alpha_i \prod_{j\in I} X_{j,u}^{g_{i,j}} - \beta_i \prod_{j\in I} X_{j,u}^{h_{i,j}}, \quad \forall i \in I, u \in U \\
& \sum_{i\in I}\left(y\alpha_i + y\beta_i + \sum_{j\in I}(yg_{i,j} + yh_{i,j})\right) \leq NP \\
& (1 - yg_{i,j}) + y\alpha_i \geq 1, \quad \forall i,j \in I \\
& (1 - yh_{i,j}) + y\beta_i \geq 1, \quad \forall i,j \in I \\
& y\alpha_i \cdot \underline{\alpha_i} \leq \alpha_i \leq y\alpha_i \cdot \overline{\alpha_i}, \quad \forall i \in I \\
& y\beta_i \cdot \underline{\beta_i} \leq \beta_i \leq y\beta_i \cdot \overline{\beta_i}, \quad \forall i \in I \\
& yg_{i,j} \cdot \underline{g_{i,j}} \leq g_{i,j} \leq yg_{i,j} \cdot \overline{g_{i,j}}, \quad \forall i,j \in I \\
& yh_{i,j} \cdot \underline{h_{i,j}} \leq h_{i,j} \leq yh_{i,j} \cdot \overline{h_{i,j}}, \quad \forall i,j \in I \\
& y\alpha, y\beta, yg, yh \in \{0,1\} \\
& \alpha \in \mathbb{R}^+, \beta \in \mathbb{R}^+, g \in \mathbb{R}, h \in \mathbb{R} \\
& \hat{X} \in \mathbb{R}^+, \hat{\dot{X}} \in \mathbb{R}^+
\end{aligned}
\tag{2.11}
$$

As in the NLP approach above, the parameters α_i, β_i and $yg_{i,j}$, $yh_{i,j}$ of the S-system are decision variables in the MINLP. Binary variables $y\alpha_i$, $y\beta_i$ and $yg_{i,j}$, $yh_{i,j}$ denote the existence of the model parameters (one if selected, and zero otherwise), all of which must lie within a given interval (i.e., $\underline{\alpha_i}$, $\underline{\beta_i}$, $\overline{\alpha_i}$, and $\overline{\beta_i}$, $\underline{g_{i,j}}$, $\underline{h_{i,j}}$, and $\overline{g_{i,j}}$, $\overline{h_{i,j}}$). The MINLP also constrains the maximum number of parameters (NP), which defines the model complexity. In model (2.11), binary variables $y\alpha$, $y\beta$, yg and yh model the selection of parameters α and β in the S-system. Accordingly, if metabolite i influences the reaction rate of species j, then the corresponding binary variable is one, and it will be zero otherwise.

Overall, our model-building approach comprises three steps, as discussed next for the two different cases (i.e., ANN and S-system). A schematic overview of the solution procedure is shown in Figure 2.3.

Step 1: Finding the slope variables of a profile

The solution method for models (2.9) and (2.11) is based on the incremental approach proposed by Michalik et al. (2009) and Voit and Almeida (2004). In

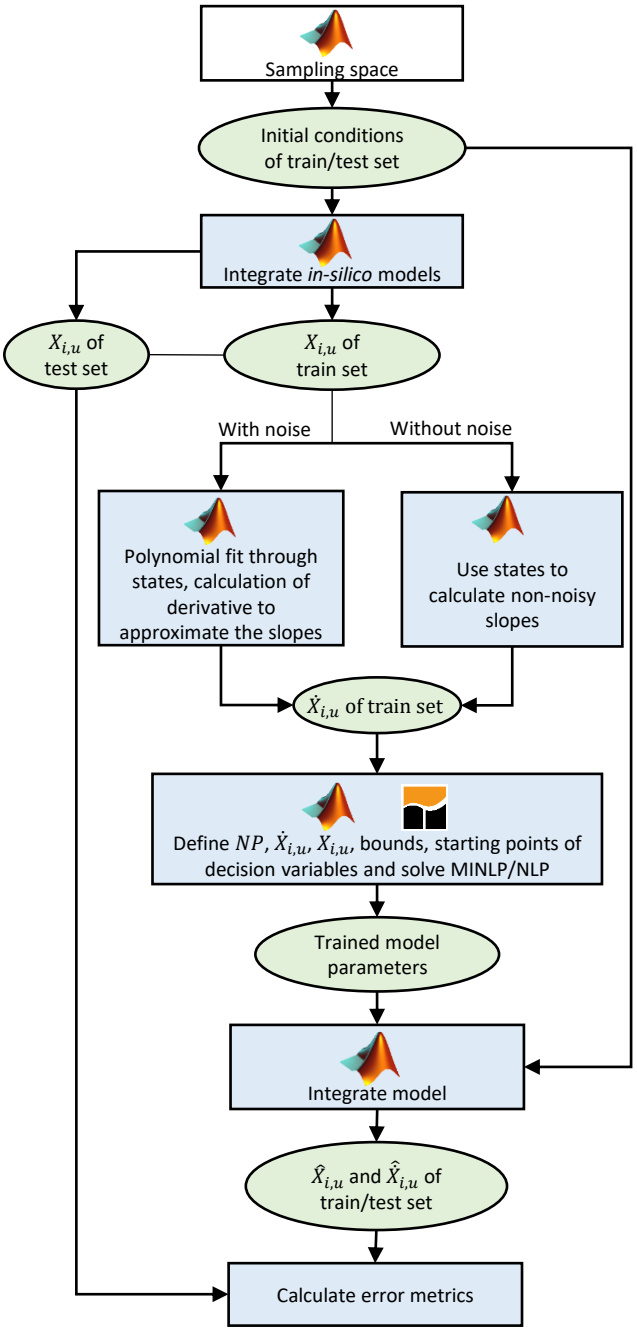

Figure 2.3. Flowchart of the proposed computational approach for the solution method based on an incremental approach. Green-colored ellipsoidal blocks represent the results of the task executed in the blue-colored squared blocks.

essence, we decouple the model training - finding the expression rate in equation (2.9) and (2.11) - from the ODE integration to simplify the solution of the

dynamic optimization problem. These slopes are determined as explained in Section 2.4.3.

Step 2: Building the rate expression from the slopes and state variables
The ANN is trained using the states of all available species $X_{i,u}$ as inputs and the corresponding slopes $\dot{X}_{i,u}$ as target values (outputs). The ANN, therefore, predicts the changes in concentration of all the species at a given time t_u.

Applying the S-system approach, once the slopes are obtained, the MINLP in equation (2.11) is formulated and solved, finding the optimal values of α_i, β_i and $yg_{i,j}$, $yh_{i,j}$, respectively. The MINLP can be solved for different bounds on the number of parameters, providing in each run a candidate model that can be later on tested in the validation set, as discussed below. If all the binary variables are fixed to a value of one, the MINLP becomes the NLP given by (2.10).

Step 3: Solving the ODE model with the built rate expressions
The trained ANN can be incorporated into the model backbone given by equation (2.2), resulting in equation (2.12):

$$\frac{dX_i}{dt} = ANN_i(X_i), \quad \forall i \in I \tag{2.12}$$

Likewise, the trained S-system shown in equation (2.7) can be incorporated into the model backbone given by equation (2.2), resulting in equation (2.13).

$$\frac{dX_i}{dt} = \alpha_i \prod_{j \in I} X_j^{g_{i,j}} - \beta_i \prod_{j \in I} X_j^{g_{i,j}}, \quad \forall i \in I \tag{2.13}$$

Since in equation (2.12) the weights ω and biases b, and in equation (2.13) the parameters α_i/β_i are all known, these ODEs can be solved by using appropriate initial conditions $X_{i,0}$ for every species i and considering the desired integration period $t = [t_0, t_f]$. It is worth to be mentioned that since from that point onwards we have a trained model available, we do not need the sampled/discrete concentration data $X_{i,u}$ points anymore. Therefore, the concentration of species i is considered to be a continuous function of the time, which is further described as $X_i = X_i(t)$.

2.3.3 Model performance

For assessing the performance of the models, an arbitrary set of initial conditions can be used to integrate the ODE, where a simulated concentration profile is compared to the experimental values. After training the models on dedicated training runs, test runs were used to assess the model performance. A detailed description of how the data splitting was achieved is shown in Section 2.4.2.

The performance is assessed via the root mean squared error ($RMSE$) and the coefficient of determination (R^2), defined as in equations (2.14). These metrics can be calculated for both the training and test sets, obtaining the training and test errors, respectively.

$$RMSE_{state} = \sqrt{\frac{1}{n}\sum_{i\in I}\sum_{u\in U}\left(\hat{X}_{i,u} - X_{i,u}\right)^2} \tag{2.14a}$$

$$RMSE_{slope} = \sqrt{\frac{1}{n}\sum_{i\in I}\sum_{u\in U}\left(\hat{\dot{X}}_{i,u} - \dot{X}_{i,u}\right)^2} \tag{2.14b}$$

$$R^2_{state} = 1 - \frac{SSR}{SST} = 1 - \frac{\sum_{i\in I}\sum_{u\in U}\left(\hat{X}_{i,u} - X_{i,u}\right)^2}{\sum_{i\in I}\sum_{u\in U}\left(X_{i,u} - \overline{X}_{i,u}\right)^2} \tag{2.14c}$$

$$R^2_{slope} = 1 - \frac{SSR}{SST} = 1 - \frac{\sum_{i\in I}\sum_{u\in U}\left(\hat{\dot{X}}_{i,u} - \dot{X}_{i,u}\right)^2}{\sum_{i\in I}\sum_{u\in U}\left(\dot{X}_{i,u} - \overline{\dot{X}}_{i,u}\right)^2} \tag{2.14d}$$

In these relationships, the predictions by the model are described by $\hat{X}_{i,u}$ (or $\hat{\dot{X}}_{i,u}$). The experimental data points and the mean of the experimental data points of the test set are described by $X_{i,u}$ and $\overline{X}_{i,u}$ (or $\dot{X}_{i,u}$ and $\overline{\dot{X}}_{i,u}$), respectively. The model predictions $\hat{X}_{i,u}$ are calculated by using input data from the training or the test set. In this work, the terms test set and test run are used interchangeably. Variables SSR and SST denote the sum of squares of residuals and the total sum of squares (proportional to the variance of the data), respectively.

2.4 Case studies

2.4.1 Software implementation

All simulations were carried out on an AMD Ryzen-5 3600 CPU and 16 GB of RAM. We used Matlab 2020a (The MathWorks Inc, 2020) to construct the modeling environment, train the ANN, and plot the results. We used the General Algebraic Modeling System (GAMS) (GAMS Development Corporation, 2020), version 35.1.0, and SBB (Bussieck & Drud, 2001), version 32.2.0, to implement and solve the MINLP for training the S-system, respectively.

2.4.2 Underlying ODE models for *in-silico* data generation

Several kinetic models were used in this study to test our approach. As described in the subsequent sections, the different case studies (CS) are of increasing complexity, starting with a simple Monod reaction and finally moving to a more complex bioprocess in batch operation mode.

In the following, the case studies and their corresponding data generation processes are described. The methods discussed above are applied to predict the concentration profiles in the reactors. The corresponding model parameters required for each case study are tabulated in the supplementary material Section A.1 Table A.1 to Table A.4. The initial conditions were generated by applying a Latin Hypercube Sampling (LHS) design (data available in Table 2.1).

For the *in-silico* data generation, the underlying ODEs were solved numerically in Matlab by using the built-in function `ode15s`. In order to simulate an operator's measurement error (same for all the species), Gaussian white noise with a mean $\mu = 0$ and variance of $\sigma^2 = 0.075$ was introduced. This leads to noisy training and testing datasets. To this end, the Matlab built-in function `normrnd` was used. Below, for each model, the applied initial conditions and the necessary parameters are indicated. A Latin hypercube sampling (LHS) design method was used to draw six training and six independent testing initial conditions. Additionally, the time span for integration is stated below for each model individually. In the five case studies, the same following settings were applied:

- The different *in-silico* runs are generated by changing only the initial conditions.

- For the generation of the test set, some initial conditions (and, therefore, also some states) exceed the trained feature space (i.e., extrapolation).

Case study I Consider a first case study as an example that is fully expressable by the S-system formalism. It was inspired by and adjusted from Esposito et al. (2000), shown in equation (2.15). The upper-case letters represent the concentrations of $X = \{A, B, C, D\}$, where some constant reaction parameters are given by k_r for $r = \{1, 2, 3\}$.

$$\begin{aligned} \frac{dA}{dt} &= -k_1 A + k_2 C \\ \frac{dB}{dt} &= -k_1 B + k_2 C \\ \frac{dC}{dt} &= k_1 A - k_2 C - k_3 C \\ \frac{dD}{dt} &= k_3 C \end{aligned} \tag{2.15}$$

Case study II The second case study is a batch reaction, shown in equation (2.16), representing the isothermal Van-de-Vusse reaction (Floudas et al., 1999).

$$\begin{aligned} \mathrm{A} &\xrightarrow{\mathrm{k_1}} \mathrm{B} \xrightarrow{\mathrm{k_2}} \mathrm{C} \\ 2\,\mathrm{A} &\xrightarrow{\mathrm{k_3}} \mathrm{D} \end{aligned}$$

$$\begin{aligned} \frac{dA}{dt} &= -k_1 A + 2k_3 A^2 \\ \frac{dB}{dt} &= k_1 A - k_2 B \\ \frac{dC}{dt} &= k_2 B \\ \frac{dD}{dt} &= k_3 A^2 \end{aligned} \tag{2.16}$$

Case study III As the third dataset, the following scheme given in equation (2.17) is used, which represents the α-Pinene reaction (Floudas et al., 1999).

$$\begin{aligned} \mathrm{A} &\xrightarrow{\mathrm{k_1}} \mathrm{E} \\ \mathrm{A} &\xrightarrow{\mathrm{k_2}} \mathrm{D} \end{aligned}$$

$$
\begin{aligned}
&\mathrm{A} \xrightarrow{\mathrm{k_5}} \mathrm{C} \\
&\mathrm{C} \underset{\mathrm{k_7}}{\overset{\mathrm{k_4}}{\rightleftharpoons}} \mathrm{D} \\
&\mathrm{D} \underset{\mathrm{k_6}}{\overset{\mathrm{k_3}}{\rightleftharpoons}} \mathrm{B}
\end{aligned}
$$

$$
\begin{aligned}
\frac{dA}{dt} &= -(k_1 + k_2 + k_5)A \\
\frac{dB}{dt} &= k_3 D - k_6 B \\
\frac{dC}{dt} &= k_4 C + k_5 A - k_7 D \\
\frac{dD}{dt} &= k_2 A + k_6 B + k_7 C - (k_3 + k_4) D \\
\frac{dE}{dt} &= k_1 A
\end{aligned}
\tag{2.17}
$$

Case study IV In addition to the above-shown case studies, used to model chemical reactions, the following case study represents a target protein production by bacteria. B, S, and P represent the biomass, the substrate concentration, and the product concentration, respectively, where these species are summarized in the set $X = \{B, S, P\}$. The process is modeled in batch mode and adapted from Turton et al. (2018).

$$
\begin{aligned}
\frac{dB}{dt} &= \phi B \\
\frac{dB}{dt} &= \Sigma B \\
\frac{dC}{dt} &= \pi B
\end{aligned}
\tag{2.18}
$$

The equations given in equation (2.18) are used to calculate the cell growth ϕ, the substrate consumption rate Σ, and product formation rate π.

$$
\begin{aligned}
\phi &= \phi_{max} \frac{S}{K_S + S} \frac{\Omega_1(A_1, T, E_1)}{1 + \Omega_2(A_2, T, E_2)} \left(1 - \frac{B}{K_\phi + B}\right) \\
\Sigma &= -\frac{1}{Y_{B,S}} \phi \\
\pi &= \frac{Y_{P,S}}{Y_{B,S}} \phi
\end{aligned}
\tag{2.19}
$$

$Y_{i,j}$ represents the yield coefficient of species j with respect to species i, Ω_1, Ω_2 represent the reaction rate constants depending on the temperature T, the temperature-independent pre-factors A_1, A_2, and the activation energies E_1, E_2, and K_S, K_ϕ represent the inhibition constants.

Table 2.1. Data used for generating the training and test sets. The lower (X_0^{lo}) and upper (X_0^{up}) bounds for the initial concentrations are indicated along with the initial (t_i) and final (t_f) times for the integration.

CS	Training				Testing			
	X_0^{lo}	X_0^{up}	t_i	t_f	X_0^{lo}	X_0^{up}	t_i	t_f
I	[0.8,0.5,0,0]	[3,2,0,0]	0	2	[0.2,0.1,0,0]	[4.5,4,0,0]	0	2
II	[5 5,0,0]	[14,10,0,0]	0	1.2	[2,2,0,0]	[16,12,0,0]	0	1.2
III	[1.5,1.5,0,0,0]	[14,11.5,0,0,1.2]	0	3	[0,0,0,0,0]	[18,15,0,0,2]	0	3
IV	[0.1,50,0]	[0.4,90,0]	0	80	[0.1,30,0]	[1,110,0]	0	80

2.4.3 Calculation of the slope variables

We apply our method to non-noisy and noisy datasets. The first step is to compute the slopes used to build the regression model following two different approaches, as explained next.

Non-noisy dataset After having integrated the ODE system, the resulting concentration profiles were considered as states $X_{i,u}$. The non-noisy slopes $\dot{X}_{i,u}$ are calculated by inserting the state variables measured at given points in time into the original *in-silico* model used to generate all the simulated experimental values.

Noisy dataset *In-silico* models/expressions are seldom available, so the slopes need to be approximated from the dynamic profiles as explained next. Notably, for CSIV, a preprocessing step based on the Savitzky-Golay (SG) filter is applied to the state values $X_{i,u}$ of the training runs, followed by a scaling procedure considering the maximum training value. A detailed description of this scaling is given in the supplementary material Section A.2.

To obtain the dynamic profile, a model of choice is fitted through the experimental data points. The calculated time profile can subsequently be derived, leading to a derivatives profile over time. For this purpose, alternative models such as exponential functions, regression splines, or simple polynomials could be chosen. Without loss of generality, in this work, we used the polynomial shown in expres-

sion (2.20). Therefore, a polynomial $\zeta_{i,u}$ of defined order q is fitted in time t to the state data points $X_{i,u}$ of each different species i and for each training run as shown in equation (2.20). For CSI/II/IV order $q = 5$ and for CSIII order $q = 4$ was chosen. The order q can be regarded as a hyperparameter, which could be tuned to reach better derivative values. There would be many ways to determine the value of the polynomial order q. One possibility could be to assess a desired fitting error metric (i.e., the $RMSE$ or MAE) for different values of q, choosing the q value leading to the lowest fitting error. Cross-validation could also be used to determine appropriate values of the polynomial order.

$$X_{i,u} \approx \zeta_{i,u} = p_{i,1} + p_{i,2}t_u + \cdots + p_{i,q+1}t_u^q \tag{2.20}$$

Where $p_{i,q}$ are the unknown polynomial coefficients, t_u is the time at the measurement point u, and $\zeta_{i,u}$ represents the time profile at time points u of the approximated state variable of species i. A multi-linear regression approach provides the polynomial coefficients p in equation (2.20). By analytically differentiating the polynomial, the coefficients of the differentiated polynomial $\dot{\zeta}_{i,u}$ are found as given in equation (2.21).

$$\dot{\zeta}_{i,u} \approx \dot{X}_{i,u} \tag{2.21}$$

This procedure was followed for all the dynamic profiles generated for different experiments. Equation (2.21), therefore, provides approximated slopes $\dot{X}_{i,u}$ at any required time point.

We shall next use the slopes information computed above to build the rate expression embedded in the dynamic model. It is worth to be mentioned that the polynomial is not further used for any step after having obtained the derivative variables at the given time point. This approach was used to calculate the derivatives instead of, for example, using forward finite derivatives. After obtaining the derivatives, the ANN and the S-system are trained to predict a derivatives by considering the state variables as input.

2.4.4 ANN architecture

The output layer includes as many neurons as species available in the system. The ANN architecture uses a two-layer network (one hidden layer, one output layer), where the hidden layer includes only three neurons. The hidden layer uses a tangential sigmoidal and the output layer a linear activation function. A Bayesian regularization backpropagation approach (Foresee & Hagan, 1997; MacKay, 1992; Rosa et al., 2020) is chosen as the training algorithm, setting an upper bound of 100 training epochs. For this task, the deep learning toolbox of Matlab was used.

2.5 Results

For each case study, we compared two approaches: In the first approach, we use the ANN as the surrogate model to predict the slopes. In the second approach, we first use the S-system considering all possible regulatory interactions between species to predict the slopes, resulting in an NLP model. We then use the S-system, considering different levels of complexity by varying the maximum number of parameters according to the model shown in equation (2.11). This leads to a series of MINLP models differing in the NP values, generating models of different sizes. The stopping criteria for the MINLP problem are the settings of the solver in GAMS (i.e., relative optimality gap, number of nodes, etc.), which are given in detail in the supplementary material Section A.4.

It is worth mentioning that the applied model (S-system) can only find the ground-truth model when it can be represented by an S-system formulation. However, even if this is not possible (i.e., CSII-IV), the proposed approach still results in an expression that can precisely explain and predict the system's behavior under study. Interpretability is enhanced because each parameter in the S-system model has a physical meaning related to how species interact with each other via regulatory loops. Additionally, one could replace the S-system with the GMA given in equation (2.5) or another model of choice.

To define appropriate starting values for the parameters to be estimated with the MINLP and NLP method, we applied the heuristic approach described in the supplementary material Section A.3. The same starting values are then used to initialize the NLP and the first MINLP. The chosen starting values are displayed

in Tables A.5 and A.6. Each subsequent MINLP uses the optimal solution of the previous MINLP to initialize the solver, where a new upper bound NP is set, where NP is increased for every new MINLP candidate. A visualization of this approach is given in Figure 2.4.

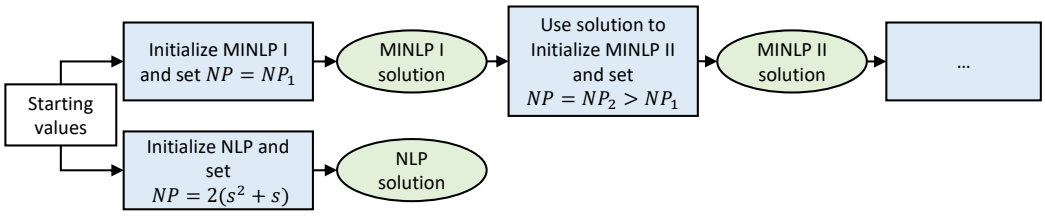

Figure 2.4. Visual representation of the model initialization for both, the NLP and MINLP models.

As shown in this figure, the first candidate MINLP is solved with a maximum number of non-zero parameters of $NP = NP_1$. After increasing the value for NP, leading to $NP = NP_2 > NP_1$, the following model candidate is solved. The largest possible value for NP is defined by the largest possible number of parameters in the S-system given in equation (2.6), which corresponds to $2(s^2+s)$, where s is the number of species present in the system.

In what follows, we provide the errors in the test set for both approaches (ANN and MINLP/NLP) for every case study. Note that models with lower training errors in the slope-space might not necessarily perform better in the state-space, as discussed in detail next. Table 2.2 and Table 2.3 summarize the obtained results, while Tables A.8 to A.15 show additional calculated error metrics (R^2 and $RMSE$) for the non-noisy slope/state spaces and the noisy slope/state spaces. In addition, Table A.16 and Table A.17 show the number of equations and variables that describe the MINLP model that was solved and the required CPU time for the solver.

As seen in Table 2.2 and Table 2.3, both methodologies provide similar performance for most cases, except for the third case study without noise. If we focus on the slope-space training error, the MINLP approach outperforms the ANN in three out of eight cases in both the noisy and non-noisy datasets. However, if we shift the focus to the state-space test $RMSE$ values, the MINLP approach outperforms the ANN in five out of eight cases. Note that a lower error in the training set and a higher error in the test set do not necessarily indicate overfitting here be-

Table 2.2. Error metrics for the different approaches are shown for the four case studies based on non-noisy data. For each case study, the best-performing approach in terms of state-space error is indicated in bold text. The indicated slope space training error is not necessarily the best, but rather the one corresponding to the best-performing model in the state space.

CS	Method	$RMSE_{state}$ test[b]	$RMSE_{slope}$ test[a]	Number of parameters	
				Model[c]	Underlying *in-silico*[d]
I	ANN	$2.23 \cdot 10^{-2}$	$8.14 \cdot 10^{-1}$	55	10 (14)
	MINLP[e]	$3.27 \cdot 10^{-4}$	$8.66 \cdot 10^{-12}$	14 (of NP=17)	
II	ANN	$6.97 \cdot 10^{-1}$	$3.91 \cdot 10^{-1}$	55	9
	MINLP[e]	$2.58 \cdot 10^{-1}$	$4.93 \cdot 10^{-1}$	11 (of NP=11)	
III	**ANN**	$8.84 \cdot 10^{-4}$	$4.78 \cdot 10^{-4}$	83	18
	MINLP[e]	$1.77 \cdot 10^{-1}$	$3.55 \cdot 10^{-2}$	21 (of NP=21)	
IV	ANN	$3.80 \cdot 10^{0}$	$7.02 \cdot 10^{-2}$	33	9
	MINLP[ef]	$0.37 \cdot 10^{0}$	$2.33 \cdot 10^{-2}$	24 (of NP=24)	

[a] Units: $\mathrm{mol\,L^{-1}\,h^{-1}}$ for CSI, II, III; $\mathrm{g\,L^{-1}\,h^{-1}}$ for CSIV.
[b] Units: $\mathrm{mol\,L^{-1}}$ for CSI, II, III; $\mathrm{g\,L^{-1}}$ for CSIV.
[c] Number of parameters in the model framework indicated by the column "method". For the ANN, the weights and biases are reported as the number of parameters. For the MINLP methods, the non-zero rate constants and exponents provide the number of parameters (in brackets, the corresponding upper bound, NP, is given).
[d] Number of parameters for chemical reactions: Number of rate constants (Tables A.1 to A.3) plus the number of non-zero reaction orders. Number of parameters for bioprocess: Number of constant parameters (Table A.4). The number given in parenthesis indicates the number of parameters that would be used to express this *in-silico* model in an S-system. If no number is indicated, this *in-silico* model cannot be fully described by an S-system.
[e] Only the best-performing model candidate of the MINLP approach in terms of state-space test error is listed.
[f] NLP represents the MINLP approach, where all binaries are set to one, leading to the NLP candidate.

cause we are moving from the slope-space to the state-space (through integration). On the other hand, the MINLP approach leads to models with fewer parameters, in addition to being more interpretable. An example of the model predictions is given in Figure 2.5, where the concentration profiles of the MINLP and the ANN approach are shown together with the observed data for CSI.

Figures A.1 and A.2 in the supplementary material Section A.7 provide further details on the MINLP iterations and the match with experimental concentrations (for the sake of simplicity, results are summarized in Table 2.2 and Table 2.3, where details are shown in the supplementary material Section A.7). In general, larger bounds on the maximum number of parameters allowed in the MINLP do not necessarily result in larger S-system models. A visualization of this observation is given in Figure 2.6 for CSII. Applying the methodology to non-noisy data (Figure 2.6 (a)) resulted in model candidates with maximum 13 parameters, although 40 parameters would have been acceptable. For the noisy data

(Figure 2.6 (b)), the same was observed only in the last iteration, where 35 parameters were chosen to be non-zero. Table A.8 to Table A.15 show that most model candidates do not reach the maximum allowed number of parameters. This is because the MINLP sometimes identifies optimal solutions with fewer parameters than the maximum allowable number, i.e., the constraint on the model size is met as a strict inequality rather than equality. Overfitting can be avoided by

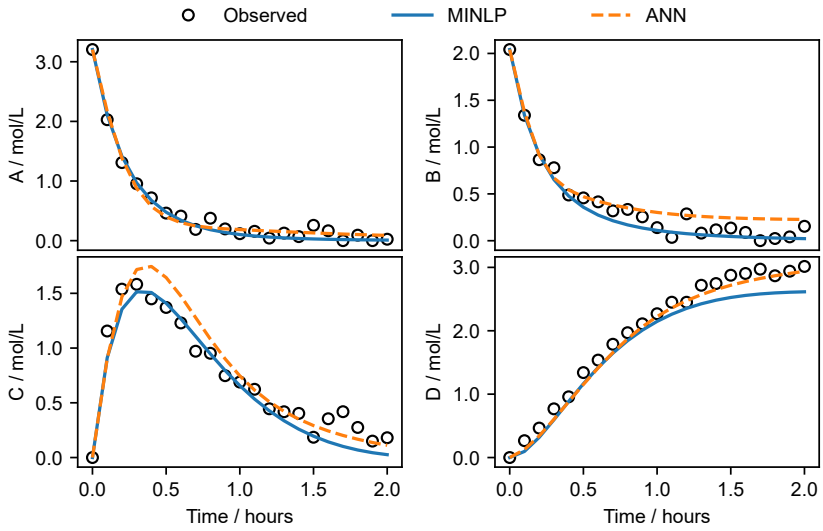

Figure 2.5. The concentration profiles of the four species in CSI are shown: The black circles represent the observed data, the dashed green line shows the ANN prediction, and the solid red line with dot markers displays the predictions by the S-system (at NP=12).

cross-validation, where the user runs several models for different values of NP and chooses the optimal value for NP such that a low test error is achieved. Moreover, sometimes the best MINLP in state-space test error is identified in the first iterations, e.g., in the noisy CSI-III and non-noisy CSI examples, and sometimes it emerges in the intermediate or final runs, e.g., in the non-noisy CSI, III, and IV examples or the noisy CSIV example. Concerning the fit to experimental data, we see how both approaches tend to predict well the concentrations profiles, except for some specific batches (for the ANN approach: non-noisy CSII batch 6, noisy CSI batch 6, noisy CSII batch 1, 3, and 6 and for the S-system approach: noisy CSI batch 2 and 4, noisy CSII batch 3, noisy CSIV batch 1) where the predictions deviate more from the original observations. It is worth to be mentioned that the above-indicated batches, where the ANN predictions deviate more from the original observations, include stagnating profiles. The ANN, therefore, seems to fail to predict such events more often than the S-system formalism does. Concerning the

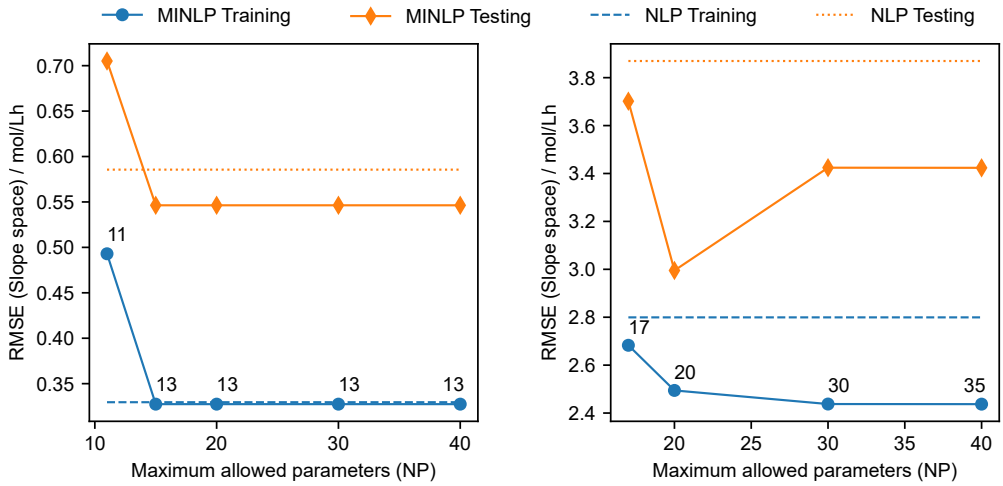

Figure 2.6. Comparison of the *RMSE* in the slope-space for CSII for (a) non-noisy data, and (b) noisy data. On the one hand, the training and test errors of the MINLP are shown as solid green and red lines with markers, respectively. The corresponding NLP training and test errors are shown as dashed-dotted lines (they are located on the horizontal axis). The numbers in the plot show the number of chosen non-zero parameters by the optimizer. The numbers in the plots indicate how many parameters the model chooses to be non-zero during the training procedure. As expected, the training error decreases by allowing more parameters to be non-zero. In (b) the testing error starts to increase again after a certain point, indicating overfitting during model training.

number of parameters, the ANNs have the same size in the two datasets (same architecture used for all calculations according to Section 2.4.4). In contrast, the MINLP approach tends to lead to fewer parameters when noise is added, i.e., in CSIV, where 24 parameters are used for the non-noisy dataset and 9 parameters are used for the noisy dataset. The detailed models are provided in Tables A.18 to A.21 in Section A.8 of the supplementary information. As seen, the S-system formulation results in compact models that could be used for further analyses and optimization.

2.6 Conclusion

Here we investigated how to build models for bioprocesses combining a first-principles backbone based on mass balances with a canonical kinetic S-system formalism. We developed an MINLP formulation to automatically identify the model structure and the values of its parameters, where binary variables denote the topology/structure of the model. The model training was performed following a two-stage approach, avoiding the iterative integration of differential equations.

Table 2.3. Error metric of the different approaches are shown for the four case studies based on noisy data. For each case study, the best-performing approach in terms of state-space error is indicated in bold text. The indicated slope space training error is not necessarily the best, but rather the one corresponding to the best-performing model in the state space.

CS	Method	$RMSE_{\text{state}}$ test[b]	$RMSE_{\text{slope}}$ test[a]	Number of parameters	
				Model[c]	Underlying *in-silico*[d]
I	ANN	$1.83{\cdot}10^{-1}$	$3.14{\cdot}10^{-1}$	55	10 (14)
	MINLP[e]	$1.51{\cdot}10^{-1}$	$3.99{\cdot}10^{-1}$	12 (of NP=12)	
II	ANN	$4.87{\cdot}10^{-1}$	$2.15{\cdot}10^{0}$	55	9
	MINLP[e]	$3.24{\cdot}10^{-1}$	$2.49{\cdot}10^{0}$	17 (of NP=20)	
III	**ANN**	$1.93{\cdot}10^{-1}$	$8.49{\cdot}10^{-2}$	83	18
	MINLP[e]	$2.55{\cdot}10^{-1}$	$1.24{\cdot}10^{-1}$	15 (of NP=15)	
IV	ANN	$9.46{\cdot}10^{-2}$	$4.42{\cdot}10^{-3}$	33	9
	MINLP[e]	$1.62{\cdot}10^{-1}$	$4.00{\cdot}10^{-3}$	9 (of NP=9)	

[a] Units: $\mathrm{mol\,L^{-1}\,h^{-1}}$ for CSI II, III; $\mathrm{g\,L^{-1}\,h^{-1}}$ for CSIV.
[b] Units: $\mathrm{mol\,L^{-1}}$ for CSI, II, III; $\mathrm{g\,L^{-1}}$ for CSIV.
[c] Number of parameters in the model framework indicated by the column "method". For the ANN, the weights and biases are reported as the number of parameters. For the MINLP methods, the non-zero rate constants and exponents provide the number of parameters (in brackets, the corresponding upper bound, NP, is given).
[d] Number of parameters for chemical reactions: Number of rate constants (Tables A.1 to A.3) plus the number of non-zero reaction orders. Number of parameters for bioprocess: Number of constant parameters (Table A.4). The number given in parenthesis indicates the number of parameters that would be used to express this *in-silico* model in an S-system. If no number is indicated, this *in-silico* model cannot be fully described by an S-system.
[e] Only the best-performing model candidate of the MINLP approach in terms of state-space test error is listed.

We applied our method to a range of case studies to showcase its capabilities. The presented approach performs similarly to models based on ANNs, even outperforming them in some cases. However, it has the additional advantage of leading to models based on a canonical form containing fewer parameters that are easier to interpret and use in optimization frameworks. Notably, we found that the ANN may fail to predict stagnating concentration profiles in some cases. Moreover, our approach is general enough to allow the data-driven part of the model to be exchanged by any kind of formalism (i.e., ANN, GP, etc.). Overall, having a model based on a canonical formalism would allow modelers to extract information about the processes and generate further insight into its behavior. In this context, the MINLP approach helps to adjust the complexity of the model considering overfitting. Hence, our approach could help find a suitable process model while simultaneously allowing practitioners to analyze the underlying formulation more easily and use it in optimization studies.

Chapter 3

Machine learning uncovers analytical kinetic models of bioprocesses

This chapter is based on the following publication: Forster T., Vazquez D., Müller C., Guillén-Gosálbez G. (2024). Machine learning uncovers analytical kinetic models of bioprocesses. *Chemical Engineering Science*, 300, 120606

Nomenclature for Chapter 3

Sets

E	$\{e \mid e$ is a symbolic mathematical expression$\}$
I, J	$\{i, j \mid i, j$ are set components$\}$
U	$\{u \mid u$ is discrete sample point$\}$

Parameters

t_0, t_f	Inital and final time
$X_{0,i}$	Initial concentration of metabolite/species i
μ	Mean of a particular property
σ^2	Variance of a particular property
γ_e	Mathematical expression identified by the BMS
θ	Generic model parameters

Variables

BMS_i and ANN_i	BMS or ANN models for species i
p	Probability
Rxn_i	Generic reaction term (production or consumption of species i)
X_i	Concentration of metabolite/species i (used as continuous variables in ODE expressions)
$X_{i,u}$	Concentration of metabolite/species i at time t_u
$\hat{X}_{i,u}$	odel predictions of the concentration of metabolite/species i at time t_u
$\dot{X}_{i,u}$	Derivatives of metabolite/species i at time t_u
$\widehat{\dot{X}}_{i,u}$	Model predictions of the derivatives of metabolite/species i at time t_u
$\bar{X}_{i,u}$	Mean of the experimental data points of species i at time t_u
t and t_u	Time and sampled time point
ζ_i	Function to smooth noisy concentration profile for species i
$\dot{\zeta}_i$	Derivative of function to approximate derivative profile for species i
$\mathcal{DL}$	Description length (objective function of the BMS)

3.1 Introduction

In recent years, modeling has gained significant attention in the bioprocesses industry, spearheaded by the improvements in mathematical tools that can be used for analysis and optimization (M. R. Mowbray et al., 2023; Narayanan, Seidler, et al., 2021). Mathematical modeling can support scientists, engineers, or other subject matter experts in designing experiments (Sadino-Riquelme et al., 2020), predicting and monitoring processes (Del Rio Chanona, Wagner, et al., 2019; Rivera et al., 2007), and reducing development and production costs (Narayanan, Seidler, et al., 2021; Narayanan et al., 2020). Modeling complex bioprocesses, however, is a challenging task, particularly when first principles formulations are sought (Mercier et al., 2014; Petsagkourakis, Sandoval, et al., 2020; D. Zhang et al., 2020). These models are nevertheless being increasingly demanded by the market, in which the number of new products originating from bioprocesses is increasing very rapidly (Narayanan et al., 2023). Bioprocess modeling requires experimental measurements to calibrate an *in-silico* model by minimizing the mismatch between experimental observations and *in-silico* predictions. A common approach relies on well-established mathematical formalisms derived from first principles, such as mass or energy balances. Kroll et al. (2017) provide a workflow for the generation of mechanistic process models, where the authors start from material balances for a certain target variable and expand the models in a mechanistic manner with new states and interactions. They apply their method to a mammalian cell culture process to model the viable cell count. A more recent work by Sha et al. (2018) provides stoichiometric and kinetic models and some commonly used mathematical approaches to describe cell systems.

An alternative to purely mechanistic modeling approaches are data-driven strategies. These methods enable model building without relying on expert knowledge (Kahrs & Marquardt, 2007; Taylor et al., 2021). Here, the structure of the model is given by the surrogate modeling approach of choice. For example in the area of process control, Willis et al. (1995) applied an artificial neural network (ANN) to model the biomass concentration in a fermentation process. In a more recent work, Tonner et al. (2017) used Gaussian process models to describe the microbial growth in bioprocesses and interrogated the obtained models to investigate perturbation effects in the systems under study. As a bridge between purely deterministic and purely data-driven methods, hybrid modeling approaches, where

mechanistic knowledge is combined with a surrogate component, have also gained popularity (von Stosch et al., 2014). This approach has been applied to a wide range of problems in science and engineering. For example, Zhang et al. (2013) proposed a hybrid kinetic mechanism where quasi-steady-state species are separated from the kinetic ODEs. Gnoth et al. (2007, 2008a, 2010) integrated ANNs in kinetic models to approximate unknown behaviors of the microorganisms. More recently, hybrid frameworks for modeling bioprocesses have been put forward by Zhang (2019), and Mowbray (2023) and colleagues. In earlier works of the authors (Forster, Vázquez, Cruz-Bournazou, et al., 2023), a method for building models that are based on canonical kinetic representations (i.e., S-system (Savageau, 1969a, 1969b, 1970)) was discussed, where observed concentration data and a pre-defined canonical form for the rate expression were used to identify a suitable model structure and simultaneously estimate its parameters.

A key point in all the modeling approaches above is to define the model structure whose parameters will be calibrated via parameter estimation methods. Ideally, the model structure and its parameters should be simultaneously determined, since the choice of a specific model structure limits the accuracy of the model. However, in practice the model structure is first defined, hopefully through a mechanistic derivation of first principles, but sometimes through a surrogate formalism. Once the structure is chosen, its parameters are calibrated by solving a parameter estimation problem where the parameters values are the decision variables and the objective function is often given by the mismatch between *in-silico* and experimental observations. Works that optimize both the model structure and its parameters are quite scarce. A well-known example in the Process Systems Engineering (PSE) literature is the ALAMO approach for the automated learning of algebraic models (Wilson & Sahinidis, 2017). This algorithm creates closed-form surrogate models by solving a mixed-integer nonlinear programming (MINLP) problem where binary variables model the selection of specific algebraic terms from a set of allowable functions and continuous ones the associated parameters. Designed for dynamic systems, Brunton et al. (Brunton et al., 2016) proposed the SINDy (Sparse Identification of Nonlinear Dynamics) algorithm, which was successfully applied to different systems. By using sparse regression techniques, SINDy provides the user with an appropriate rate model for the ODE. Sun and Braatz (2020) developed an algorithm that combines nonlinear feature generation followed by sparse regression to learn interpretable nonlinear models, called al-

gebraic learning via elastic net (ALVEN). Other works, such as those by Willis and von Stosch (2017), use a problem-tailored approach for extracting ODEs from process data by formulating a hybrid semi-parametric modeling framework using mixed integer programming and multivariate rational functions. These modeling methods have the advantage of only requiring data and, therefore, can be set up without any expert knowledge about the system. Nonetheless, they assume a set of basis functions that must be combined linearly to form the algebraic expressions sought, which constrains the feasible set of plausible mathematical models that could explain given data.

Another approach for identifying closed-form expressions is symbolic regression (SR), which is based on the principles of genetic programming (Keane et al., 1993; Koza, 1994). In contrast to the main tools mentioned above, such as ALAMO, SINDy, or ALVEN, SR methods represent mathematical equations as expression trees (Cozad & Sahinidis, 2018). Employing a defined search procedure (i.e., mainly stochastic algorithms (Diveev & Shmalko, 2021) like a genetic algorithm (Cranmer et al., 2020) or Markov-chain Monte Carlo (MCMC) (Guimerà et al., 2020)), SR simultaneously identifies the tree structure and involved parameters in order to optimally represent observed data (Cozad & Sahinidis, 2018; Neumann et al., 2020). While previous approaches specified the basis functions, SR only requires a pool of allowed operators, and the functions are created from the available pool and given data. SR has been successfully applied in various fields, for example, McKay et al. (1997) used an SR approach to model a vacuum distillation column and a chemical reactor system. In a later work, the authors applied SR to develop a model of a food extrusion process (McKay et al., 1999). Vladislavleva et al. (2013) used an available software package named DataModeler (2023) to predict energy outputs of wind farms by considering weather data. Schmidt and Lipson (2009) discovered physical laws from experimental data using SR to identify nonlinear relationships. In recent contributions, researchers used SR to discover new perovskite catalysts (Weng et al., 2020) and to recover a variety of physical expressions (Udrescu & Tegmark, 2019). Other works resulted in commercially available SR software, such as Eureqa (Schmidt & Lipson, 2009) or TuringBot (2023). Cranmer et al. (2020) implemented an open-source SR algorithm named PySR (Cranmer, 2020) in Python that was applied to cosmology problems. Similarly, Guimerà et al. (2020) developed the Bayesian machine scientist (BMS), a SR algorithm based on an MCMC approach. These approaches were applied

in kinetic modeling for heterogeneous catalysis (De Carvalho Servia & Del Rio Chanona, 2023a; De Carvalho Servia et al., 2024), process design (Ferreira, Pedemonte, & Torres, 2019; Ferreira, Torres, & Pedemonte, 2019; Negri et al., 2022), process optimization (Forster, Vázquez, & Guillén-Gosálbez, 2023a, 2023b) or to model links between energy-related impacts and socioeconomic drivers (Vázquez et al., 2022).

Here, we apply SR techniques for kinetic model building in bioprocesses. In contrast to previous works that developed fully black-box or hybrid models based on standard surrogates (e.g., ANN and GPs) (Del Rio Chanona, Wagner, et al., 2019; Gnoth et al., 2010), here we apply SR to find a suitable kinetic expression and associated parameters. Specifically, our approach combines the BMS with a two-step decomposition algorithm inspired by the works of Miró (2014), Voit and Almeida (2004), Michalik et al. (2009), and Brendel et al. (2006). The goal is to identify reaction rates from observed concentration profiles of species, where the rate equation is determined via SR. De Carvalho Servia et al. (2024) recently applied SR using pySR (Cranmer, 2020) for heterogeneously catalyzed reactions. However, we here focus on bioprocesses and instead use the BMS for SR (Guimerà et al., 2020). Numerical examples show that the BMS can identify closed-form surrogate rate expressions that perform similarly compared to ANN-benchmark models. Following the successful application of the BMS in other problems, including the approximation of process simulations (Negri et al., 2022), process optimization (Forster, Vázquez, & Guillén-Gosálbez, 2023a), and the investigation of energy-related impacts and socioeconomic drivers in macroeconomic studies (Vázquez et al., 2022), here we show that it can also be used to find kinetic expressions that explain given data precisely.

The remainder of this article is organized as follows: First, the problem statement is described in detail. Subsequently, the proposed methodology is discussed. Afterward, the case studies are introduced, and the results are summarized. Finally, the conclusions of the work are drawn.

3.2 Problem statement

Without loss of generality, in this work, we consider a generic ideal batch reactor with constant volume V and different species $i \in I$ taking part in some reactions. The mass balance of such a system can be described by expression (3.1). In this

description, X_i might be the concentration of microbial cells or of a given species in the bioreactor, and $X = [X_1, X_2, \ldots, X_i]$ represents the vector of all metabolite concentrations. On the left-hand side of the equation, dX_i/dt (or $\dot{X}$), refers to the accumulation term.

$$\frac{d}{dt}X_i = \dot{X}_i = Rxn_i(X), \quad \forall i \in I \tag{3.1}$$

The $Rxn_i(X)$ term represents an expression that is unknown to the modeler and that depends on the concentration of all the species (state variables) collected in vector X. This is a common situation arising in bioprocess development, because the underlying metabolic pathways in such systems can be very complex (Guillén-Gosálbez et al., 2013; Mercier et al., 2014; Petsagkourakis, Sandoval, et al., 2020; D. Zhang et al., 2020). This complexity is given by the potentially large feedback loops between a wide range of species and the nonlinear nature of these interactions. In this work, we will approximate $Rxn_i(X)$ using a symbolic regression method that generates an algebraic expression without assuming any pre-defined structure of that reaction rate. Hence, here we do not rely on any canonical formalism to derive the kinetic model.

The goal, then, is to find a suitable expression for $Rxn_i(X)$ in equation (3.1) such that the mismatch between the model predictions and the experimental observations is minimized. Note that in this work, we assume that neither the structure of $Rxn_i(X)$ nor the involved parameters are known, unlike in a standard parameter estimation problem as discussed by Voit and Almeida (2004) or Brendel et al. (2006). Therefore, herein, we aim to find both, the rate expressions and their parameters simultaneously by only using the available concentration measurements. It is worth to mention that in the subsequently proposed approach, the modelling of a rate in the form $Rxn_i(X)$ for a species i is only possible for species that can be measured in the sampled data. If no data is available for species i, a parameter estimation and, therefore, a model building for such a species is not directly possible. Such a case might be encountered if some species have a shorter lifetime than the sampling frequency. Consequently, our modelling approach focuses on species that can be sampled, not on non-sampled or hidden species. The section that follows introduces our approach.

3.3 Methodology

In time-series-related problems, the concentrations (subsequently also called states) X_i are often considered to be continuous in time, i.e., $X_i(t)$. However, usually only discrete concentration values are available at the sampling times. Therefore, we consider a discrete notation based on a series of time points $u \in U$. The complete profile of one species i can therefore be described by expression (3.2).

$$X_{i,u} \in \left[X_{i,0}, X_{i,1}, X_{i,2}, \ldots, X_{i,|U|}\right], \quad \forall i \in I \tag{3.2}$$

From such a sampled array, we are interested in searching for a suitable model for the rate expression that can predict the time-dependent evolution by using the initial conditions at time t_0. This model-building task is typically formulated as a general dynamic optimization problem. In such an optimization problem, the sum of squared residuals (SSR) between the observed data point $X_{i,u}$ and the model prediction $\hat{X}_{i,u}$ is minimized, by optimizing the values of some unknown model parameters θ. The problem can therefore be formulated as given in expression (3.3)

$$\begin{aligned} \min_{\beta} SSR &= \sum_{i \in I} \sum_{u \in U} \left(X_{i,u} - \hat{X}_{i,u}\right)^2 \\ s.t. \widehat{\dot{X}}_{i,u} &= \mathcal{M}_i\left(X_{j,u}, \theta\right), \quad \forall i \in I,\ j \in I,\ u \in U \\ \hat{X}_{i,0} &= X_{i.0}, \quad \forall i \in I \\ \hat{X}, \widehat{\dot{X}} &\in \mathbb{R}^+ \end{aligned} \tag{3.3}$$

in equation (3.3), the predicted derivative $\widehat{\dot{X}}_{i,u}$ of species i at time point u is calculated by a model $\mathcal{M}_i(X, \theta)$ with some trainable parameters θ that well approximate the underlying reaction rate $Rxn_i(X)$. The model building process to approximate $Rxn_i(X) \approx \mathcal{M}_i(X, \theta)$ is discussed below. The initial conditions $X_{i,0}$ are usually known values. However, finding the concentration profiles $X_{i,u}$ for a given system requires solving the ODEs, either simultaneously or sequentially. The latter might lead to stiff ODEs and can therefore often make numerical integration very difficult and inefficient (Tjoa & Biegler, 1991). Moreover, effectively handling the existence of binary variables in this approach would remain chal-

lenging. Michalik et al. (Michalik et al., 2009) and Voit and Almeida (Voit & Almeida, 2004) proposed alternative approaches to simplify the dynamic problem shown above based on a reformulation of the original model in the derivative space instead of the state space. This reformulation is given in expression (3.4).

$$\begin{aligned} \min_{\beta} SSR &= \sum_{i \in I} \sum_{u \in U} \left(\dot{X}_{i,u} - \widehat{\dot{X}}_{i,u} \right)^2 \\ s.t. \widehat{\dot{X}}_{i,u} &= \mathcal{M}_i \left(X_{j,u}, \theta \right), \quad \forall i \in I,\ j \in I,\ u \in U \\ \hat{X}_{i,0} &= X_{i.0}, \quad \forall i \in I \\ \hat{X}, \widehat{\dot{X}} &\in \mathbb{R}^+ \end{aligned} \tag{3.4}$$

To solve the problem given in expression (3.4), the derivatives $\dot{X}_{i,u}$ have to be obtained from the discrete time profiles of the observed state variables $X_{i,u}$. Such derivates can then be subsequently used to train a suitable kinetic model $\mathcal{M}_i \left(X, \theta \right)$. This strategy avoids integrating the dynamic system in expression (3.3), at the expense of performing the regression in the space of reaction rates, which poses some challenges concerning the computation of derivatives leading to low errors in the original dynamic space of state variables. This is because the derivatives determined experimentally can be affected by experimental errors, which may lead to good predictions in the reaction rates space but poor in the original states variables space.

The method of choice follows an incremental approach for building the surrogate model, as shown in Figure 3.1, where the details of the steps are given below in Section 3.3.1.

The discussed procedure starts with collecting noisy concentration data $X_{i,u}$ for different species i and times u. To smooth out the noise in the measurements, a univariate function in time $\zeta_i(t)$ is fitted to the data. In the second step, this identified function $\zeta_i(t)$ can be derived analytically and the derivatives $\dot{\zeta}_{i,u}$ can be evaluated at the experimental time points. Third, the state values $X_{i,u}$ are linked to the calculated derivatives $\dot{\zeta}_{i,u}$ by an appropriate model found via SR. The model, therefore, approximates the $Rxn_i(X)$ term given in equation (3.1). Last, the trained models can be incorporated into a system of ODEs, which is

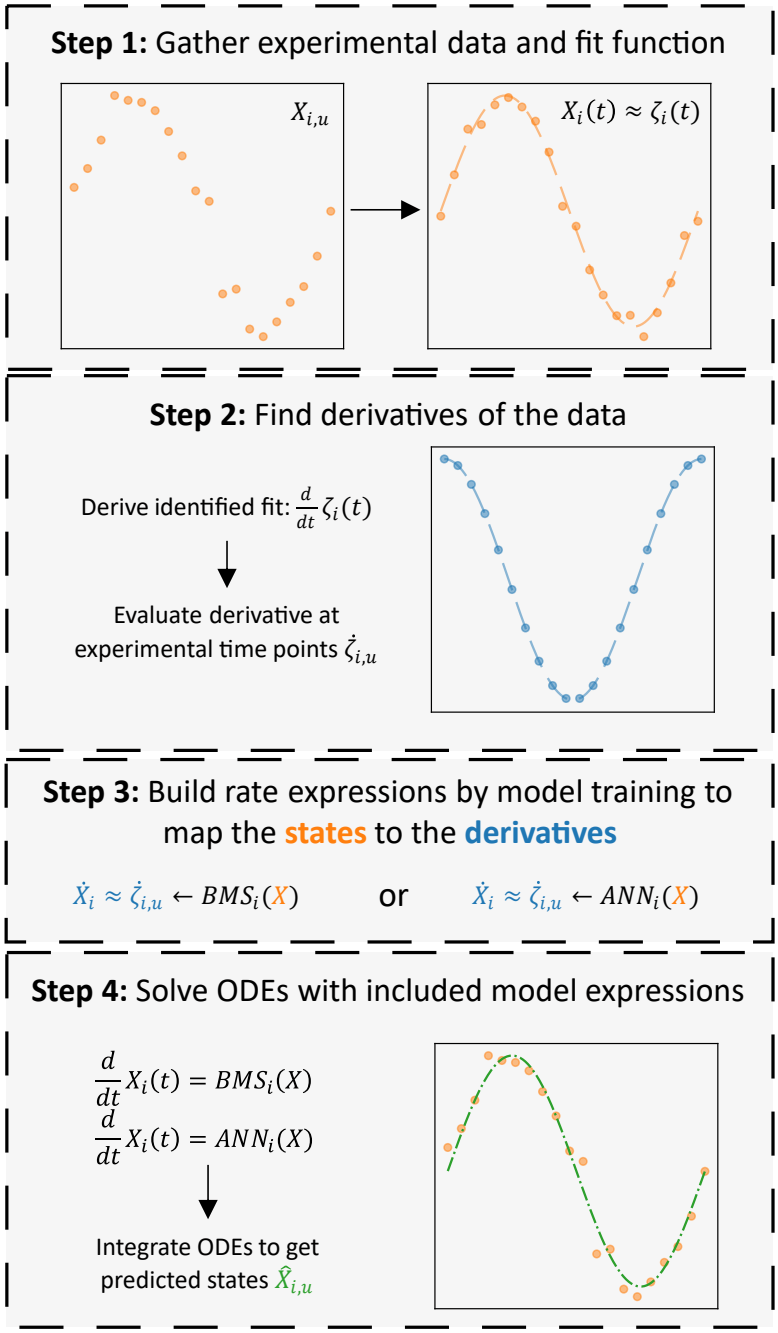

Figure 3.1. Overview of the approach for building a rate expression. In the first step, a function $\zeta_i(t)$ is fit to data points for each species i. The functions $\zeta_i(t)$ are then derived analytically (step 2). In step 3, models (BMS or an ANN) are trained to map the states to the calculated derivatives $\dot{\zeta}_{i,u}$. Last, in step 4, the models are incorporated into a system of ODEs which can be solved with appropriate initial conditions.

solved using specific initial conditions. In the following subsection, these four steps are discussed in more detail.

3.3.1 Incremental approach for model building

The procedure is schematically shown in Figure 3.1. There are several possible ways to derive data numerically. A comparison of three possible methods to derive a noisy sinusoidal signal is given in Figure 3.2. The simplest method is the differentiation via forward finite differences. The main disadvantage of this approach is the amplification of noise during the derivation process. Therefore, a smoothing step is preferred before differentiating noisy data, for example using a Savitzky-Golay filter (Savitzky & Golay, 1964). Here, however, we used instead a polynomial or a univariate BMS to fit a function $\zeta_i(t)$ to the noisy data, as given in expression (3.5). The polynomial approach was successfully demonstrated in an earlier work by the authors (Forster, Vázquez, Cruz-Bournazou, et al., 2023). The symbolic fit using the univariate BMS was inspired by De Carvalho Seriva et al. (2024), where the authors demonstrated an approach for fitting and deriving the observed data. In the present work, we adapted this approach and use a different toolbox.

$$X_i\left(t\right) \approx \zeta_i = \begin{cases} p_{i,1} + p_{i,2}t + \ldots + p_{i,q+1}t_u^q \\ BMS_i(t) \end{cases} \quad , \quad \forall i \in I \tag{3.5}$$

In the case of the polynomial approach, the unknown parameters p have to be regressed to the noisy data, while when using the BMS the structure and parameters are both to be found. Both, the polynomial and the algebraic expression identified by the BMS are univariate in time. In both cases, the resulting expressions can subsequently be derived analytically, as given in equation (3.6). The derivatives can be evaluated at the experimental time points $t_u,\ u \in U$.

$$\dot{X}_i\left(t\right) \approx \dot{\zeta}_i\left(t\right) = \frac{d}{dt}\zeta\left(t\right), \quad \forall i \in I \tag{3.6}$$

Steps 1 and 2: Fitting univariate function and estimating derivatives. In the case of the polynomial approach, we defined a set of polynomial degrees $q \in Q$. The different polynomials are fit to the noisy data and the corresponding

Bayesian information criteria (BICs) are calculated. The polynomial with the lowest BIC is subsequently differentiated analytically as given above. In the case of the univariate BMS, we defined a threshold for the coefficient of determination (R^2). The BMS is trained with a given number of steps (discussed in more detail below). If the R^2-threshold is not reached, then the training steps are doubled. This procedure is repeated for a given number of times at most. After that, the identified algebraic expression can be derived analytically. As shown in Figure 3.2 (b), the approximated derivatives are more accurately calculated by the smoothing methods given in expressions (3.5) and (3.6) compared to forward finite difference differentiation. However, the first and last sample points might still comprise some error even after applying such smoothing techniques. To reduce this noise impact further, one possibility is to disregard the initial and last sample points for the subsequently discussed model training, which is also used in other works (Willis & von Stosch, 2017).

An in-depth analysis of how the derivative approximation methods perform under different noise levels and data set sizes is given in the supplementary information Section B.1. The results summarized in Figure B.1 show that the symbolic estimation approach given in expression (3.5) seems to work well suited even in presence of noise and scarce data sets.

Step 3: Building the rate expression. Rate expressions map some discrete states $X_{i,u}$ to the obtained derivatives $\dot{\zeta}_{i,u}$. The identified model, therefore, is intended to predict the changes in concentration of the species at a given time. To identify this model, we use an SR tool, the BMS. Upon model training, the BMS identifies an algebraic expression that approximates the reaction term as given in expression (3.7). To benchmark our results, we compare them to those from an ANN, as shown in expression (3.8). The reason for this choice is that ANNs are generally regarded as good approximators (Psichogios & Ungar, 1992). Additional details on the symbolic regression tool are discussed below in Section 3.3.2.

$$Rxn_i\left(X\right) \approx \mathcal{M}_i\left(X\right) = BMS_i\left(X\right), \quad \forall i \in I \tag{3.7}$$

$$Rxn_i\left(X\right) \approx \mathcal{M}_i^{benchmark}\left(X\right) = ANN_i\left(X\right), \quad \forall i \in I \tag{3.8}$$

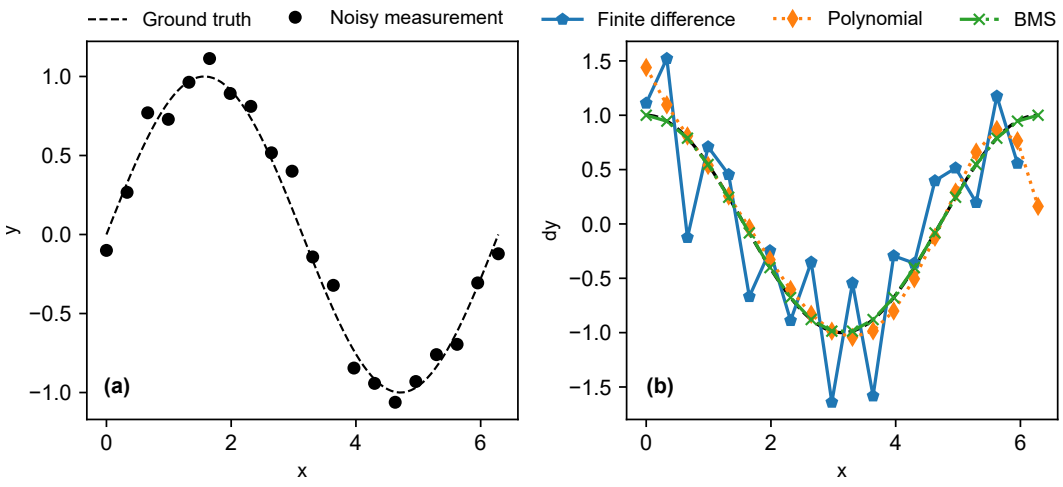

Figure 3.2. (a) Noisy measurements (circles) together with the underlying sinusoidal ground truth (dashed line). (b) Comparison of numerical differentiation methods. The dashed black line represents the cosinusoidal ground truth of the derivative (covered up by the BMS approach). The blue pentagons with the solid line represent the derivatives by forward finite differences. The orange diamonds with the dotted line represent the derivatives obtained by the polynomial approach discussed above. The green crosses with the dashed-dotted line represent the derivatives of the BMS approach.

Step 4: Solving the ODE model with the built rate expressions. The fully trained models can be incorporated into the ODE in expression (3.1), resulting in the ODEs given in equation (3.9) for the BMS approach, and in equation (3.10) for the ANN approach.

$$\frac{d}{dt}X_i = BMS_i\left(X\right), \quad \forall i \in I \tag{3.9}$$

$$\frac{d}{dt}X_i = ANN_i\left(X\right), \quad \forall i \in I \tag{3.10}$$

These ODEs can be solved for the initial conditions $X_{i,0}, i \in I$ and considering an integration period $t = [t_0,\ t_f]$.

3.3.2 Background to the Bayesian machine scientist

In this work, we do not assume any pre-defined model structure to search for suitable rate expressions. Upon model training, the BMS returns an algebraic closed-form expression, which can subsequently be incorporated into the system of ODEs to be integrated. We now provide an overview of how the BMS works.

For further information, the reader is referred to the original paper (Guimerà et al., 2020). The algorithm identifies a suitable mathematical expression by searching through a space of expressions represented as symbolic trees. To perform the search through this space of expressions, several allowable moves from an initial tree can be done by the algorithm.

The space of possible mathematical expressions γ is described by E. Starting from one symbolic representation $\gamma_e, e \in E$, we perform changes in the tree leading to different mathematical expressions. One example of such a tree evolution is shown in Figure 3.3 (a). The addition of the two main terms in γ_1 is replaced by a multiplication, which leads to the expression γ_2. A further replacement of the addition in γ_2 leads to the expression γ_3, which explains the observed data points (black circles) better than γ_1 or γ_2. Another adaptation would be the elementary tree replacement (i.e., exchanging the complete sub-tree $(\beta + \delta)$ by another sub-tree). For each resulting expression, a goodness-of-fit metric can be calculated. The SR algorithm then proceeds to search the space of expressions, seeking the expression with the best goodness of fit. This search is stochastic, as in other evolutionary algorithms (Costa & Oliveira, 2001; Guimerà et al., 2020).

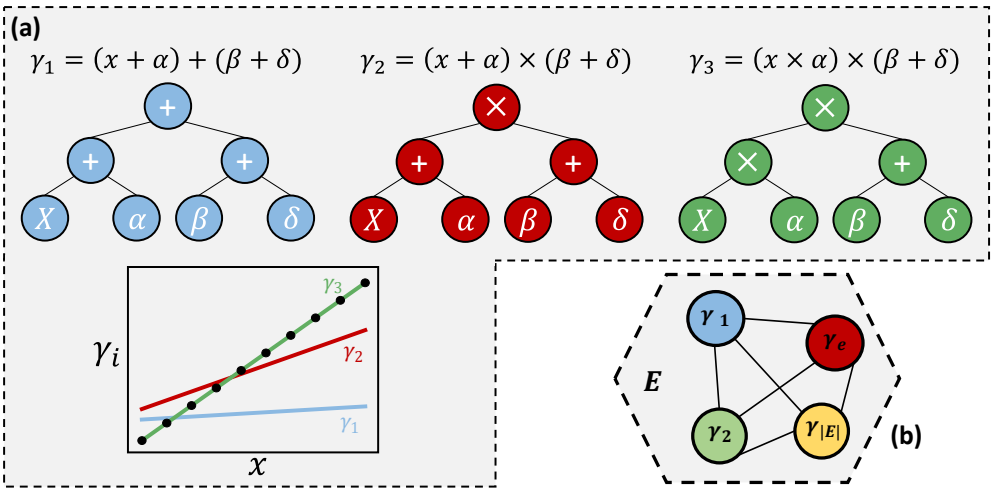

Figure 3.3. (a) Several equations are represented as symbolic trees. From $\gamma_1 = (x_1 + \alpha) + (\beta + \delta)$, a node replacement can be performed to reach $\gamma_2 = (x_1 + \alpha) \times (\beta + \delta)$. A further node replacement can be done to obtain the equation $\gamma_3 = (x_1 \times \alpha) \times (\beta + \delta)$. The expression for γ_3 (green line) ends in the best possible model to fit the data (black circles) compared to γ_1 (blue line) and γ_2 (red line) in the lower part of the figure. (b) The space E of all possible expressions γ_e is schematically shown as a dashed polygon.

The BMS can provide closed-form algebraic expressions from data based on a set of user-defined mathematical operations (i.e., addition, subtraction, multiplication,

etc.). In the algorithm, a conditional probability $p(\gamma_e|D)$ is assigned to each expression γ_e, $e \in E$ (the space of symbolic trees shown schematically in Figure 3.3 (b)) used to fit some data D, which is calculated according to Bayes Theorem (Bishop, 2006; Murphy, 2013), given by equation (3.11):

$$p\left(\gamma_e|D\right) = \frac{p\left(D|\gamma_e\right)\ p(\gamma_e)}{p(D)} \tag{3.11}$$

Where $p(D)$ represents the marginal likelihood of some data D. $p(D)$ is independent of γ_e and therefore acts as a normalization constant. Marginalizing over the parameters ϕ_e associated with expression γ_e (Murphy, 2013), the numerator in expression (3.11) can be expressed as an integral over the space of all possible parameter values Φ_e (Guimerà et al., 2020). The description length $\mathcal{DL}\left(\gamma_e\right)$ then describes the resulting integral (Guimerà et al., 2020; Hansen & Yu, 2001; Murphy, 2013), given in equation (3.12).

$$\mathcal{DL}\left(\gamma_e\right) = -\log\left[\frac{1}{p(D)}\int_{\Phi_e} d\phi_e p(D|\gamma_e,\phi_e)p(\phi_e|\gamma_e)p(\gamma_e)\right] \tag{3.12}$$

Computing the numerical value of the integral included in the description length is challenging (Guimerà et al., 2020; Murphy, 2013). It has been shown (Grünwald, 2007; Murphy, 2013) that the entire metric can be approximated through the Bayesian information criterion (BIC) and the prior of the corresponding symbolic expression γ_e, as shown in expression (3.13)

$$\mathcal{DL}\left(\gamma_e\right) \approx \frac{BIC(\gamma_e)}{2} - \log\left(p\left(\gamma_e\right)\right) \tag{3.13}$$

Therefore, the plausibility of observing an expression γ_e conditioned on some data D is obtained by the description length $\mathcal{DL}(\gamma_e)$. In other words, during the stochastic search, the description length (and therefore a metric for the plausibility of observing an expression γ_e) serves as objective function which is being minimized. As visible in expression (3.13), to compute the description length, the prior knowledge about expression γ_e is required as $p(\gamma_e)$. Guimerà et al. (2020) used a pre-defined corpus of equations from Wikipedia. After parsing the publicly available equations, the number of operations were counted that were present

in the expression. Based on this information, they created distributional information about operators in equations, which were subsequently used as the prior distributions $p(\gamma_e)$ (Guimerà et al., 2020).

According to Grünwald (2007), $\mathcal{DL}(\gamma_e)$ can be understood as an encoded length of the expression γ_e (number of natural units). A Markov-chain Monte Carlo (MCMC) (Hastings, 1970) algorithm is used to explore the space E of expressions, where the number of MCMC iterations is defined by the user. After evaluating the description length of each expression $\mathcal{DL}(\gamma_e)$ – which represents the goodness-of-fit metric and therefore the objective function – the BMS keeps the most plausible one, representing the expression with the shortest description length (the best goodness-of-fit).

3.3.3 Model performance metrics

For assessing the performance of the models, an arbitrary set of initial conditions can be used to integrate the ODE, comparing the simulated and experimental profiles values. After training the models on dedicated training runs, separated test runs were used to assess their performance. A detailed description of how the data is generated and split into training and test sets is shown in Section 3.4.

The performance is assessed via the root mean squared error ($RMSE$) and the coefficient of determination (R^2), defined as given in equations (3.14) and (3.15). These metrics can be calculated for both the training and test sets, obtaining the training and test errors, respectively. They can be calculated for the concentration (state) space or the derivative space.

In these relationships, the predictions by the model are described by $\hat{X}_{i,u}$. The experimental data points and the mean of the experimental data points of the data are described by $X_{i,u}$ and $\bar{X}_{i,u}$, respectively. Variables SSR and SST denote the sum of squares of residuals and the total sum of squares (proportional to the variance of the data), respectively. The error metrics in expressions (3.14) and (3.15) can be calculated for the state and derivative variables.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i\in I}\sum_{u\in U}\left(\hat{X}_{i,u} - X_{i,u}\right)^2} \tag{3.14}$$

$$R^2 = 1 - \frac{SSR}{SST} = 1 - \frac{\sum_{i\in I}\sum_{u\in U}\left(\hat{X}_{i,u} - X_{i,u}\right)^2}{\sum_{i\in I}\sum_{u\in U}\left(X_{i,u} - \bar{X}_{i,u}\right)^2} \tag{3.15}$$

3.3.4 Implementation details

All calculations were carried out on an Intel®Core™ i7-8700 CPU and 16 GB of RAM. To construct the sampling dataset, we used Python 3.10 with NumPy v1.24.3 and pyDOE v0.3.8. For the BMS training, the hyperparameter values are those given in the original article of the BMS (Guimerà et al., 2020), i.e., 5% probability of root replacement, 45% probability of node replacement, and 50% probability of elementary tree replacement. The allowed unitary operations included $\exp(x)$, $\log(x)$, x^2, x^3, and $\sqrt{x}$, while the binary operations consisted of $+$, $-$, $\div$, $\times$, x^y. The maximum number of MCMC steps was chosen to be 10^4. The neural network training was performed with Scikit-learn v1.0.2 (Pedregosa et al., 2011). A grid search with a 3-fold cross-validation was performed to tune and find appropriate hyperparameters for this benchmark model. Parameters considered during the grid search were the hidden layer size, the activation function, the learning rate, and the initial learning rate. Details of this grid search and the settings of the fixed hyperparameters are given in Section B.3 of the supporting information.

3.4 Case studies

Subsequently, two different case studies are presented. We employed Latin hypercube sampling (LHS) together with the bounds given in Table 3.1 to generate different initial conditions, with each set of initial conditions representing a different batch. For each case study shown below, 13 batch runs were simulated in total. From those, 10 batches were used to train the models and 3 were taken as test batches. We added normally distributed noise (NumPy) with a mean of $\mu = 0$ and a variance of $\sigma^2 = 0.2$ to the profiles obtained from integrating the different batches to create more realistic data (more significant noise level in lower numerical ranges to resemble measurement errors). For the two case studies, sev-

eral scenarios were considered, which are summarized in Figure 3.4. To study the influence of the amount of data available, we generated profiles with 40 and 20 time points per batch. It is worth to be mentioned that the time spans of the subsequently introduced case studies are 80 hours and 180 hours, respectively. A sampling rate of 20 points within this time frame results in one sample every two hours and every approximately 9 hours. Indeed, it should be kept in mind that a reduction of the sampling frequency will result in a reduction in accuracy of the derivative approximation, which is discussed in more detail in the supporting information Section B.1. To calculate the derivatives from the data, the polynomial fit or symbolic regression fit, both described in expression (3.5), were used. Hence, four different scenarios for each case study were explored. The resulting scenarios are described by the abbreviations Poly-20, Poly-40, SR-20, and SR-40, depending on the number of points per batch and the method for derivative approximation (3.4). As an example, CSI-Poly-40 describes the scenario of CSI with the polynomial approach for the derivative calculation and 40 samples per batch and species. The case studies are also collected and published on GitHub (https://github.com/forstertim/insidapy).

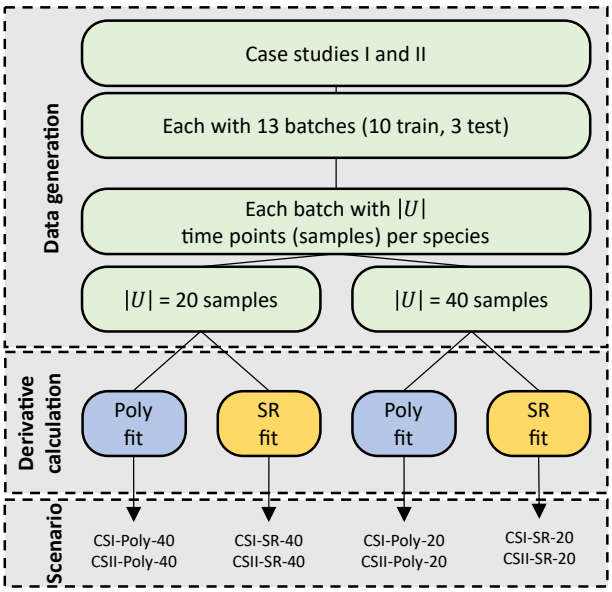

Figure 3.4. An overview of the organization of the case studies is shown schematically. For each of the base case studies discussed below, batches with either 20 or 40 samples were generated. Then, either the polynomial or symbolic regression approach was applied to calculate the derivatives.

3.4.1 Case study I

A bioprocess is considered where some bacteria produce a specific product while consuming a substrate. The mass balances are given in expressions (3.16), where the variables B, S, and P (all in $\mathrm{g\,L^{-1}}$) represent the biomass, substrate, and product concentration, respectively. These species are summarized in the vector $X = \{B, S, P\}$. The process is modeled in batch mode and was adapted from Turton et al. (2018).

In these mass balances, ϕ ($\mathrm{h^{-1}}$) models the growth rate. $Y_{i,j}$ represents the yield coefficient of species j with respect to species i. The expressions $\Omega(A, T, E_A)$ represent Arrhenius reaction rates that depend on the temperature T and temperature-independent pre-factors and activation energies $E_{A,1}$, $E_{A,2}$, A_1, and A_2, respectively. The parameters K_s and K_ϕ represent the half-saturation constants. Data was generated for the interval $t = [0, 80]$ h. The values of the parameters are given in Table B.3 of the supporting information. As mentioned in Section 3.3.1, the first two and last five points (polynomial approach) or the first two and last two points (BMS approach) were excluded for model training.

$$\begin{aligned} \frac{dB}{dt} &= \phi \cdot B \\ \frac{dS}{dt} &= -\frac{1}{Y_{B,S}} \cdot \phi \cdot B \\ \frac{dP}{dt} &= \frac{Y_{P,S}}{Y_{B,S}} \cdot \phi \cdot B \\ \text{with } \phi &= \phi_{max} \cdot \frac{S}{K_S + S} \cdot \frac{\Omega\left(A_1, T, E_{A,1}\right)}{1 + \Omega\left(A_2, T, E_{A,2}\right)} \cdot \left(1 - \frac{B}{K_\phi + B}\right) \end{aligned} \tag{3.16}$$

3.4.2 Case study II

Here, we focus on a bioprocess studied by Del Rio Chanona (2019). The system of ODEs in expressions (3.17) is based on a Monod model, a Logistic model, and a Luedeking-Piret model (D. Zhang, Dechatiwongse, Del Rio Chanona, et al., 2015), where cell growth, cell decay, and substrate uptakes are considered. For a detailed description, the reader is referred to the work of Del Rio Chanona (2019).

$$
\begin{aligned}
\frac{dB}{dt} &= \mu \frac{N}{N+K_N}\frac{C}{C+K_C}\frac{P}{P+K_P}B - \mu_d B^2 \\
\frac{dC}{dt} &= -Y_{C1}\left(\mu \frac{N}{N+K_N}\frac{C}{C+K_C}\frac{P}{P+K_P}B - \mu_d B^2\right) - Y_{C2}B \\
\frac{dN}{dt} &= -Y_{N1}\left(\mu \frac{N}{N+K_N}\frac{C}{C+K_C}\frac{P}{P+K_P}B - \mu_d B^2\right) - Y_{N2}B \\
\frac{dP}{dt} &= -Y_{P1}\left(\mu \frac{N}{N+K_N}\frac{C}{C+K_C}\frac{P}{P+K_P}B - \mu_d B^2\right) - Y_{P2}B
\end{aligned}
\tag{3.17}
$$

In this system, the variables B, C, N, and P represent the biomass, carbon, nitrogen, and phosphate concentrations, respectively (all in $\mathrm{mg\,L^{-1}}$). These species are summarized in the vector $X = \{B, C, N, P\}$. The parameters K_N, K_C, and K_P represent the half-velocity coefficient of the corresponding substrates, where the parameters Y_{i1} and Y_{i2} are growth-dependent and growth-independent yield coefficients of the species $i = C, N, P$. The biomass growth and death are given by μ and μ_d. The concentration of the biomass is divided by 1000 so that the originally reported parameter values can be used (D. Zhang, Dechatiwongse, Del Rio Chanona, et al., 2015). The time window investigated corresponds to $t = [0,\ 180]$ h. The values of the parameters are given in Table B.4. As in CSI, the first two and last five points (polynomial approach) or the first two and last two points (BMS approach) were excluded for model training.

3.5 Results

Below, the results of the BMS are compared to the ones obtained with the ANN. A summary of the obtained coefficients of determination (R^2) for the model training and testing is given in Table 3.2. The performance metrics are displayed for the

Table 3.1. Lower and upper bounds used for generating the training and test sets. With those bounds and a Latin Hypercube Sampling approach, different initial conditions were generated. These were used to solve the systems of ODEs in expressions (3.16) and (3.17) to create different batch runs.

	CSI			**CSII**			
Species	B	S	P	B	C	N	P
Lower bound	0.1	50	0	216	108	450	17
Upper bound	0.4	90	0	264	132	550	21
Unit	$\mathrm{g\,L^{-1}}$	$\mathrm{g\,L^{-1}}$	$\mathrm{g\,L^{-1}}$	$\mathrm{mg\,L^{-1}}$	$\mathrm{mg\,L^{-1}}$	$\mathrm{mg\,L^{-1}}$	$\mathrm{mg\,L^{-1}}$

different scenarios (as visualized in Figure 3.4), while results are also depicted in Figure 3.6 for CSI and Figure 3.7 for CSII. These plots show the calculated derivative values against the model predictions for both modeling approaches (BMS and ANN). Additional results are given in Section B.5 of the supporting information.

In general, both models achieve similar performance in both the derivative and the state (concentration) space, with our approach often outperforming the ANN, but not by much, as discussed next. Recall that the models should not only train well in the derivative space but also after integration since we are interested in predicting concentration profiles. Therefore, we focus first on the model with best performance in the state space of the unseen test batches (in Table 3.2, the highest R^2 value of the test set is highlighted in bold). The best-performing models are identified by the BMS in most scenarios, although the differences with the ANN are small. The only exception where the ANN outperforms the BMS is in CSI-SR-40, although also there, differences are marginal. In addition to the data given in Table 3.2, Figure 3.5 shows one of the test batches results for the models identified in scenarios CSI-Poly-40 (top row) and CSI-Poly-20 (bottom row). As shown in this figure, the models are well able to predict the evolution of the concentration, even if a lower sampling frequency was used (20 vs. 40 samples). From the results shown in Table 3.2, having fewer data points per batch does not significantly impact the performance of the models. Also, there was no clear difference in performance when comparing the two differentiation approaches.

Considering the reaction rates space, both models lead to very similar performance in all scenarios. Interestingly, although trained only in the derivative space, both models can predict well after integration. This would support the assumption that the rate expressions can be well approximated by both models.

Although both models seem to perform similarly throughout the case studies, there is one significant advantage of using BMS. After identification of the rate expression, the model is provided in analytical form and can be, arguably, interpreted more easily than purely data-driven models. For CSI (CSI-Poly-40), the most plausible expressions obtained by the BMS for the ODE system are given in expressions (3.18)-(3.20) as an example. Additionally, the corresponding estimated values of the parameters in expressions (3.18)-(3.20) are given in Table 3.3. The identified BMS models with the corresponding estimated model parameters

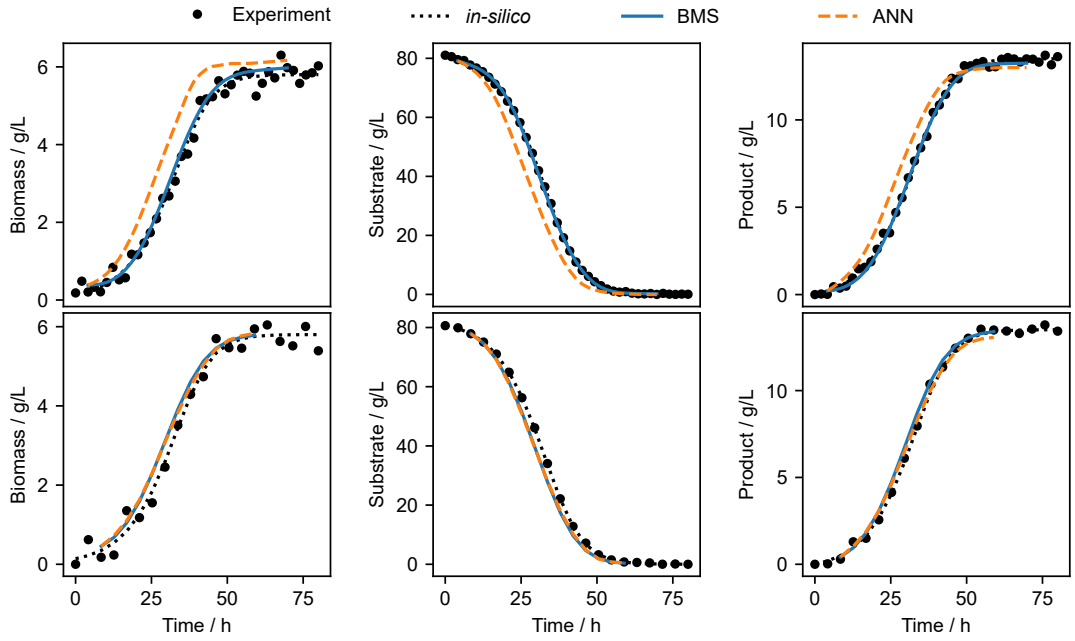

Figure 3.5. The concentration profiles of the three species in CSI are shown together with the model predictions. The top row represents the scenario CSI-Poly-40, whereas the bottom row represents the scenario CSI-Poly-20. The black circles represent the observed noisy data, where the dotted line represents the underlying ground truth. The dashed orange line represents the ANN prediction, whereas the blue solid line represents the BMS predictions. It is worth mentioning the model predictions are only shown for the experimental time points that were used for model training, since some initial and last samples were removed from the training, as discussed in Section 3.3.1.

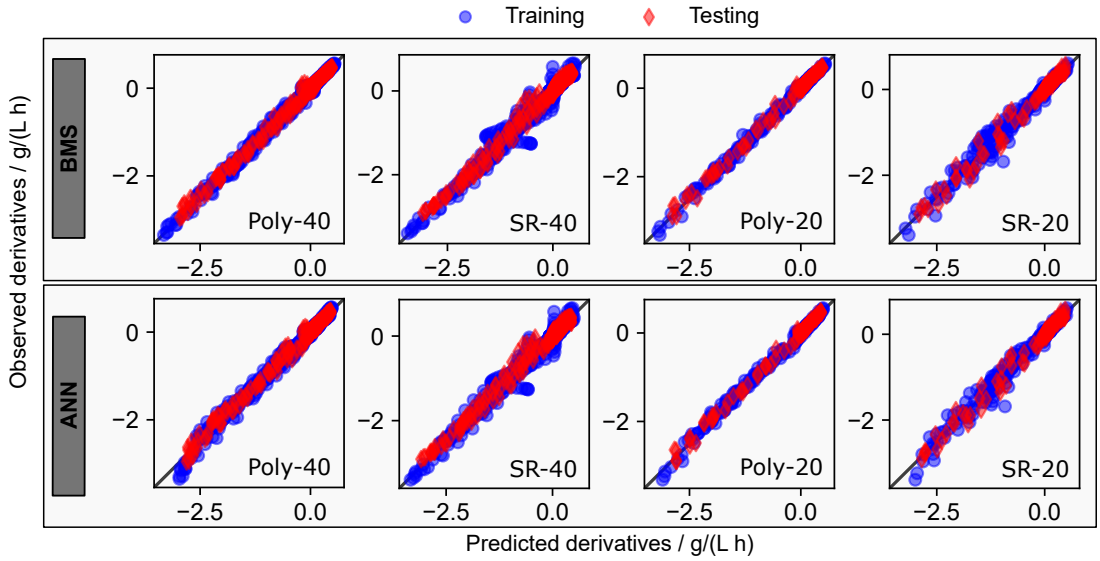

Figure 3.6. The observed values are plotted against the model prediction values for CSI. The columns represent the different scenarios of the case study. The top row shows the results obtained from the BMS predictions, whereas the bottom row shows the results from the neural network. Blue circles represent the training data, whereas red diamonds correspond to the test data. The black line represents the values where the observed value corresponds to the model predictions.

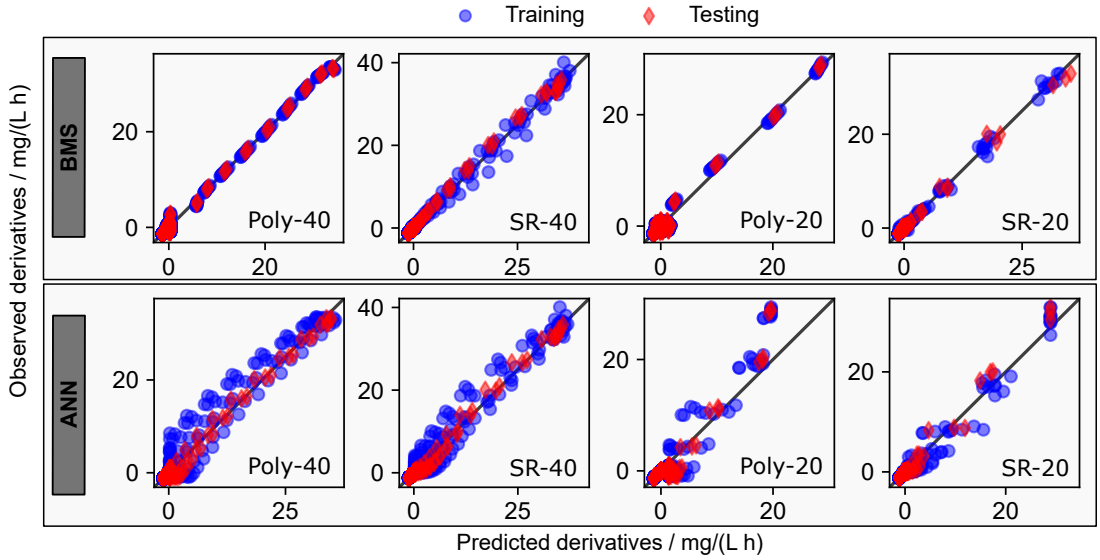

Figure 3.7. The observed values are plotted against the model prediction values for CSII. The columns represent the different scenarios of the case. The top row shows the results obtained from the BMS predictions, whereas the bottom row shows the results from the neural network. Blue circles represent the training data, whereas red diamonds correspond to the test data. The black line represents the values where the observed value corresponds to the model predictions.

Table 3.2. The coefficients of determination (R^2, unitless) are shown for the training and testing runs (notation: train/test) for the two case studies and their respective scenarios. For each case scenario, the best-performing approach in terms of state-space performance is indicated in bold text. The CPU times for the model training are indicated as mean values of the times for training models of the different species. In the ANN case, the time for the grid search is included. The raw values of the CPU times are indicated in Section B.5 of the supporting information.

CS	Scenario	CPU model training [s]		State		Derivative	
		BMS	ANN	**BMS**	**ANN**	**BMS**	**ANN**
I	Poly-40	7271	88	0.961 / **0.999**	0.837 / 0.986	0.995 / 0.995	0.992 / 0.994
	SR-40	4132	88	0.977 / 0.900	0.959 / **0.990**	0.979 / 0.990	0.982 / 0.988
	Poly-20	5803	59	0.998 / **0.995**	0.994 / 0.994	0.997 / 0.996	0.996 / 0.996
	SR-20	8167	57	0.993 / **0.989**	0.988 / 0.984	0.981 / 0.986	0.983 / 0.988
II	Poly-40	9384	130	1.000 / **1.000**	0.996 / 0.998	0.995 / 0.995	0.958 / 0.989
	SR-40	9275	162	1.000 / **1.000**	0.997 / 0.999	0.995 / 0.997	0.973 / 0.993
	Poly-20	2120	155	1.000 / **1.000**	0.996 / 0.996	0.986 / 0.986	0.885 / 0.901
	SR-20	4492	151	1.000 / **1.000**	0.999 / 0.999	0.997 / 0.994	0.971 / 0.978

for the other scenarios are summarized in Section B.6 of the supporting information.

$$\frac{dB}{dt} = a_0 \left((S \cdot B)^{a_1 + \left(\frac{a_2}{a_2 + P}\right)} \right) \tag{3.18}$$

$$\frac{dS}{dt} = a_0 + a_2 - a_1 \left(P + B^{S \cdot a_1}\right) \cdot S^{\left(\frac{1}{S \cdot P}\right)^{a_0}} \tag{3.19}$$

$$\frac{dP}{dt} = -\left(\frac{a_2 \left(B \cdot P\right)^{a_1}}{S^{a_0}}\right) \left(\left(\frac{a_2}{P}\right) + \frac{a_0}{S + (a_1^S)}\right) \tag{3.20}$$

After the model training and the deployment for predicting the time dependency of the concentration profiles, one can analyze the obtained ODEs to gather some

Table 3.3. Parameter values of the most plausible algebraic models identified by the Bayesian machine scientist for each case study given in expressions (3.18)-(3.20).

Parameter	Rate equation for CSI		
	dB/dt	**dS/dt**	**dP/dt**
a_0	$4.164 \cdot 10^{-3}$	$5.343 \cdot 10^{-2}$	$-1.797 \cdot 10^{0}$
a_1	$8.285 \cdot 10^{-1}$	$2.072 \cdot 10^{-2}$	$4.059 \cdot 10^{-1}$
a_2	$-1.086 \cdot 10^{-1}$	$2.302 \cdot 10^{-2}$	$4.175 \cdot 10^{-3}$

qualitative knowledge from those closed-form expressions. This will be shown with the example of the biomass growth and the substrate consumption. It is worth to be mentioned that this analysis is done on a conceptual and qualitative level to extract some knowledge and trends about the underlying system.

Considering the growth of the biomass B in equation (3.18), all three species - B, S, and P - seem to influence the change in biomass concentration. These findings can be interpreted using the underlying ground truth model in (3.16), which was used to generate the noisy data. In this underlying ground truth model the product concentration P is not involved in the rate equation of the biomass. Nevertheless, the BMS equation takes also P into account in (3.18). However, taking a closer look at the exponent in this equation, namely $a_1 + (a_2/(a_2 + P))$, one can observe Monod-type similarities with an asymptotic behavior. The value of this entire exponent converges towards a given value a_2, which is displayed in Figure 3.8 (a).

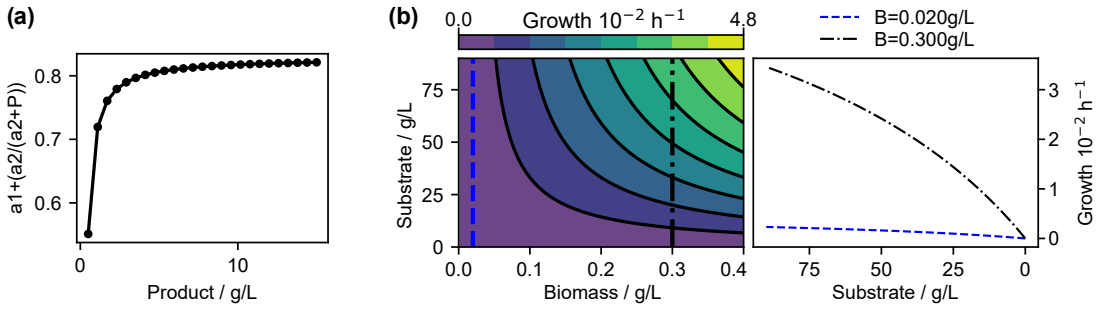

Figure 3.8. (a) The exponent in equation (3.18) is shown as a function of the product concentration P. In (b), the biomass growth $\phi \cdot B$ given by the underlying system in equation (3.16) is visualized as a function of the substrate concentration S and the biomass concentration B. Additionally, in (b), two scenarios are highlighted by the blue dashed (constant biomass of $0.02\,\mathrm{g\,L^{-1}}$) and black dotted-dashed lines (constant biomass of $0.30\,\mathrm{g\,L^{-1}}$), for which the growth is shown as a univariate function of the substrate concentration.

Although the BMS considers the product in the identified model expression for dB/dt, the effect of a change in P is more significant in the beginning and becomes less important throughout the reaction (when the product is formed, and its concentration increases). In other words, the main influences on dB/dt result from the part $a_0(S \cdot B)$, for most of the reaction time, since the exponent has more or less a similar value around ≈ 0.8 (Figure 3.8 (a)) during most of the time, which is in-line with the underlying ground truth equation in (3.16) (no impact of P): Figure 3.8 (b) displays the true change of biomass ($\phi \cdot B$) as a

function of the substrate and biomass concentrations. Considering two specific values of the biomass (i.e., $0.02\,\mathrm{g\,L^{-1}}$ or $0.30\,\mathrm{g\,L^{-1}}$), the growth can be shown as a function of the substrate. In case of low biomass availability (blue dashed line), the growth seems to be less dependent on the substrate, whereas in case more biomass is available (black dotted-dashed line), the substrate concentration shows a greater impact on the growth. In such a case, as expected, as soon as the substrate level drops, a significant decrease in growth rate can be observed (right part of Figure 3.8 (b) for the black dotted-dashed line). The predicted time series profiles by the BMS given in Figure 3.5 show a good accuracy also in the beginning and at the end of the process operation, for which the mentioned significant drop in the growth needs to be captured. The BMS was able to describe such trends without the need of chemical or biological background knowledge. If the growth predicted by the BMS - the right-hand side of equation (3.18) - is visualized (Figure 3.9), a similar trend can be observed, although slight numerical discrepancies are observable compared to the underlying system in Figure 3.8 (b).

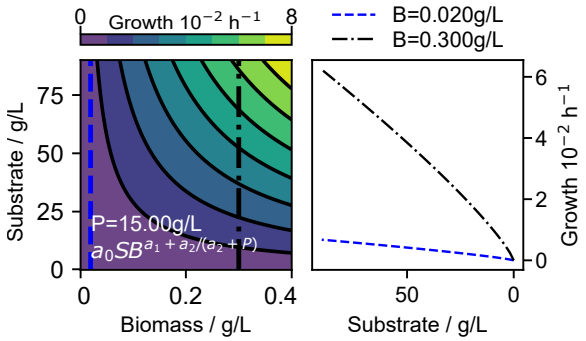

Figure 3.9. The biomass growth identified by the BMS equation is visualized as a function of the substrate concentration S and the biomass concentration B (for the indicated concentration of the product). Similarly to Figure 3.8, two scenarios are highlighted by the blue dashed (constant biomass of $0.02\,\mathrm{g\,L^{-1}}$) and black dotted-dashed lines (constant biomass of $0.30\,\mathrm{g\,L^{-1}}$), for which the growth is shown as a univariate function of the substrate concentration.

A similar analysis can be performed for example for the identified equation of the substrate consumption rate, given in (3.19). The BMS identified an expression where all state variables show an inhibiting influence on the rate of S. In other words, the consumption of the substrate is enhanced by increasing the concentration of the other species in the system. Due to the closed-form availability of the model, a deeper analysis of the rate equation is possible, which is showcased by a

further decomposition of the identified expression into individual terms, namely h_1, h_2, and h_3 given below. Compared to the pure ANN, this poses an advantage since knowledge about a system can be extracted.

$$h_1 = a_2 + a_0 \tag{3.21}$$

$$h_2 = a_1 \cdot P \cdot S^{(1/(S \cdot P))^{a_0}} \tag{3.22}$$

$$h_3 = a_1 \cdot B^{S \cdot a_1} \cdot S^{(1/(S \cdot P))^{a_0}} \tag{3.23}$$

Since h_1 only consists of constants, this term is disregarded for the time being, since no metabolite influences it. Considering the terms h_2 and h_3, it is observable that the constant a_1 and the part $S^{(1/(S \cdot P))^{a_0}}$ is the same for both terms. In case one is interested in the significance of the individual parts, the numerical ratio of the two terms will matter, since both terms, h_2 and h_3 have the same sign and therefore the same impact on the consumption of the substrate. Creating such a ratio $\psi = h_2/h_3 = P/B^{Sa_1}$ will result in the following consumption rate of the substrate (still disregarding h_1):

$$dS/dt \approx -a_1 \cdot P \cdot S^{(1/(S \cdot P))^{a_0}} - a_1 \cdot B^{S \cdot a_1} \cdot S^{(1/(S \cdot P))^{a_0}} \tag{3.24}$$

$$\approx \underbrace{-a_1 \cdot P \cdot S^{(1/(S \cdot P))^{a_0}}}_{h_2} \underbrace{- a_1 \cdot \frac{P}{\psi} \cdot S^{(1/(S \cdot P))^{a_0}}}_{h_3} \tag{3.25}$$

With this, one can observe that if $\psi > 1$, it results in a case where $h_3 < h_2$. On the other hand, if $\psi < 1$, the case $h_3 > h_2$ is obtained. Visualizing the value of ψ for different ranges of the biomass and substrate concentration and a given value of the product concentration (P) in Figure 3.10, one can observe how the terms change their numerical relevance compared to each other (which term has more impact on the substrate concentration). With growing product concentration P, the value of ψ starts to grow as well ($\psi >> 1$), leading to higher contributes by the term h_2.

The obtained closed-form expressions models bring not only the advantage of being able to extract some knowledge on the system's behavior. Due to the algebraic form of the models, another useful benefit is the possibility to calculate the gradients analytically. This opens the opportunity to include these models for example in deterministic optimization algorithms, where the objective functions and constraints need to be available in closed-form manner (Bongartz & Mitsos,

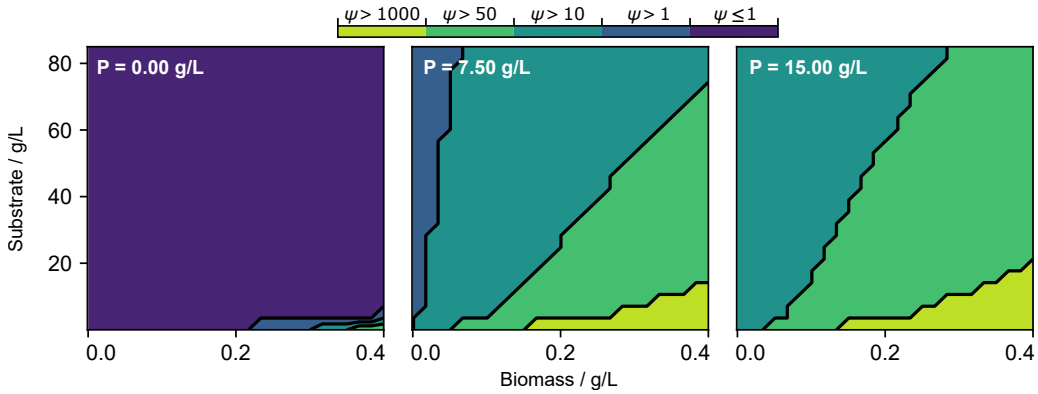

Figure 3.10. The contour plots of the numerical ratio ψ are shown for three different levels of the product concentration P, which are [0, 7.5, 15] $\mathrm{g\,L^{-1}}$. The colors of the contour represent the value of ψ.

2019; Misener & Floudas, 2014; E. M. Smith & Pantelides, 1999; Tawarmalani & Sahinidis, 2002).

Despite the above-discussed advantages the closed-form analytical equations provide, there are also disadvantages, where the high CPU times for the BMS model training is one of the main drawbacks. Considering the averaged CPU times for the BMS training in Table 3.2, the models required at least 35 min (CSII-Poly-20) and at most 156 min (CSII-Poly-40). The exact CPU times are documented in Section B.5 of the supporting information. As discussed in earlier works (Forster, Vázquez, & Guillén-Gosálbez, 2023a; Negri et al., 2022; Vázquez et al., 2022), the BMS in general requires significantly more training time than the benchmark surrogates (i.e., ANN and GP). This is because the latter are based on a fixed canonical formalism and highly efficient algorithms, such as the used Python packages Scikit-learn (Pedregosa et al., 2011). Also, the BMS algorithm was originally designed by the authors to only allow the number of MCMC steps as a stopping criterion (Guimerà et al., 2020). The evolution of the description length, given in expression (3.13), is shown in Figure 3.11 as a function of the number of executed MCMC steps.

To compare the case studies, the description lengths were scaled to a range between zero and one. For CSI, it can be observed in Figure 3.11 (top) that after around 800 MCMC steps, the description length does not significantly change. A similar picture is observed in Figure 3.11 (bottom) for CSII, where the most significant decline in the description length was achieved in the first 2000 MCMC steps.

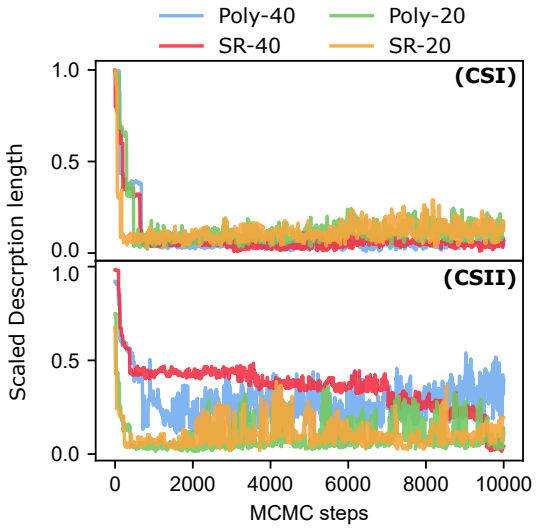

Figure 3.11. The scaled mean description lengths are visualized for each MCMC step for CSI (top) and CSII (bottom). The mean results from averaging the description lengths of the different BMS models obtained for each species of each scenario (Poly-40, SR-40, Poly-20, SR-20).

The only exception can be observed in the scenario CSII-SR-40, where the description length declines gradually. These observations imply that the models identified after those steps perform similarly in terms of training predictions.

3.6 Conclusion

In this work, we investigated the use of machine learning to identify kinetic models of bioprocesses without assuming a pre-defined model structure. A symbolic regression algorithm, the Bayesian machine scientist, was employed to generate suitable models considering their error and level of similarity with a predefined corpus of equations. The model training was performed following a two-step approach, thus avoiding the iterative integration of differential equations, by using two methods to calculate derivatives, i.e., polynomial fitting and univariate symbolic regression. Also, the influence of the sample size was studied. Our approach was applied to two different case studies to showcase its capabilities. Our method performed slightly better than ANNs, while leading to analytical expressions that can be more easily analyzed. However, the BMS leads to higher computational times, which might be reduced in the future as symbolic regression algorithms reach higher maturity levels. Future work should focus on guiding the SR algorithm more efficiently towards equations that are more likely to explain the

data precisely, for example by using tailored standard kinetic equations during the training of the SR algorithm.

Chapter 4

Algebraic surrogate-based process optimization using Bayesian symbolic learning

This chapter is based on the following publications: Forster T., Vazquez D., Guillén-Gosálbez G. (2023a). Algebraic surrogate-based process optimization using Bayesian symbolic learning. *AIChE Journal*, e18110. and Forster T., Vazquez D., Guillén-Gosálbez G. (2023b). Global optimization of symbolic surrogate process models based on Bayesian learning. *Computer Aided Chemical Engineering*, 52, 1241-1246.

Nomenclature for Chapter 4

Sets

E	$\{e \mid e$ is a closed-form algebraic expression$\}$
I	$\{i \mid i$ is a decision variable and represents a process condition$\}$
J	$\{j \mid j$ is a parameter$\}$
M	$\{m \mid m$ is an equality constraint$\}$
N	$\{n \mid n$ is an inequality constraint$\}$
R	$\{r \mid r$ is a reactor, used for case study IV$\}$
S	$\{s \mid s$ is a sample consisting of an input vector w_s and the corresponding output $o_s\}$
$\mathbb{R}$	Real numbers

Parameters

Φ	Specified process condition

Variables

$f(x)$	Target process response depending on input variables x and process specifications Φ
$\underline{f}$ and $\overline{f}$	Lower and upper bound of an observed response f
$\overline{F}(x)$	Surrogate model approximating $f(x)$ only depending on input variables x
g_n	Inequality constraint n in an optimization problem
h_m	Equality constraint m in an optimization problem
o_s	A response/output of a sample s
x_i	Process specification/condition and decision variable i
$\underline{x_i}$ and $\overline{x_i}$	Lower and upper bound of decision variable i
$\tilde{x}/\tilde{f}$ or $\tilde{w_s}/\tilde{o_s}$	Scaled variable/sample or normalized response/output
z_e	Closed-form algebraic expression
$w_{s,i}$	A sample s of a decision variable i

4.1 Introduction

The optimization of process flowsheets is a fundamental problem in Process Systems Engineering, for which several approaches have been proposed to date. Originally, process optimization relied on mechanistic models based on some knowledge of the system (Haydary, 2019). Such models provide a closed-form description that enables the direct application of deterministic optimization algorithms, including global optimization (GO) methods. One possible example of deterministic global flowsheet optimization, the reader is referred to the work of Bongartz and Mitsos (Bongartz & Mitsos, 2019).

Wherever complex systems are studied, a first-principles model is not easy to design. Some authors proposed building tailored data-driven models, which can simplify the optimization task. For example, Sun and Braatz (2020) introduced ALVEN (Algebraic Learning Via Elastic Net) to identify a nonlinear interpretable model for manufacturing data. Another widely known approach is ALAMO (Cozad et al., 2014; Wilson & Sahinidis, 2017), which considers a range of basis functions to build algebraic models for given data. However, these methods constrain the model structure because they rely on given monomials and transformations of the input variables, which can result in less accurate approximations. More recently, data-driven (also referred to as black-box) models emerged to deal with problems in which the underlying phenomena cannot be easily described. They include (but are not limited to) classical regression models such as polynomial regression (Ostertagová, 2012) and state-of-the-art machine learning (ML) algorithms. Black-box models require little physical knowledge about the process (Narayanan, Luna, et al., 2021). Moreover, with the latest ML packages/platforms available, such as scikit-learn (Pedregosa et al., 2011), Tensorflow (Abadi et al., 2015), and PyTorch (Paszke et al., 2019) for python, the Matlab ML toolbox (The MathWorks Inc, 2024), and even low-code/user-friendly alternatives like AutoML (Guyon et al., 2017) and KNIME (Berthold et al., 2007), ML models can be implemented quickly and reliably. However, despite being easy to build, they follow a pre-determined structure (Cozad & Sahinidis, 2018) and may extrapolate poorly. In the context of PSE, artificial neural networks (ANN) and Gaussian processes (GP) were, for example, applied by Del Rio Chanona et al. (2019) to simulate a wastewater biotreatment and by Gnoth et al. (2010) in a bioprocess simulation.

Data-driven approaches have also been applied to build surrogates of mechanistic models that are hard to optimize. For example, Jones et al. (1998) introduced a response surface methodology for expensive multimodal functions, where they applied Bayesian optimization. Following a similar approach, Quirante et al. (2015) optimized distillation columns with surrogate models based on Kriging interpolation. Later works (Quirante & Caballero, 2016; Quirante et al., 2018) extended this methodology to replace other units with Kriging models (GP regression), leading to a hybrid simulation-optimization modeling framework.

Optimizing standard data-driven ML models is not straightforward due to their intrinsic complexity and nonlinearities (Mitsos et al., 2009; Schweidtmann & Mitsos, 2019). Here, the standard approach is to solve these models to local optimality. However, more recently, tailored deterministic GO algorithms for data-driven models emerged. Notably, Schweidtmann and Mitsos (2019) introduced a global optimization approach named MAiNGO (McCormick-based Algorithm for mixed-integer Nonlinear global optimization) to optimize ANNs globally, which was later extended to use GP models (Schweidtmann et al., 2021). It built on earlier works that integrated machine learning models in optimization problems by applying the created toolbox named MeLOn (Schweidtmann et al., 2020) (Machine Learning models for Optimization). Ceccon et al. (2022) presented OMLT, an optimization and machine learning toolkit to optimize trained ANNs or gradient-boosted trees. Boukouvala et al. (2017) introduced a methodology for the global optimization of constrained grey-box problems using Kriging models and derivative-free global optimization and applied it to pressure swing adsorption. Later, Boukouvala and Floudas (2017) presented the algorithmic framework ARGONAUT to globally optimize general constrained grey-box problems using the ANTIGONE solver. A parallel version named p-ARGONAUT was subsequently published by Beykal et al.(2018). In a recent work, Paulson and Lu (2022) proposed the COBALT (constrained Bayesian optimization of computationally expensive grey-box models exploiting derivative information) algorithm for constrained grey-box optimization problems, combining GP models with state-of-the-art optimizers (J. Paulson & Lu, 2021).

It is important to note that the surrogates used in process optimization are approximations of the original systems (e.g., mechanistic models) but are not rigorous relaxations of the original model. In other words, they do not necessarily provide rigorous bounds on the optimal solution of the original model. Hence, even if

the surrogate is globally optimized, there is no guarantee that its global optimum will be, in turn, the global optimum of the original model. Notwithstanding this important observation, it is appealing to identify the global optimum of the surrogate as, if the surrogate is accurate enough, this would likely lie close to the global optimum of the original model. In recent years, data-driven modeling based on symbolic regression (SR) has been attracting growing interest. Often referred to as genetic programming (Cozad & Sahinidis, 2018; Keane et al., 1993; Koza, 1994), the goal here is to find closed-form mathematical expressions based on expression trees using mainly evolutionary algorithms (Diveev & Shmalko, 2021), although more recently deterministic MINLP methods were also applied (Cozad & Sahinidis, 2018). One of the advantages of this regression method is that it does not necessarily assume a pre-determined model structure (e.g., in a multivariate regression) or a set of alternative model structures (e.g., for the ALAMO approach). SR was successfully applied in many different fields. For example, Tsionas and Assaf (2020) applied SR to tourism research. McKay et al. (1997) used an SR approach to model a vacuum distillation column and a chemical reactor system. In a later work, McKay et al. (1999) applied SR to develop a model of a food extrusion process. In the control area, Keane et al. (1993) proposed an approach to approximate an impulse response for a linear time-invariant system. In a more recent work by Schmidt and Lipson (2009), the authors discovered physical laws from experimental data using SR to identify nonlinear relationships. In the cosmology field, Cranmer et al. (2020) applied SR to components of a trained graph neural network to extract explicit physical relations.

The advantage of using algebraic surrogates is that, besides improving interpretability, they enable the use of deterministic Global Optimization (GO) algorithms (Androulakis et al., 1995; I. E. Grossmann, 1996; Ryoo & Sahinidis, 1995; E. M. B. Smith & Pantelides, 1997; Tawarmalani & Sahinidis, 2002; Zamora & Grossmann, 1999). Stochastic GO methods are mainly applied when the objective function is unknown, meaning the algebraic expression of the function and its corresponding derivatives are not available, which prevents the use of deterministic methods (Bradford et al., 2018). Such algorithms may find the global optimum; however, they need an infinite running time to guarantee global optimality (Ryoo & Sahinidis, 1995). Simulated annealing (Hwang, 1988), genetic algorithms (Holland, 1992), or particle swarm algorithms (Kennedy & Eberhart, 2006) are the most widespread stochastic GO methods. On the other hand, deterministic GO

methods are guaranteed to identify the global solution - within a given ϵ-tolerance - in a finite number of iterations (Androulakis et al., 1995; Horst & Tuy, 1996; Schweidtmann et al., 2019, 2021). Typical deterministic GO methods are based on spatial-branch-and-bound, while other approaches (Kesavan et al., 2004) applied alternative schemes inspired by the original outer approximation algorithm by Duran and Grossmann (Duran & Grossmann, 1986b). For example, Bergamini et al. (2005) presented a logic-based outer-approximation algorithm for MINLPs showing some nonconvexities. Here, the master problem incorporates piece-wise linear approximations of the nonlinear terms, so it yields a lower bound (when minimizing). The algorithm is guaranteed to identify the global optimum within a given tolerance for a sufficient enough number of iterations. State-of-the-art solvers implementing such strategies include BARON (Sahinidis, 1996; Tawarmalani & Sahinidis, 2005) and ANTIGONE (Misener & Floudas, 2014), which mainly differ in the bound tightening (Belotti et al., 2009; Puranik & Sahinidis, 2017) and relaxation methods implemented (Locatelli & Schoen, 2013; Misener & Floudas, 2014; Tawarmalani & Sahinidis, 2002). Despite providing an optimality gap within which the global optimum should fall, they require explicit, closed-form mathematical expressions to be available (Bongartz et al., 2020). Therefore, in the context of process optimization, such deterministic GO methods could be applied if equation-oriented flowsheet models are at hand (Bongartz & Mitsos, 2019).

This work explores the use of SR coupled with state-of-the-art GO algorithms in the context of process optimization. The most important reasons for this two-stage approach are that the user does not have to make *a priori* assumptions about the mathematical structure of the model and that, subsequently, well-established of-the-shelf deterministic GO algorithms can be applied once the algebraic surrogate is at hand. Focusing on an SR approach presented by Guimerà et al. (2020), called the Bayesian machine scientist (BMS), we first derive algebraic regression models of the desired processes. The BMS was already applied to some case studies. The work by Negri et al. (2022) applied this regression method to approximate process simulations. Vazquez et al. (2022) used it to model the link between energy-related impacts and socioeconomic drivers. However, up to now, the resulting equations were never used to simplify the global optimization of those systems, which will be explored in the present work. After identifying a suitably well-fitting model equation, available global deterministic solvers are

used to search for the global optimum of the algebraic surrogate. We show the advantages of this approach in several case studies covering unit operations and full flowsheets. To our knowledge, this is the first work that applies SR to the global optimization of process flowsheets, opening new avenues for the application of algebraic surrogates in process optimization.

The remainder of the paper is organized as follows: First, the problem statement is detailed, followed by the methodology. Afterward, the case studies are introduced, and the results are discussed. Finally, the conclusions of the work are drawn.

4.2 Problem statement

Here, without loss of generality, we shall consider an existing process operated in a steady state. Known process parameters, for example, equipment size and flow rates, are described by $\phi_j, j \in J$, where J is the set of known properties. The decision variables are denoted by x_i, where $i \in I$ refers to the set of variables to be optimized. Therefore, there are $|I|$ degrees of freedom to be varied to optimize a user-defined target objective $f(x, \phi)$ (either minimized or maximized). For the sake of simplicity, the known parameters ϕ are subsequently skipped from the notation, leaving the process described by the degrees of freedom, $f(x)$. Additionally, unless otherwise indicated, in what follows, we will focus on minimization problems. A schematic representation with an example is given in Figure 4.1.

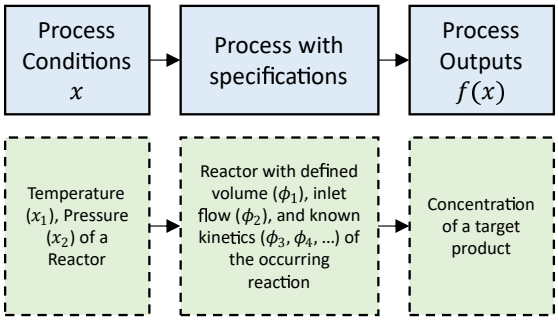

Figure 4.1. In the top row (blue, solid squares), a schematic representation of a considered process is given with its degrees of freedom x, fully specified process conditions and equipment properties ϕ and a target objective $f(x)$ that is required to be optimized. In the bottom row, an example is given in the green dashed squares.

The goal of this work is two-fold: First, we wish to find a suitable expression

$F(x)$ that maps the inputs x sufficiently well to the target objective and therefore approximates the response $f(x)$ accurately. This mapping is described by the closed-form surrogate expression $F(x)$, which can be generated by various methods discussed in the introduction. Herein, however, it should be identified without assuming a pre-defined model structure, which is achieved by relying on flexible symbolic trees that include prior knowledge:

$$\begin{aligned} & F(x) \approx f(x) \\ & \text{s.t.} \underline{x} \leq x \leq \overline{x} \\ & x \in \mathbb{R} \end{aligned} \tag{4.1}$$

With $\mathbb{R}^{|I|}$ being the domain of input variables. After being able to approximate the studied process $f(x)$ with an appropriate surrogate $F(x)$, the main goal of this work is to globally optimize the surrogate $F(x)$ to find the values x^*. In mathematical terms, such an optimization task can be formulated as a nonlinear programming problem (NLP), as described in equation (4.2):

$$\begin{aligned} f^* = & \min_x f(x) \\ & \text{s.t.} g_n(x) \leq 0, \quad \forall n \in N \\ & h_m(x) = 0, \quad \forall m \in M \\ & \underline{x} \leq x \leq \overline{x} \\ & x \in \mathbb{R}^{|I|} \end{aligned} \tag{4.2}$$

Where x represents the aforementioned process conditions. Available inequality constraints $g_n(x)$ and equality constraints $h_m(x)$ are also considered. These constraints represent, for instance, allowable operating ranges for equipment units, such as maximum allowed flow rates, or desired specifications, such as maximum permitted emissions into the environment. The lower and upper bounds of the decision variables are denoted by $\underline{x}$ and $\overline{x}$, respectively. For example, we may be interested in defining upper bounds on the temperatures and pressures or quality specifications on product purity. The global optimization of chemical processes is challenging due to the highly nonlinear and non-convex nature of the mechanistic equations describing their behavior (e.g., Antoine equation, Arrhenius expression, etc.). Moreover, commercial simulation software suffers from numerical noise and

convergence issues. At the same time, monolithic formulations implemented in algebraic modeling systems are hard to initialize, and the implementation of the equations themselves, e.g., mass and energy balances and thermodynamic equations, might not be trivial. A possible way to simplify the optimization is to resort to surrogate models $F(x)$ in which the original problem is approximated as follows:

$$\begin{aligned} F^* = & \min_x F(x) \\ & \text{s.t.} g_n(x) \leq 0, \quad \forall n \in N \\ & h_m(x) = 0, \quad \forall m \in M \\ & \underline{x} \leq x \leq \overline{x} \\ & x \in \mathbb{R}^{|I|} \end{aligned} \tag{4.3}$$

Where we assume that model F is given in an algebraic form. Problem (4.3) could then be solved for global optimality using state-of-the-art GO solvers. Hence, we aim to construct an accurate representation of $F(\cdot)$ and globally optimize the resulting problem.

4.3 Methodology

4.3.1 Modeling framework

We propose an approach to tackle the problem above that follows two main steps. First, we build an algebraic surrogate model $F(x)$ that approximates the objective function computed from the detailed model (i.e., real process) $f(x)$ sufficiently well. To this end, we use symbolic regression tools that build algebraic expressions from data without assuming any aprioristic model structure. Second, we solve the surrogate model-based optimization problem using standard global optimization packages.

Step 1: Data generation using process simulation
A schematic overview of the data generation process is given in Figure 4.2. In order to gather the data required for the surrogate model generation, a flowsheet of the desired case study is modeled using Aspen HYSYS. An objective function $f(x)$ is defined, which is computed from the values of the dependent variables,

i.e., degrees of freedom, defined in the process model. To map inputs to outputs, we define the input variable $w_{s,i}$, where $s \in S$ refers to a given sample, and $i \in I$ refers to a degree of freedom, i.e., a feature of the surrogate model. The output vector is denoted as $f(w_s)$, or f_s in short. Therefore, the sampling matrix is generated with the desired number of samples $|S|$ using the Latin hypercube sampling (LHS).

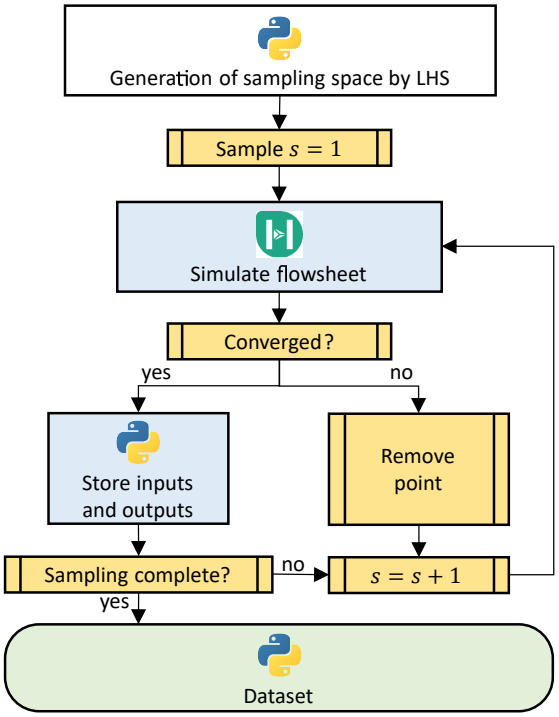

Figure 4.2. Schematic representation of the data collection procedure. Starting from a sampling space defined by a Latin hypercube sampling approach, the vectors $w_{s,i}$ will be sent to HYSYS. If the flowsheet does converge, the sample (consisting of the input vector w_s and the sampled output $f(w_s)$) is stored. If the flowsheet does not converge, the complete sample is removed, and the next input vector is sent to HYSYS. As soon as the sampling is completed, the algorithm is terminated. It is worth to be mentioned that the final dataset only consists of those samples where the flowsheet converged.

Step 2: Data pre-processing

Appropriate data treatment is required before training the model with the raw data. Notably, data scaling and selection help improve the model's accuracy and robustness (Arora, 2012; Elble & Sahinidis, 2012; A. C. Müller & Guido, 2017). A schematic representation of the data treatment strategy is given in Figure 4.3. We included a data selection step, where only samples that fall between the limits of a user-defined range $\underline{f} \leq f(w_s) \leq \overline{f}$ were considered. The motivation for this is that

the optimal solution might not lie close to samples showing poor performance in terms of objective function values. After selecting the data for the model training, our proposed framework includes a feature scaling step. Specifically, the values of the input vector $w_{s,i}$ of each sample s are scaled to a range of user-defined values, describing the resulting scaled vector as $\tilde{w}_{s,i}$. Finally, the response value $f(w_s)$ of each input vector is normalized within another user-defined range, where the resulting response is described by $\tilde{f}(w_s)$. After having preprocessed the data, the existing values are updated (to simplify notation):

$$w_{s,i} \leftarrow \tilde{w}_{s,i} \quad \text{and} \quad f(w_s) \leftarrow \tilde{f}(w_s), \qquad \forall s \in S, i \in I \tag{4.4}$$

The scaling and standardization steps support the model training: The maximum likelihood estimator of the model parameters is obtained by solving a least-squares problem (Guimerà et al., 2020). This can require an iterative procedure that relies on the calculation of the Jacobian (Fletcher, 2000; Nocedal & Wright, 2006). Scaling the data can bring the numerical values of the derivatives in the same order of magnitude, simplifying the model training (Nocedal & Wright, 2006). All three pretreatment steps (selection, scaling, normalization) do not need to be performed concurrently.

Subsequent to the above-mentioned data treatment, the resulting dataset S is split into two proper subsets:

$$S^{TR} \subset S \quad \text{and} \quad S^{TE} \subset S \tag{4.5}$$

where S^{TR} and S^{TE} represent the training and test subsets, respectively. The training subset is later used for model training, whereas the test subset is used for model testing.

Step 3: Surrogate model generation

The goal here is to find an appropriate surrogate expression $F(x)$ that maps the input data x well to the corresponding objectives $f(x)$. This work aims to show the possibility of constructing a closed-form surrogate without assuming a predefined mode structure. As mentioned previously, we apply SR to perform this task, which represents mathematical expressions by a symbolic tree, as shown in

Figure 4.4 (a).

The space of possible expressions is described by E. Starting from one symbolic representation $z_e, e \in E$ one can perform changes in the tree that lead to a different mathematical expression. Such tree evolutions include but are not limited to node replacement (i.e., changing the addition operator in Figure 4.4 by a division operator) and elementary tree replacement (i.e., exchanging the complete sub-tree $(\alpha - \beta)$ by another tree). For each resulting expression, a goodness-of-fit metric can be calculated. The SR algorithm then searches for an expression leading to the best goodness-of-fit metric, akin to other evolutionary algorithms. In this work, we use the SR algorithm developed by Guimerà et al. (2020), the BMS, to simplify the optimization of process flowsheets. The BMS provides a closed-form algebraic expression from data based on a set of user-defined mathematical operations (i.e.,

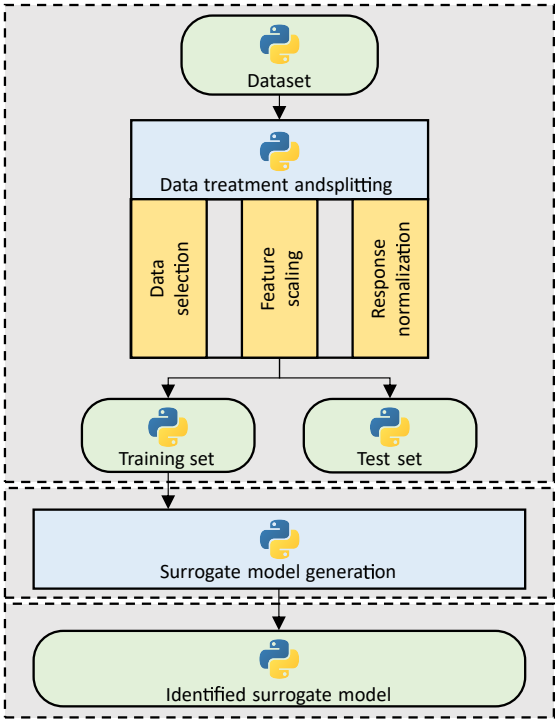

Figure 4.3. Schematic representation of the data treatment and surrogate model generation procedure. In the first step, the user might choose to pretreat the data or not: Data selection will reduce the number of samples. Feature scaling brings the inputs of the sample matrix (e.g., the vectors $w_{s,i}$) to the same range of numerical values. Response normalization will bring the response values (e.g., the values of w_s) to the same range of numerical values. After the pretreatment, the data is split into a training and a test set, where the training set is used to train the surrogate model. The result of this framework is an identified surrogate model in a closed-form algebraic expression.

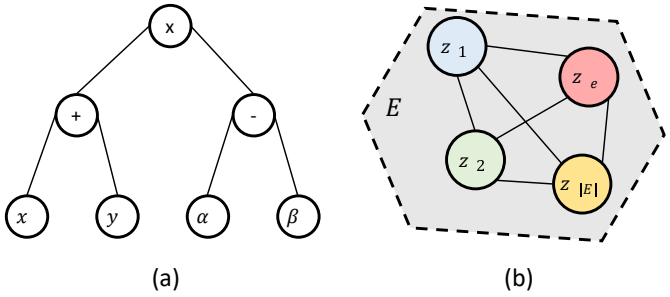

Figure 4.4. (a) Representation of the mathematical expression $z(x, y) = (x + y)(\alpha - \beta)$ as a symbolic tree. (b) The space E of all possible expressions z_e is schematically shown as a dashed polygon.

addition, subtraction, multiplication, etc.). We next provide an overview of how the BMS works. For further information, the reader is referred to the original paper (Guimerà et al., 2020). In the algorithm, a conditional probability $p(z_e|D)$ for each expression $z_e, e \in E$ (the space of symbolic trees) to fit some data D is calculated according to Bayes Theorem (Bishop, 2006; Murphy, 2013):

$$p(z_e|D) = \frac{p(D|z_e)p(z_e)}{p(D)} \tag{4.6}$$

In this expression (4.6), D represents the observed data, and $p(D)$ is the marginal likelihood of the data (independent of z_e and therefore acting only as a normalization constant). Using marginalization over the parameters θ_e associated with expression z_e (Murphy, 2013), the numerator in equation (4.6) can be expressed as an integral over the space of all possible parameter values θ_e (Guimerà et al., 2020). This resulting integral can be described by the so called description length $L(z_e)$ (Guimerà et al., 2020; Hansen & Yu, 2001; Murphy, 2013):

$$\begin{aligned} L(z_e) &= -\log\left[p(D|z_e)p(z_e)\right] \\ &= -\log\left[\int_{\Theta_e} p(D|z_e, \theta_e)p(\theta_e|z_e)d\theta\right] \end{aligned} \tag{4.7}$$

Computing precisely this integral is challenging (Guimerà et al., 2020; Murphy, 2013). However, under certain assumptions (Grünwald, 2007; Murphy, 2013), it can be approximated by the Bayesian information criterion (BIC) and the prior of the corresponding expression z_e:

$$L(z_e) \approx \frac{BIC(z_e)}{2} - \log\left(p(z_e)\right) \tag{4.8}$$

Therefore, the plausibility of an expression z_e conditioned on some data D is obtained by the description length $L(z_e)$. According to Grünwald (2007), $L(z_e)$ can be understood to describe the corresponding mathematical expression in terms of 'number of natural units' (encoded length of the expression). The BMS uses an MCMC (Hastings, 1970) algorithm to explore E, defined as the space of possible closed-form expressions. To this end, the user can define how many MCMC iterations the BMS should perform. After evaluating the description length of each expression, the BMS uses the most plausible one, representing the expression with the shortest description length (the best goodness-of-fit).

Step 4: Optimization and validation

After identifying an appropriate surrogate model in the form of an algebraic expression $F(x)$, the model shown in expression (4.3) is optimized to find the global optimum F^*. A schematic representation of this model-based optimization procedure is given in Figure 4.5, where the simplest form of such an optimization can be formulated as a box-constrained problem (with lower and upper bounds, $\underline{x}$ and $\overline{x}$, on the variables):

$$\begin{aligned} F^* &= \min_x F(x) \\ \underline{x} &\leq x \leq \overline{x} \\ x &\in \mathbb{R}^{|I|} \end{aligned}$$

If required, constraints can be added to this optimization problem – which might include technical constraints on process units or simple mass balances – leading to the formulation in equation (4.3). If challenging constraints are needed, they could also be replaced with surrogate models embedded in the formulation. In this work, the optimization is carried out using available state-of-the-art GO solvers to identify the model-based global optimum F^* within a user-defined ϵ-tolerance. The obtained solution x^* is then sent to the rigorous mechanistic model to re-assess the objective function (second block in Figure 4.5). From the rigorous simulation outcome, the observed process response $f^* = f(x^*)$ can be extracted. Using this observed response $f(x^*)$ and the model-based optimum $F(x^*)$, a defined error metric can be calculated, as discussed in Section 4.3.2. This metric indicates how

well the surrogate model $F(x)$ is able to approximate the real process $f(x)$ in the optimal solution found.

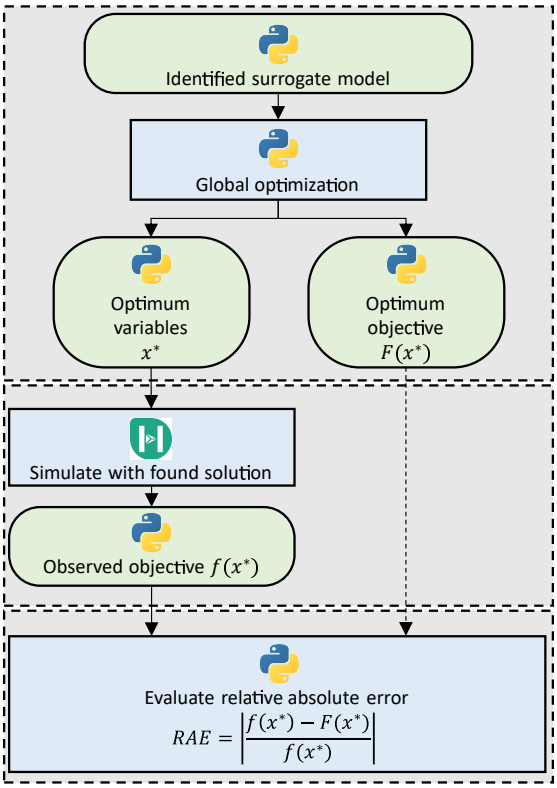

Figure 4.5. Schematic representation of the model-based optimization and model validation procedure. By using the identified surrogate model, an optimization problem is formulated (might include constraints) and solved to global optimality. The identified minimizer x^* is further sent to the flowsheet in HYSYS, where the observed response $f(x^*)$ is extracted. Lastly, the mismatch between the model-based optimum $F(x^*)$ and the observed response $f(x^*)$ is calculated as a relative absolute error.

4.3.2 Performance and comparison metrics

Training and testing errors are obtained for both the training and test subsets S^{TR} and S^{TE} using the root mean squared error ($RMSE$), mean absolute error (MAE), and coefficient of determination (R^2), as follows:

$$
\begin{aligned}
RMSE &= \sqrt{\frac{1}{n}\sum_{s\in S}\left(f(w_s)-F(w_s)\right)^2} \\
MAE &= \frac{1}{n}\sum_{s\in S}|f(w_s)-F(w_s)| \\
R^2 &= 1 - SSR/SST = 1 - \sum_{s\in S}\frac{\sum_{s\in S}(F(w_s)-f(w_s))^2}{\sum_{s\in S}(f(w_s)-\mu_f)^2}
\end{aligned}
\tag{4.9}
$$

In the relationships in equation (4.9), the predictions by the model are described by $F(w_s)$ using the given input vector w_s of one sample s. The observed process response and the mean of the observed process responses are described by $f(w_s)$ and μ_f, respectively. As already mentioned, both, the model predictions $F(w_s)$ and the observed response $f(w_s)$ are calculated by using input data from the training or test set. Variables SSR and SST denote the sum of squares of residuals and the total sum of squares (proportional to the variance of the data), respectively. Similarly, the quality of the surrogate is measured by the relative absolute error (RAE), given by (4.10), which quantifies the mismatch between the surrogate and the mechanistic model outcomes for a given solution x.

$$
RAE = \left|\frac{f(x)-F(x)}{f(x)}\right| \tag{4.10}
$$

In addition to these error metrics, the time required for both the model training and the model-based optimization is reported as central processing unit (CPU) time. Lastly, both the solver and model status are reported.

4.3.3 Benchmarking and implementation details

We compare our results to those obtained using the MAiNGO algorithm (Bongartz et al., 2020; Schweidtmann & Mitsos, 2019), which was also applied to the global optimization of process flowsheets. We want to highlight that the intention is not to claim the superiority of one methodology over the other but to have a proven and rigorous benchmark for our approach. We train a GP using the same training data and optimize it using MAiNGO. For a detailed description of how MAiNGO operates, the reader is referred to the outstanding works by these authors. We reference the implementation details from Schweidtmann and Mitsos (2019): MAiNGO represents a branch-and-bound optimization solver that is

implemented in C++. The convex relaxations of the constraints and the objective are linearized with the use of sub-gradients. The resulting linear program is solved by CPLEX. For upper bounding, the problem is locally optimized using the SLSQP algorithm. All calculations were carried out on an Intel®Core™i7-8700 CPU and 16 GB of RAM. The software and corresponding versions are provided next. To construct the sampling dataset, we used Python 3.8.11 with NumPy v1.21.2 and pyDOE v0.3.8. The process flowsheet was simulated using Aspen HYSYS v11. Python and HYSYS were linked through the COM interface. The algorithm provided by Guimerà et al. (2020) was used to train the BMS, whereas the GP training was performed using GPyTorch v1.6.0. The symbolic equation generated by the BMS was globally optimized using the General Algebraic Modeling System (GAMS) (GAMS Development Corporation, 2022) v40.2.0 interfacing with the BARON v22.7.23 and ANTIGONE (where explicitly mentioned) v41.3.0 solver. The trained GP was optimized using MAiNGO v0.5.0.

4.4 Case studies

4.4.1 Flowsheets for data generation

As described next, we solve several case studies (CS) of increasing complexity regarding the number of degrees of freedom. For all CSs, a Latin hypercube sampling (LHS) design method was used to sample the input vectors w_s, considering 200 (CSI) or 1000 (CSII-IV) samples, for which 20% were used for the test set. The time to collect these samples is reported in the supplementary information Section C.1. It is worth mentioning that the same dataset was used to train different models (for each CS). The indicated bounds were used to set up the input vectors (Table 4.1). These input vectors were sent from Python to HYSYS via the COM interface. The observed responses $f(w_s)$ were then retrieved from Aspen-HYSYS. Below, each CS is briefly discussed, where additional process information/parameters are given in the supporting material.

CSI – Compressor plant The first case study is a compressor plant modeled in Aspen-HYSYS, represented in Figure 4.6, as introduced by Schweidtmann and Mitsos (2019). A pre-defined feed is split into two individual compressors. The split ratio towards compressor one is denoted by b, resulting in the split ratio towards compressor two taking a value $1 - b$. Each compressor is modeled with different efficiency curves, where the efficiency varies depending on the inlet flow.

Table 4.1. The lower and upper bounds for the variables are shown for each case study. These values were used in order to set up the Latin hypercube sampling matrix. Additionally, they were applied as decision variable bounds for the subsequent optimization. The order of the values $\underline{x}_i$ and $\overline{x}_i$ is given below for each individual case study.

CS	Lower bound $\underline{x}$	Upper bound $\overline{x}$
I	0.57 [−]	0.765 [−]
II	100 °C, 160 bar	400 °C, 230 bar
III	180 °C, 4500 kmol h^{-1}, 0.001 [−], 1.25 [−], 35 m^3, 4500 kPa	240 °C, 6500 kmol h^{-1}, 0.05 [−], 1.8 [−], 55 m^3, 5500 kPa
IV	0 [−], 0 [−], 230 °C, 230 °C, 230 °C, 160 bar , 160 bar, 160 bar	1 [−], 1 [−], 400 °C, 400 °C, 400 °C, 230 bar , 230 bar, 230 bar

After the compression, the outlets of the two compressors are mixed and sent to the final outlet. Details about the fixed process parameters (i.e., feed flow rate, feed composition, compressor curves, compressor ratio, etc.) are given in Chapter C.2. There is only one degree of freedom, i.e., the split fraction, so the input vector reduces to a scalar defined as $x = b$. The goal is to minimize the total duty required to operate the plant, where the measured response is consequently described by $f(x) = Q(x)$.

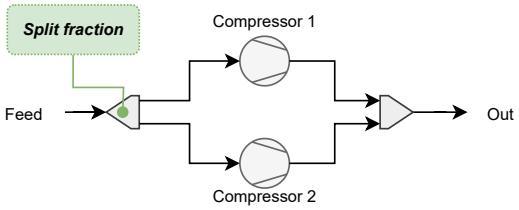

Figure 4.6. Process flowsheet of the compressor plant under study (case study I). The decision variable, $x = b$, for the optimization problem is indicated in the green box.

CSII – Ammonia reactor The second CS represents a chemical reactor, modeled as a multitubular plug flow reactor implemented in Aspen-HYSYS (Figure 4.7). The inlet mass flow is set to a specific value. The temperature T and pressure P dictate the reaction rate. The reactor is operated under adiabatic conditions with a pressure difference of 1 bar. The reactor properties (i.e., the volume, the number of tubes, etc.) are given in the supplementary information Section C.2. In this case study, the degrees of freedom, and therefore the decision variables in the optimization problem, are the temperature and pressure of the feed: $x = [T, P]$. The goal is to maximize the outlet conversion X_{N_2} of nitrogen

to ammonia at the reactor outlet $f(x) = X_{N_2}(x)$.

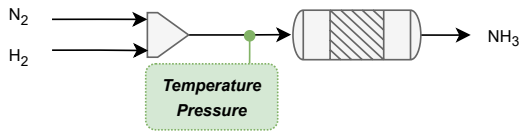

Figure 4.7. Process flowsheet of the ammonia reactor under study (case study II). The decision variables for the optimization problem are indicated in the green box, $x = [T, P]$.

CSIII – Methanol plant The third numerical example optimizes a methanol (MeOH) plant modeled in Aspen-HYSYS (Figure 4.8). Carbon dioxide and hydrogen are fed to the system and adjusted to the required pressure and temperature before being sent to a multitubular plug flow reactor. The reactor outlet goes through two flash drums and into a distillation column. Methanol is collected in the distillate, while water is the main product at the bottom. The vapor streams of the flash drums are sent to a recycle stream (from the first flash) and to a purge (from both drums). The data used in the calculations is provided in the supporting information Section C.2. The degrees of freedom to be optimized are highlighted in green text in Figure 4.8: Reaction temperature, reaction pressure, purge ratio of the splitter, reactor volume, hydrogen flow rate, and reflux ratio of the distillation column. Therefore, the input vector can be described by: $x = [T, P, \eta, V, F, \zeta]$. The objective of this CS is to minimize the unitary cost (UC) of methanol (x) = $UC(x)$, considering a fixed CO_2 flow. The exact calculation of the UC is provided in the supporting information Section C.2. We note that for each sampling point for x, the MINLP model developed by Yee and Grossmann (1990) (SYNHEAT) is solved to obtain the optimum heat exchanger network (HEN) and, therefore, the optimum cost of the HEN.

CSIV – Ammonia reactor series The next case study considers a series of multitubular ammonia reactors similar to the one defined in CS1 (Figure 4.9). A pre-defined feed is sent to a splitter and divided into three different streams. The split fractions b_1 and b_2 (splits towards reactors 1 and 2) are the degrees of freedom for the optimization problem. The third split fraction b_3 is calculated from the mass balance $1 = b_1 + b_2 + b_3$ and is therefore specified. Additionally, the temperatures (T_r) and pressure (P_r) for the different reactors $r \in R = \{1, 2, 3\}$ are also degrees of freedom. The rest of the reactor properties (i.e., the volume, the number of tubes, etc.) are fully defined and displayed in the supporting

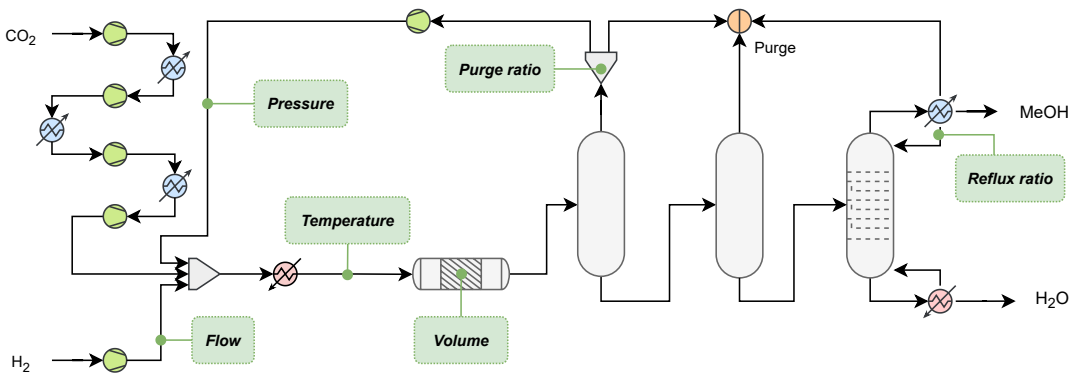

Figure 4.8. Process flowsheet of the MeOH plant under study (case study III) adapted from Vázquez et al. (2021). The decision variables for the optimization problem are indicated in the green boxes, $x = [T, P, \eta, V, F, \zeta]$.

information Section C.2. Therefore, the decision variables for the optimization process are the split fractions toward reactors 1 and 2, and the temperature and pressures of the reactors: $x = [b_1, b_2, T_1, T_2, T_3, P_1, P_2, P_3]$. The objective of this CS is to maximize the overall conversion of nitrogen to ammonia X_{N_2}, described by $f(x) = X(x)$, where X denotes the overall conversion of the process. A simple constraint is added to ensure the splits fulfill the mass balance.

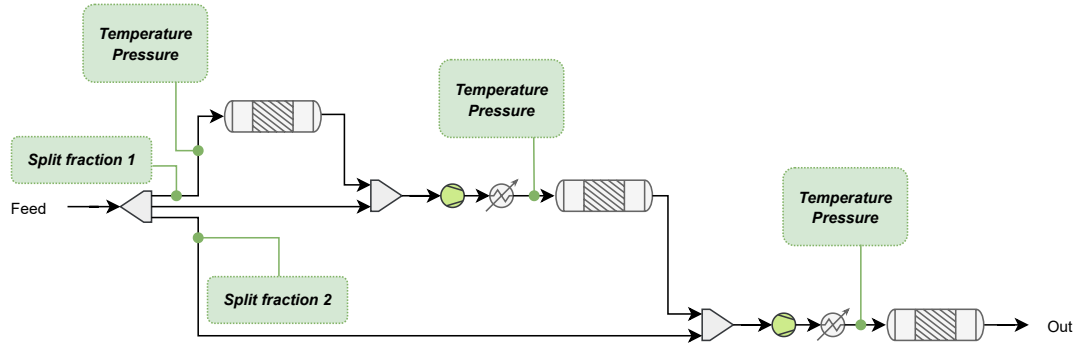

Figure 4.9. Process flowsheet of the ammonia reactor series under study (case study IV). The decision variables for the optimization problem are indicated in the green boxes, $x = [b_1, b_2, T_1, T_2, T_3, P_1, P_2, P_3]$.

4.4.2 Default properties of the training and optimization algorithm

We trained the BMS and the GP considering the parameters and settings shown in Table 4.2. The BMS parameter values are those given in the original article (Guimerà et al., 2020). Regarding the optimization, we applied BARON v20.4.14.

The solver settings and conditions to optimize the trained BMS models are stated in Table 4.2. In all four cases, the starting point for the decision variables was set to the midpoint between the respective upper and lower bounds (4.1). All other settings were set to the default values of the indicated solver version. For optimizing the GP with MAiNGO, in CSII, III, and IV, the option of a pure multi-start (with a maximum of 20 local searches) was additionally activated in case the optimality gap could not be closed. This option will not guarantee global optimality since the pure random multi-start is performed with individual local optimizers. The starting points for the decision variables were chosen to be the same as for BARON. All other settings were set to the default values of the indicated solver version provided by the developers of MAiNGO.

Table 4.2. Settings and hyperparameters for the training and optimization of the Bayesian machine scientist and the Gaussian process.

Model	**Settings**	**CSI**	**CSII**	**CSIII**	**CSIV**
BMS training	MCMC iterations	$18{\cdot}10^3$	$60{\cdot}10^3$	$20{\cdot}10^3$	$40{\cdot}10^3$
	Allowed operations [a]	$\exp(x), \log(x), x^2, x^3, \sqrt{x}, +, -, \div, \times, x^a, \lvert x\rvert$			
	Root replacement probability	0.05			
	Node replacement probability	0.45			
	Elementary tree replacement probability	0.5			
BARON settings	Relative optimality gap	10^{-9}	0	0	0
	Node limits	10^6	10^6	10^8	10^6
	Maximum wall clock time (reslim)	$6{\cdot}10^2$	$6{\cdot}10^2$	$4{\cdot}10^4$	$4{\cdot}10^4$
GP training	Training epochs	$2{\cdot}10^2$	$2{\cdot}10^3$	$1{\cdot}10^3$	$1.5{\cdot}10^3$
	Mean function	Constant			
	Kernel	Matern with n=5/2			
	Likelihood	Gaussian			
	Optimizer	Adam			
	Learning rate	0.1			
MAiNGO settings	ϵ_R [b]	$\epsilon_R = [10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}]$			
	Max CPU [b]	16 500 s	5600 s	41 000 s	123 500 s
	ϵ_R [c]	$\epsilon_R = 10^{-3}$			
	Max CPU [c]	3600 s			

[a] The absolute function $|x|$ was only applied to CSI-III.
[b] Relative optimality gaps and maximum allowed CPU times for inactive multi-start runs.
[c] Relative optimality gaps and maximum allowed CPU times for multi-start runs.

4.5 Results

When discussing the results, we first shall focus on the model training and subsequently on the model-based optimization performance. As mentioned in Section 4.3.3 (benchmarking), the goal of this work was to show that it is possible to model precisely and optimize process flowsheets effectively using symbolic regression. We are not claiming that our approach is consistently superior but that it is competitive and might become even more efficient in the future with further developments in symbolic regression. This field is currently evolving very rapidly. We show that symbolic regression not relying on basis functions but rather on flexible symbolic trees incorporating prior knowledge can precisely model process flowsheets effectively. Although the BMS was previously applied in other works (Negri et al., 2022; Vázquez et al., 2022), to the best of our knowledge, Bayesian symbolic regression was never used to facilitate the global optimization of process flowsheets. In this section, after comparing the performance of the proposed approach with the GP being optimized by MAiNGO, we briefly describe how the same workflow can be followed using a linear basis function (LBF) model in the optimization part in Section 4.5.2. However, in this work, we will not go into a detailed analysis of the LBF approach but rather describe the advantages and disadvantages of the method.

4.5.1 Model training

The model training and testing results are given in Table 4.3. A graphical representation of the results from Table 4.3 is depicted in Figure 4.10, which shows the observed versus predicted (OVP) values of both modeling approaches.

In general, both trained models can explain the variance in the data sufficiently well when considering $R^2 > 0.85$ as acceptance criteria. This acceptance level of $R^2 > 0.85$ was chosen by the authors based on experience. It could be further fine-tuned, which was, however, not the goal of this work and is therefore left for future work. This is the case for all case studies (Table 4.3). Additionally, the models are considered to perform similarly (compared to each other) if the relative difference in R^2 is below 5%, which is confirmed in all CSs (0.4% for CSI and II, 0.6% for CSIII, and 1.9% for CSIV). Throughout the CSs, it can be observed that the ability to explain the training data variance decreases with increasing dimensionality of the problems. The BMS was run using the number

Table 4.3. The training performance criteria are summarized for the Bayesian machine scientist (BMS) and the Gaussian process (GP). Each row represents one case study (CS). The CPU time (in seconds) needed for the model training is shown in the left part of the table. The error metrics (root mean squared error, mean absolute error, coefficient of determination) are shown for the training and testing data (format: training/testing). The root mean square error ($RMSE$) and the mean absolute error (MAE) units are given in the last column for each case study, where the coefficient of determination ($R2$) is a unitless quantity. The identified algebraic expressions are indicated in Table 4.4, whereas the corresponding model parameters are reported in Table 4.5.

Train CPU [s]		***RMSE***		***MAE***		$\boldsymbol{R^2}$	
BMS	**GP**	**BMS**	**GP**	**BMS**	**GP**	**BMS**	**GP**
16500	12	0.152/0.152	0.010/0.013	0.117/0.117	0.005/0.007	0.996/0.996	1.000/1.000
5600	141	0.832/0.962	0.537/0.935	0.487/0.630	0.120/0.270	0.994/0.993	0.998/0.993
40600	6	0.017/0.014	0.002/0.009	0.011/0.010	0.002/0.006	0.954/0.970	0.999/0.989
123000	114	3.035/3.097	1.504/2.775	2.340/2.321	1.094/2.064	0.888/0.868	0.972/0.894

[a] Units for the case studies: kW (CSI), % (CSII), \$/kg (CSIII), and % (CSIV)

of MCMC iterations indicated in Table 4.2 as the stopping criterion. This led to CPU times of at least 5600 s in all the cases. In contrast, the GP could be trained much faster, in around 6 to 141 s seconds, depending on the CS. This fast training of the GP was expected, as there are highly efficient algorithms tailored to GPs, while the same is not true for symbolic regression. Besides, the GP relies on a given mathematical formalism, while symbolic regression assumes no pre-defined model structure. The $RMSE$ and MAE of the BMS surrogate

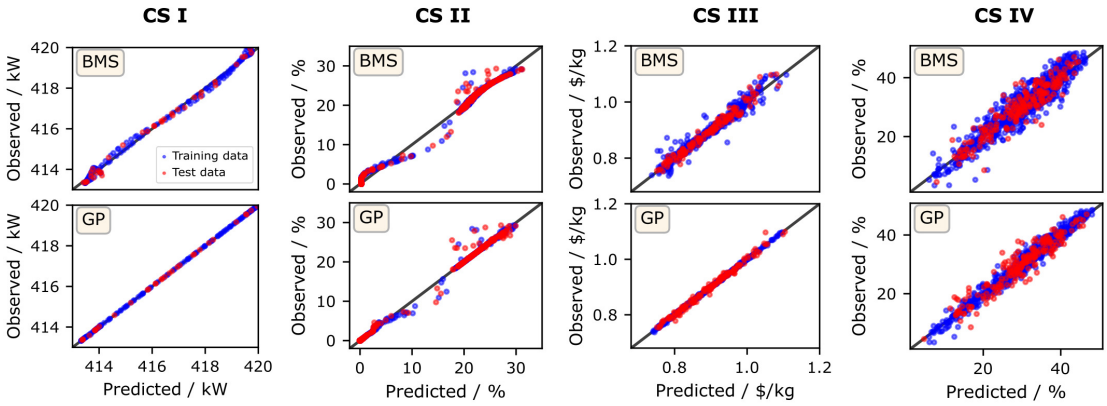

Figure 4.10. Observed vs. predicted (OVP) values for the different case studies are shown in the columns. The top row shows the OVP results obtained from the Bayesian machine scientist (BMS) predictions, whereas the bottom row shows the OVP results from the Gaussian process (GP) predictions. Blue points represent the training data, whereas red points correspond to the test data. The black line represents the values where the observed value corresponds to the model predictions.

is similar to that provided by the GP in all the cases except for CSI, with the R^2 values being close to one in all the examples, excluding CSIV. The BMS reached very similar errors in the training and the test sets. This shows that it is well-regularized and, therefore, less prone to overfitting, in line with the authors' expectations (Guimerà et al., 2020). The GP, on the other hand, showed a higher discrepancy between the training and the test set. It is worth mentioning that we did not tune the hyperparameters for training the GP. The model training only aimed to get a sufficiently well-trained surrogate model that is appropriate to be subsequently sent to the optimizer. The errors of the BMS indicated in Table 4.3 correspond to the closed-form expression identified to have the highest plausibility and, therefore, the lowest description length. The models with the lowest description length are reported in Table 4.4, whereas the corresponding estimated parameter values are shown in Table 4.5. It is worth mentioning that the training data was not always scaled, which explains the high difference in orders of magnitude of the estimated parameters in Table 4.5.

Table 4.4. The most plausible closed-form expressions for each case study (CS) identified by the Bayesian machine scientist (BMS) are shown. The corresponding estimated parameter values are reported in Table 4.5. In the right part of the table, the cell is colored in green if the stated expression includes the corresponding variable. On the other hand, it is colored in red if the variable is not included in the equation. Grey color is used where the corresponding variable is not included in the case study. The variable descriptions for each case study are given in Section 4.4.1

CS	Most plausible surrogate model $F(x)$ identified by the BMS	Variables							
I	$a_3\left(a_3+\left(a_2\left(a_3^b+b^{(a_1(a_1+b))^{a_0 b}}\right)\right)^{a_1^{\exp b}}\right)$	b	-	-	-	-	-	-	-
II	$\left(a_6+\frac{-\frac{a_4 P}{a_4+\left(\frac{P}{a_3}\right)^{a_7 T}}+P}{a_6(a_5+T)}\right)^{a_1}$	T	P	-	-	-	-	-	-
III	$a_{10}P+a_2\left(a_{11}\left(a_9^2\left(T^{\frac{a_0^F}{a_3}}\right)^{2a_6}\right)^{a_1^{T(a_8+V)}}(a_7+T)+a_3+F\left(a_3+\eta\right)\left(a_3+F\right)\right)+\eta^2+\eta$	T	P	η	V	F	ζ	-	-
IV	$-a_{11}b_1+a_{11}T_3+b_2^{a_9+a_9^{T_1}}-\left(a_3^{T_2}\right)^{T_2+P_2}+\frac{a_0+b_1+T_3\exp P_2}{a_1^{a_2 P_3+T_3}a_9+a_4}$	b_1	b_2	T_1	T_2	T_3	P_1	P_2	P_3

Table 4.5. Parameter values of the most plausible surrogate models (Table 4.4) identified by the Bayesian machine scientist for each case study (CS).

Parameter	CS			
	I	**II**	**III**	**IV**
a_0	$-6.435 \cdot 10^{1}$	$1.000 \cdot 10^{0}$	$1.542 \cdot 10^{2}$	$-4.304 \cdot 10^{0}$
a_1	$7.128 \cdot 10^{-1}$	$3.248 \cdot 10^{-1}$	$2.137 \cdot 10^{-5}$	$2.196 \cdot 10^{10}$
a_2	$2.280 \cdot 10^{1}$	$1.000 \cdot 10^{0}$	$-2.495 \cdot 10^{0}$	$2.712 \cdot 10^{-1}$
a_3	$1.487 \cdot 10^{1}$	$2.013 \cdot 10^{3}$	$-1.384 \cdot 10^{0}$	$1.504 \cdot 10^{-4}$
a_4	-	$3.167 \cdot 10^{59}$	$1.000 \cdot 10^{0}$	$1.451 \cdot 10^{0}$
a_5	-	$-2.132 \cdot 10^{2}$	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$
a_6	-	$1.056 \cdot 10^{-2}$	$-9.134 \cdot 10^{-2}$	$1.000 \cdot 10^{0}$
a_7	-	$2.177 \cdot 10^{-1}$	$-3.614 \cdot 10^{0}$	$1.000 \cdot 10^{0}$
a_8	-	-	$1.099 \cdot 10^{0}$	$1.000 \cdot 10^{0}$
a_9	-	-	$1.029 \cdot 10^{-1}$	$2.336 \cdot 10^{-5}$
a_{10}	-	-	$-2.734 \cdot 10^{-1}$	$1.000 \cdot 10^{0}$
a_{11}	-	-	$-4.735 \cdot 10^{-1}$	$-1.519 \cdot 10^{0}$

The BMS finds fairly complex expressions, including many nonlinear terms. The rightmost column of Table 4.4 summarizes the inclusion (or exclusion) of certain variables in the identified equation. The BMS excludes ζ (reflux ratio) in CSIII and P_1 (pressure in reactor 1) in CSIV. Therefore, according to the BMS, these variables marginally influence the target process response f, so they are omitted. This is a relevant piece of important information since the variables chosen by the BMS will represent the decision variables of the subsequent optimization problem. For problems with low dimensionality, one can visualize the identified model together with the training data in a two- or three-dimensional plot. By doing so, one might be able to guess where the optimum lies, as shown in Figure 4.11 for CSI (a) and CSI (b, c).

4.5.2 Model-based optimization

The ability to identify the global optimum of the true model depends on the accuracy of the surrogate model and the performance of the GO algorithm. For example, as seen, the models for CSI and CSII displayed in Table 4.4 are able to represent the training data precisely. As a result, the global optimum identified by BARON should lie near the global optimum of the original model, as shown in Figure 4.11 (a) and (c). Since this is a model-based global optimization, the optimum objective is only an approximation to the underlying system. Therefore, a discrepancy between the training data and the optimum point is expected. This

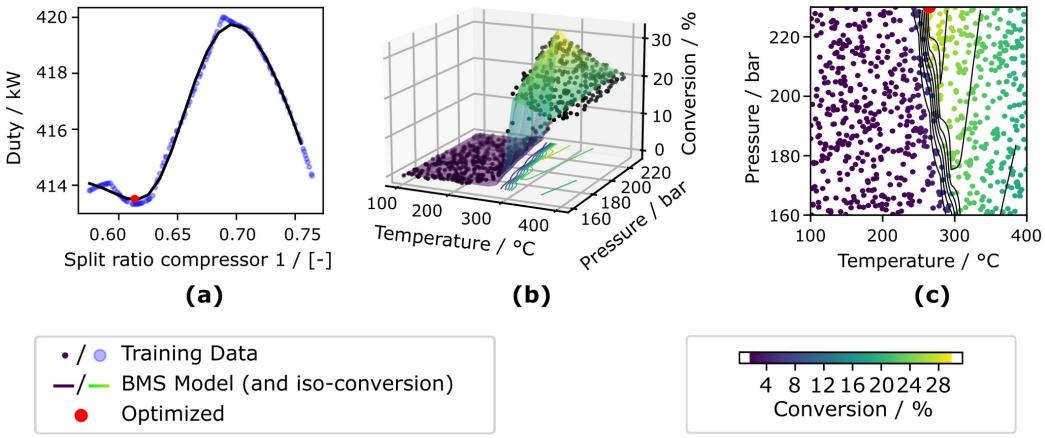

Figure 4.11. The training data (circles) is shown together with the model predictions for case study I (a), and case study II (b,c). In (a), the Bayesian machine scientist (BMS) predictions are shown as a black line. In (b), the surface represents the BMS predictions projected onto the $T-P$-plane and presented by the contour lines. These projections are shown in detail in (c). The identified model-based optimum value is indicated as a red circle in (a) and (c).

phenomenon is visible in Figure 4.11 (a). The same concept applies to higher dimensions.

The full optimization results are reported in Table 4.6 (for the BMS approach), Table 4.7, and Table 4.8 (for the GP approach), which we will discuss in detail. BARON/ANTIGONE solved all the global optimality problems in around 1 s, except for CSIV, which was optimized in 2 s. This emphasizes the considerable advantage of having a closed-form expression at hand when globally optimizing the surrogate. The solver returns to have found the global optimum for every case study within the optimality gap chosen for all case studies (Table 4.2).

For the optimization with MAiNGO, we varied the optimality gap (ϵ_R) of MAiNGO for the values ($\epsilon_R = [10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}]$. The maximum allowed CPU times were set to the corresponding training time the BMS required in the CSs: 16 500 s (CSI), 5600 s (CSII), 41 000 s (CSIII), and 123 500 s (CSIV). Multi-start was not applied. With these settings, MAiNGO could only solve the first two examples to global optimality (Table 4.7), while in the other two CSs, it failed to close the lowest chosen optimality gap of $\epsilon_R = 10^{-1}$ within the maximum allowed CPU time and reports a local solution instead. The reader is also referred to the supplementary material Section C.4, where we reported the complete results for the different

values of the relative optimality gap without a multi-start option.

For completeness, we additionally used a multi-start option in MAiNGO with a maximum allowed CPU time of 3600 s and the default relative optimality gap of $\epsilon_R = 10^{-3}$, where these results are shown in Table 4.8. It is observed that the multi-start option leads to better solutions in terms of the objective function (and CPU time). As mentioned above, the purpose of this manuscript was not to prove the superiority of our proposed approach over MAiNGO. Therefore, we further consider the best possible results that one can obtain by MAiNGO, which are obtained by the multi-start results given in Table 4.8.

Moreover, both approaches, BARON and MAiNGO (multi-start), lead to similar solutions in terms of objective function value, except for the last case where MAiNGO performs better despite following a simple multi-start approach that fails to guarantee convergence to the global optimum of the surrogate.

Specifically, in the smallest problem (CSI), the two solvers identified the same model-based optimum value, where the *RAE* (mismatch between the model-based optimum and the simulation) is zero in both cases. Slightly different model-based optima were identified in CSII and CSIII. In CSII, solutions displaying conversions of 31.46% and 31.03% were found by BARON and MAiNGO, respectively. In CSIII, unitary cost values of 0.716 \$/kg (BARON) and 0.709 \$/kg (MAiNGO) were found. Moreover, the *RAE*s were in a similar range of 6-7% for CSII, and 2% for CSIII. In the largest problem (CSIV), different model-based optima were identified: 46.21% and 54.66% conversions were found by BARON and MAiNGO, respectively. The final RAE, i.e., the mismatch between the objective function of the surrogate at its optimum and the true value of the objective function at the same point, is around 2% for the BMS-BARON approach, and around 9% in the GP-MAiNGO approach.

It is worth mentioning that the maximum time limit of 3600 s for the optimization in MAiNGO (multi-start) was never reached. In CSII, MAiNGO identified $F(x^*_{GP}) = 31.03\%$ to be the optimum of the surrogate model (reported to be a feasible point due to the multi-start option), with an actual value in HYSYS of $f(x^*_{GP}) = 29.40\%$. On the other hand, the BMS-BARON approach led to an actual value in HYSYS of $f(x^*_{BMS}) = 29.31\%$. A similar result was observed in CSIII ($f(x^*_{GP}) = 0.720\$/kg$ versus $f(x^*_{BMS}) = 0.727\$/kg$) and CSIV ($f(x^*_{GP}) = 49.94\%$

Table 4.6. The optimization performance criteria and results are summarized for the Bayesian machine scientist. The results are shown for the different case studies individually (columns). In the first row, the optimization direction is given. The CPU time (in seconds) needed for model-based optimization is shown in the second row, followed by the model status obtained from BARON/ANTIGONE. The solution x^* found during the optimization is evaluated in Aspen HYSYS to obtain $f(x^*)$, which is then compared to the value of the surrogate $F(x^*)$ in the same point to determine the relative absolute error (which measures the mismatch between the surrogate and the original process model in the optimal solution found).

	CSI	**CSII**	**CSIII**	**CSIV**
Optimization direction	min	max	min	max
CPU Optimization	1 s	1 s	1 s	2 s
Model status	Globally optimal	Globally optimal[a]	Globally optimal	Globally optimal
$F(x^*_{BMS})$	413 kW	31.46 %	0.716 \$ kg^{-1}	46.21 %
x^*_{BMS}	0.61 [−]	265 °C 230 bar	209 °C 5848 kmol h^{-1} 0.001 [−] 1.526 [−] 55 m^3 5497 kPa	1 [−] 0 [−] 400 °C 382 °C 311 °C 195 bar 230 bar 230 bar
$x^* \rightarrow$ HYSYS $\rightarrow f(x^*)$	413 kW	29.31 %	0.727 \$ kg^{-1}	47.03 %
RAE	0 %	7 %	2 %	2 %

[a] BARON reported locally optimal solution. ANTIGONE, however, reports globally optimal solution with the same values for x^* as BARON reports.

versus $f(x^*_{BMS}) = 47.03\%$). The reason why MAiNGO is providing slightly better solutions might be due to its better model training capabilities: The GP could be trained to achieve lower training errors and high R^2 values (Table 4.3). Considering, for example, Figure 4.10 and Figure 4.11, the GP is perfectly predicting the training data ($R^2 = 1.000$). Since we then use the trained model as objective function in the optimization problem, the identified model-based optimum will most probably be closer to the observed one. A similar issue arises when considering an LBF model for the tasks described above. This approach can be a very effective modeling technique for certain case studies. However, the modeler needs to consider a variety of basis functions to reach a sufficient accuracy level. To show the advantages and disadvantages of LBF vs. BMS, an LBF model was applied to CSIV. The detailed results are given in the supporting information Section C.3.

Table 4.7. The optimization performance criteria and results are summarized for the Gaussian process without multi-start in MAiNGO. The results are shown for the different case studies individually (columns). The optimality gap was varied for $\epsilon_R = [10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}]$, where only the results are reported where the lowest ϵ_R was reached for the maximum allowed CPU time (16 500 s (CSI), 5600 s (CSII), 41 000 s (CSIII), and 123 500 s (CSIV)). In the first row, the optimization direction is given. The CPU time (in seconds) needed for model-based optimization is shown in the second row, followed by the model status obtained. Rows four and five indicate CPU times and model status. The solution x^* found during the optimization is evaluated in Aspen HYSYS to obtain $f(x^*)$, which is then compared to the value of the surrogate $F(x^*)$ in the same point to determine the relative absolute error (which measures the mismatch between the surrogate and the original process model in the optimal solution found).

	CSI	**CSII**	**CSIII**	**CSIV**
Optimization direction	min	max	min	max
CPU Optimization	7 s	4476 s	41 000 s	123 500 s
Model status	Global optimum	Global optimum	Feasible point	Feasible point
$F(x^*_{GP})$	413 kW	37.81 %	0.714 \$ kg^{-1}	54.49 %
x^*_{GP}	0.61 [−]	276 °C 198 bar	233 °C 5750 kmol h^{-1} 0.001 [−] 1.250 [−] 53 m^3 5375 kPa	1 [−] 0 [−] 292 °C 334 °C 337 °C 230 bar 200 bar 200 bar
$x^* \rightarrow$ HYSYS $\rightarrow f(x^*)$	413 kW	27.45 %	0.729 \$ kg^{-1}	50.04 %
RAE	0 %	38 %	2 %	8 %

In brief, a small set of basis functions ϕ of the input features x were chosen to fit the model $f(x) = w\phi(x)$. Using the least absolute shrinkage and selection operator (LASSO) and cross-validation to tune the hyperparameter, we found an R^2 of 0.828 for the training and an R^2 of 0.757 for the test data (vs. R^2 values of 0.888 and 0.868 for the BMS in training and testing, respectively). Although a sparse expression can be generated, it performs worse than the BMS for our case study. Furthermore, since the accuracy of the model is low, the optimization can lead to significant discrepancies between the optimal model output $F(x^*)$ and the process output $f(x^*)$, shown in Table 4.10.

Note, however, that the quality of the BMS could be improved. For example, the number of MCMC steps in the BMS training could be increased to let the

Table 4.8. The optimization performance criteria and results are summarized for the Gaussian process with multi-start in MAiNGO. The results are shown for the different case studies individually (columns). The optimality gap is chosen as the default value of $\epsilon_R = 10^{-3}$ and the maximum CPU time was set to 3600 s. In the first row, the optimization direction is given. The CPU time (in seconds) needed for model-based optimization is shown in the second row, followed by the model status obtained. Rows four and five indicate CPU times and model status. The solution x^* found during the optimization is evaluated in Aspen HYSYS to obtain $f(x^*)$, which is then compared to the value of the surrogate $F(x^*)$ in the same point to determine the relative absolute error (which measures the mismatch between the surrogate and the original process model in the optimal solution found).

	CSI	**CSII**	**CSIII**	**CSIV**
Optimization direction	min	max	min	max
CPU Optimization	1 s	26 s	22 s	89 s
Model status	Feasible point	Feasible point	Feasible point	Feasible point
$F(x^*_{GP})$	413 kW	31.03 %	0.709 \$ kg^{-1}	54.66 %
x^*_{GP}	0.61 [–]	261 °C 224 bar	232 °C 5861 kmol h^{-1} 0.001 [–] 1.250 [–] 54 m^3 5500 kPa	1 [–] 0 [–] 293 °C 334 °C 339 °C 230 bar 206 bar 197 bar
$x^* \rightarrow$ HYSYS $\rightarrow f(x^*)$	413 kW	29.40 %	0.720 \$ kg^{-1}	49.94 %
RAE	0 %	6 %	2 %	9 %

algorithm further sample through the space of plausible expressions. Performing a parameter tuning could further support the exploration of the space of expressions (i.e., probabilities of evolutionary operations). Moreover, according to Guimerà et al. (2020), the priors could be tailored to the specific problem at hand to improve further the performance of the model training.

Table 4.9 shows the best solution for the training samples (BS). This solution was compared to the ones of the BMS and the GP. Considering the model-based solution $F(x^*_i)$, for CSI, II, and III, the deviation of the BARON solution from the best training point is similar to the one of the MAiNGO solution. For CSIV, this deviation is slightly higher for the MAiNGO solution. This may happen due to the larger dimension of this problem (8 decision variables), which leads to a lower R^2 (Table 4.3). Since a model-based optimization is performed, it might happen

that the identified optimum solution is not necessarily better than the sampling points due to the mismatch between the surrogate and the true model.

Table 4.9. The identified model-based optima $F(x_i^*)$ for the BMS and the GP are shown together with the identified solution $f(x^*)$ after inserting x^* into HYSYS. The best solution (BS) obtained by sampling the training data is also indicated. The relative absolute error (RAE) between $F(x_i^*)$ and $f(x_i^*)$ is shown.

	CSI		**CSII**		**CSIII**		**CSIV**	
	Minimization		**Maximization**		**Minimization**		**Maximization**	
	BMS	**GP**	**BMS**	**GP**	**BMS**	**GP**	**BMS**	**GP**
$F(x_i^*)$	413 kW	413 kW	31.46 %	31.03 %	0.716 $ kg^{-1}	0.709 $ kg^{-1}	46.21 %	54.66 %
$f(x_i^*)$	413 kW	413 kW	29.31 %	29.40 %	0.727 $ kg^{-1}	0.720 $ kg^{-1}	47.03 %	49.94 %
BS	413 kW		29.31 %		0.737 $ kg^{-1}		48.58 %	

Finally, in this work, the impact of the size of the training dataset was studied for one of the case studies. As shown in Table 4.9, the best sample found in the training set is close to the optimal solution identified with the optimized BMS and GP models, and more so in the cases with a search space of lower dimension (e.g., gap between $f(x_i^*)$ and $F(x_i^*)$ is found <0.3% in CSI and II with one and two decision variables, respectively, and <3.2% in CSIII and IV, with six and eight variables, respectively). Moreover, considering the bounds on the optimization variables displayed Table 4.1 and focusing on CSII, we repeated the calculations for several training set sizes (Table 4.10, where the rows represent the different data set sizes of which 20% was used for the test set, as described in Section 4.4.1). According to the results, surrogate optimization tends to lead to better solutions. However, as expected, the quality of the best solution from the sampling improves as we increase the number of samples. Moreover, Figure 4.12 shows the three-dimensional plots of the corresponding BMS predictions together with the training data.

For the expressions obtained with 200 and 400 samples, models with R^2 values lower than 0.85 were obtained in the test set (bold values in Table 4.10). According to the acceptance criteria defined in Chapter 4.5.1 (i.e., $R^2 > 0.85$), these models are not performing well enough. All models obtained by the GP led to a testing $R^2 > 0.85$. Optimizing the obtained models led to the results shown in Table 4.11 (for the BMS) and Table 4.12 (multi-start MAiNGO).

In general, the optimizer used for the identified BMS expressions (either BARON

Table 4.10. The training performance criteria are summarized for the Bayesian machine scientist (BMS) and the Gaussian process (GP) for CSII. Each row shows the number (No) of samples (training plus testing) together with the CPU time needed for sampling this data set. The CPU time (in seconds) required for the model training is also shown. The error metrics (root mean squared error, mean absolute error, coefficient of determination) are shown for the training and testing data (format: training/testing). The units root mean square error ($RMSE$) and the mean absolute error (MAE) units are percentages (conversion), where the coefficient of determination (R^2) is a unitless quantity. Bold R^2 are indicated where the corresponding BMS models fulfill the acceptance criteria of having a model that fits the test data well enough ($R^2 > 0.85$).

Sampling		**Training CPU**		***RMSE***		***MAE***		$\boldsymbol{R^2}$	
No	**CPU**	**BMS**	**GP**	**BMS**	**GP**	**BMS**	**GP**	**BMS**	**GP**
200	73	2388	9	1.504/2.252	0.930/0.794	0.961/1.318	0.333/0.322	0.981/0.957	0.993/0.995
400	130	3136	14	1.784/1.562	0.982/0.621	0.966/0.904	0.299/0.255	0.974/0.980	0.992/0.997
600	195	5290	60	0.824/0.973	0.599/1.478	0.465/0.481	0.158/0.370	0.994/**0.993**	0.997/0.983
800	257	5616	113	0.949/1.198	0.636/1.092	0.549/0.648	0.158/0.282	0.993/**0.989**	0.997/0.991
1000	600	5600	141	0.832/0.962	0.537/0.935	0.487/0.630	0.120/0.270	0.994/**0.993**	0.998/0.993

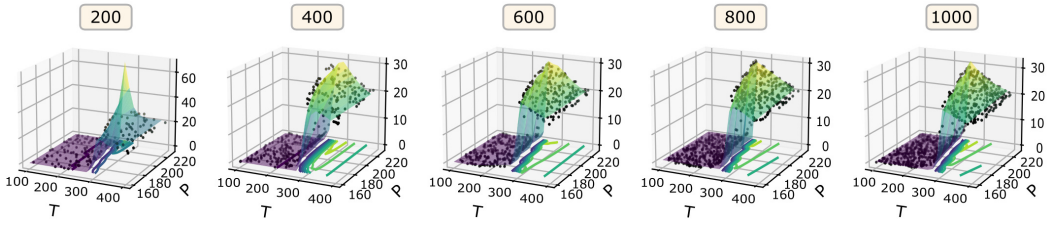

Figure 4.12. The training data (circles) is shown together with the BMS predictions (surface) for case study II for the cases with 200 (most left) up to 1000 samples (most right). The contour lines on the $T-P$-plane of the model predictions are shown as colored lines.

or ANTIGONE) led to globally optimal solutions, except for the model identified using the dataset with 200 samples. Since a multi-start was used in MAiNGO, feasible points were reported instead of the global solutions. It is again observed that the GO of the BMS models is executed faster than the optimization of the trained GPs. Additionally, the mismatches between the model-based optimal solutions $F(x_i^*)$ and the HYSYS outputs $f(x_i^*)$ are generally higher than in the solutions found with MAiNGO. Lastly, the surrogate approaches identified solutions that were better than the best training set sample in all cases except for the case with 400 samples.

In the BMS model trained with 200 samples, BARON found a locally optimal solution. The identified x_{BMS}^* was very close to the identified global solution with 1000 samples. However, the corresponding solution x_{BMS}^* led to a large mismatch

between the HYSYS output $f(x^*_{BMS})$ and $F(x^*_{BMS})$. A visual reason for this is reported in Figure 4.12 (model peak with 200 samples). This might indicate that the training data set is too small for the BMS, leading to broad likelihoods in the parameter space, which the authors of the BMS (Guimerà et al., 2020) also discussed. Considering the optimization of the GP trained with 200 samples and optimized with MAiNGO, the *RAE* was slightly smaller. Nevertheless, the identified minimizer x^*_{GP} led to a worse solution in HYSYS compared to the BMS (27.22% vs. 29.24%). By increasing the size from 200 to 1000 samples, it is observed that both approaches tend to identify optimal solutions better than the best sample in the training set (bold values in Table 4.11 and Table 4.12). However, the gap between the optimal solutions from surrogate optimization and sampling does not change significantly by varying the size of the training set.

Overall, for this case study, the sampling sizes can be regarded to be appropriate for the model training and subsequent optimization.

Table 4.11. The optimization performance criteria and results are summarized for the Bayesian machine scientist for CSII. The results are shown for the different data sets. In the first row, the optimization direction is given. The CPU time (in seconds) needed for model-based optimization is shown in the second row, followed by the model status obtained from BARON (or ANTIGONE if indicated). The solution x^* found during the optimization is evaluated in Aspen HYSYS to obtain $f(x^*)$, which is then compared to the value of the surrogate $F(x^*)$ in the same point to determine the relative absolute error (which measures the mismatch between the surrogate and the original process model in the optimal solution found). The best training sample found (BS) is reported in the last row. Bold values show a HYSYS output $f(x^*)$ better or equal than BS.

Size of the dataset	**200**	**400**	**600**	**800**	**1000**
Optimization direction	max	max	max	max	max
CPU Optimization	1 s	1 s [a]	2 s	1 s [b]	1 s
Model status	Locally optimal	Globally optimal [a]	Globally optimal	Globally optimal [b]	Globally optimal
$F(x^*_{BMS})$	68.85 %	30.26 % [c]	28.36 %	29.58 % [c]	31.46 %
x^*_{BMS}	266 °C 230 bar	288 °C 230 bar	273 °C 220 bar	257 °C 230 bar	265 °C 230 bar
$x^* \rightarrow$ HYSYS $\rightarrow f(x^*)$	29.24 %	27.62 %	28.38 %	29.84 %	29.31 %
RAE	58 %	9 %	0 %	1 %	7 %
BS	28.35 %	28.94 %	29.18 %	29.59 %	29.31 %

[a] ANTIGONE (v41.3.0, rel. optim. gap: 10^{-9}, max. CPU time: 600 s, node limits: 10^6) closed the optimality gap. BARON reached the max CPU time of 600 s.

[b] ANTIGONE (v41.3.0, rel. optim. gap: 10^{-9}, max. CPU time: 600 s, node limits: 10^6) closed the optimality gap. BARON closed the optimality gap ($\approx$1 s) with local optimality.

[c] The same solution was obtained for ANTIGONE and BARON.

4.6 Conclusion

This work introduced a method for the global optimization of process models based on algebraic expressions built from data via SR. Other approaches based on ML algorithms are hardly interpretable and lead to complex formulations. In contrast, our approach derives closed-form algebraic expressions in the space of degrees of freedom using well-established mathematical operators and Bayesian learning methods.

Numerical examples show that the algebraic models built by our method display a similar level of accuracy as those constructed with GPs. However, they can be more easily optimized to global optimality using state-of-the-art solvers than

Table 4.12. The optimization performance criteria and results are summarized for the Gaussian process with multi-start in MAiNGO for CSII. The results are shown for the different data sets. The optimality gap is chosen as the default value of $\epsilon_R = 10^{-3}$ and the maximum CPU time was set to 5600 s (roughly the time required for training the BMS models shown in Table 4.3). In the first row, the optimization direction is given. The CPU time (in seconds) needed for model-based optimization is shown in the second row, followed by the model status obtained. Rows four and five indicate CPU times and model status. The solution x^* found during the optimization is evaluated in Aspen HYSYS to obtain $f(x^*)$, which is then compared to the value of the surrogate $F(x^*)$ in the same point to determine the relative absolute error (which measures the mismatch between the surrogate and the original process model in the optimal solution found). The best training sample found (BS) is reported in the last row. Bold values show a HYSYS output $f(x^*)$ better or equal than BS.

Size of the dataset	**200**	**400**	**600**	**800**	**1000**
Optimization direction	max	max	max	max	max
CPU Optimization	2 s	5 s	11 s	20 s	26 s
Model status	Feasible point	Feasible point	Feasible point	Feasible point	Feasible point
$F(x^*_{GP})$	36.02 %	33.96 %	52.05 %	33.15 %	31.03 %
x^*_{GP}	282 °C 201 bar	273 °C 230 bar	262 °C 229 bar	262 °C 225 bar	261 °C 224 bar
$x^* \rightarrow$ HYSYS $\rightarrow f(x^*)$	27.22 %	28.73 %	29.44 %	29.37 %	29.40 %
RAE	24 %	15 %	43 %	11 %	6 %
BS	28.35 %	28.94 %	29.18 %	29.59 %	29.31 %

GP models that cannot be globally optimized in short CPU times. Because the algebraic surrogate is slightly less accurate, its global optimum is not guaranteed to optimize the original model globally and might be outperformed by simple multi-start strategies.

Although the BMS shows a very high computational time needed for training and leads to less accurate models, this might change in the future as better SR algorithms become available and a similar level of maturity is reached relative to standard ML training methods, e.g., GP and ANNs. Moreover, having a closed-form expression at hand, which could be tuned by appropriately modifying the SR settings, could help in cases where the resulting optimization problem has to be solved many times, as in real-time optimization problems.

Chapter 5

Algebraic surrogate-based flexibility analysis of process units with complicating process constraints

This chapter is based on the following publications: Forster T., Vazquez D., Moreno-Palancas I. F., Guillén-Gosálbez G. (2024a). Algebraic surrogate-based flexibility analysis of process units with complicating process constraints. *Computers and Chemical Engineering*, 184, 108630. and Forster T., Vazquez D., Moreno-Palancas I. F., Guillén-Gosálbez G. (2024b). Flexibility Analysis Using Surrogate Models Generated via Symbolic Regression. *Computer Aided Chemical Engineering*, 53, 2791-2796.

Nomenclature for Chapter 5

Sets

E	$\{e \mid e$ is a symbolic mathematical expression$\}$
G	$\{g \mid g$ is a non-complicating constraint$\}$
H	$\{h \mid h$ is a complicating constraint$\}$
I	$\{i \mid i$ is a sample$\}$
J	$\{j \mid j$ is a constraint$\}$
K	$\{m \mid m$ is an uncertain parameter$\}$
N	$\{n \mid n$ is a control variable$\}$
T	Set of uncertain parameters that maintain the process feasible
$\mathbb{R}$	Real numbers

Parameters

d	Design parameters of the process under consideration
M	Big-M reformulation parameter
$\Delta\theta_k^{min}$ and $\Delta\theta_k^{max}$	Maximum upper and lower deviation from a nominal point of the uncertain parameter k
$\underline{\theta_k}$ and $\overline{\theta_k}$	Lower and upper bounds of the uncertain parameter k

Variables

f_j	Process constraint
$\hat{f}_g$	Non-complicating process constraint
$\widetilde{f}_h$	Complicating process constraint
s_j, s_g, and s_h	Slack variables of constraints $f_j, \hat{f}_g$, and $\widetilde{f}_h$
t	Time
u	Upper bound for the constraint f_j
y_j, y_g, and y_h	Binary variable of constraints $f_j, \hat{f}_g$, and $\widetilde{f}_h$
z_n	Control variable n
δ	Scaled deviation from nominal point
θ_k	Uncertain parameter k
θ_k^c	Critical value of the uncertain parameter k
θ_k^N	Nominal operating point of the uncertain parameter k
γ_e	Symbolic expression e
λ_j, λ_g, and λ_g	Lagrange multiplier of constraints $f_j, \hat{f}_g$, and $\widetilde{f}_h$
ω_i	Feature vector of sample i used for the model training
$\mathcal{L}$	Lagrange polynomial
$\mathcal{DL}$	Description length of Bayesian machine scientist

5.1 Introduction

Uncertainty is always present in science and engineering. This uncertainty can reveal itself in, for example, product demands (Petkov & Maranas, 1997), supply chain and scheduling activities (Ehrenstein et al., 2019), and even in process design and operation (Pistikopoulos, 1995). A broad overview of various aspects of uncertainty, specifically in the Process Systems Engineering (PSE) field, is given by the works by Sahinidis (2004), Li and Ierapetritou (2008), and Grossmann et al. (2014). When uncertainty is not taken into account, designing and optimizing process units assuming deterministic values for the uncertain parameters can lead to suboptimal solutions or, in the worst case, to infeasibilities during operation (Ben-Tal & Nemirovski, 2002; I. E. Grossmann et al., 1983; Z. Li et al., 2011). Thus, it is common to embed uncertainty in the specifications of the problem, e.g., in the field of pharmaceutical development, the guidelines of the International Council for Harmonisation of Technical Requirements for Pharmaceuticals for Human Use (ICH) define that critical quality attributes (CQAs) are valid within a given acceptable range (FDA, 2010), even considering variations due to uncertain input conditions.

Considering the effect of uncertainties in optimization problems during the early design and operation of chemical plants is especially important for chemical processes. This is because optimal solutions tend to meet process constraints and quality requirements as deterministic inequalities or equalities, so any perturbation over the nominal conditions may have strong implications on their feasibility. There are two main mathematical methods in operations research to account for uncertainties in optimization problems, namely stochastic programming (Birge & Louveaux, 2011; Ierapetritou & Pistikopoulos, 1994; Z. Li & Ierapetritou, 2012; Marti & Kall, 1995; Prekopa, 1995; Shapiro et al., 2021) and robust optimization (Ben-Tal et al., 2009; Ben-Tal & Nemirovski, 2002; Z. Li & Ierapetritou, 2008; Lin et al., 2004). Li and Grossmann (2021) considered chance-constrained programming as another approach for optimization problems under uncertainty, yet (arguably) it could also be regarded as a generalization of robust optimization, in which distributions are specified for the uncertainties and a level of probability is defined to satisfy constraints (I. E. Grossmann et al., 2016).

The flexibility index is an alternative approach for accounting for uncertainties that has been used mainly in process design (Pistikopoulos, 1995). Developed

by the PSE community back in the 1980s (I. E. Grossmann et al., 1983; Halemane & Grossmann, 1983; Swaney & Grossmann, 1985a, 1985b), its primary goal is to assess the ability of a design to remain feasible against variations in the parameter values during the plant operation (Boukouvala et al., 2010; I. E. Grossmann et al., 1983). In essence, this is done by quantifying the feasibility of a given design, which describes if a process is feasible or infeasible within a given range. Mathematically, the feasibility function can be calculated by solving a min-max-optimization problem, which will be discussed in detail later in this work. Grossmann et al. (1983) geometrically interpret this feasibility function as the depth of the feasible region since it quantifies a deviation from the nominal constraints. Based on this concept, the authors describe the flexibility index (I. E. Grossmann et al., 1983; Swaney & Grossmann, 1985a, 1985b), which characterizes the size of the region of feasible operation (T) in the space of uncertain parameters. This region T should be a subset of the entire feasible region (Q. Zhang et al., 2016). In other words, the flexibility index describes the maximum range over which the involved uncertain parameters can vary (independently) such that the process remains feasible (I. E. Grossmann et al., 1983; Pulsipher et al., 2019). Alternatively, other metrics to quantify process flexibility were put forward. Those methods include for example the resilience index (Morari et al., 1985), and stochastic measures such as the design reliability (Kubic & Stein, 1988) and the stochastic flexibility index (Pistikopoulos & Mazzuchi, 1990; Straub & Grossmann, 1990, 1993). The stochastic flexibility index was developed to tackle the limitation of the flexibility index to address discrete and continuous uncertainties at the same time (Straub & Grossmann, 1990), or to use arbitrary probability distribution of the uncertainties (Rogers & Ierapetritou, 2015b).

The flexibility index can be computed using deterministic mathematical models (Pistikopoulos, 1995; Pulsipher et al., 2019) as long as process constraints are described in a closed-form algebraic manner (Floudas et al., 2001; Ierapetritou, 2001; Pistikopoulos & Ierapetritou, 1995; Straub & Grossmann, 1993). Specifically, the main methods to quantify the flexibility index include vertex searches (I. E. Grossmann et al., 1983; Swaney & Grossmann, 1985a, 1985b), active set strategies with KKT reformulations (I. E. Grossmann & Floudas, 1987), or branch-and-bound approaches (Ostrovsky et al., 1994) that were based on the evaluation of the lower and upper bounds of the feasibility function. Since global optimality cannot be guaranteed using local solvers for such bounding methods (Migdalas et al., 1998),

a global optimization approach was developed using reformulation and relaxation approaches for the feasible region (Floudas et al., 2001). We note that some of these approaches rely on specific convexity assumptions (Goyal & Ierapetritou, 2002, 2003; I. E. Grossmann & Floudas, 1987).

When some constraints are not available in algebraic closed form, analyzing process flexibility becomes much more complex and a straightforward computation of the flexibility index with state-of-the-art deterministic solvers is not possible anymore. This might happen, for instance, if the only knowledge about the system consists of observations of input and output data due to limited process understanding (Boukouvala & Ierapetritou, 2012). Additionally, very complex underlying process dynamics (ordinary or partial differential equations) can be another reason why constraints are difficult to be derived in a closed-form manner (Ding & Ierapetritou, 2021). Even if some knowledge about the process dynamics can be described by differential equations that could be discretized (i.e., orthogonal collocation on finite elements (Carey & Finlayson, 1975)), finding a solution might still be challenging due to the size of the reformulated optimization problem.

Several works applied adaptive sampling techniques with Kriging interpolation (Krige, 1951), also known as Gaussian process regression (Rasmussen & Williams, 2006), to perform flexibility analyses when dealing with situations where closed-form models for process constraints are inexistent or challenging to be constructed (Boukouvala & Ierapetritou, 2012; Boukouvala et al., 2011; Ding & Ierapetritou, 2021; Rogers & Ierapetritou, 2015a, 2015b; Z. Wang & Ierapetritou, 2017). Broadly speaking, these methods are used to approximate the feasibility function, namely the function that evaluates the feasibility of the model for given values of the decision variables and the parameters. Such data-driven strategies are applicable to processes with non-convex feasible regions (Rogers & Ierapetritou, 2015a, 2015b). Similarly, other works substitute the Gaussian process models with neural networks (Metta et al., 2021). In a very recent work by Sachio et al. (2023), the authors developed a highly flexible framework that performs a design space identification followed by a design space analysis. The researchers used a Sobol sampling approach with a subsequent approximation of the design space by alpha shapes, where the usage of alpha shapes was also successfully described in earlier works for feasibility analysis (Banerjee & Ierapetritou, 2005). All the methods mentioned in this paragraph approximate the feasibility function with a surrogate and they do not rely on the original deterministic flexibility index, but rather they

use alternative flexibility metrics.

Here, we shall develop an alternative approach for flexibility problems, focusing on the computation of the flexibility index where challenging process dynamics or hard-to-model process constraints are encountered. While more refined flexibility metrics have been proposed (Pistikopoulos & Mazzuchi, 1990; Straub & Grossmann, 1990), we focus on the original flexibility index metric due to the already existing methods for its computation applicable to analytical closed-form models, into which we reformulate process models with complicating constraints as explained later in the article. In the following, we use the term "complicating constraints" to describe hardly accessible or completely inaccessible constraints, that is, constraints that are either hard to model in algebraic form and/or hard to handle in an optimization model. In essence, here we shall replace those constraints with algebraic surrogates built with a symbolic regression algorithm (SR). These algebraic surrogates are hence subsequently incorporated into the original flexibility analysis formulation, thereby simplifying the flexibility analysis. SR algorithms aim to find the model structure and associated parameters that fit some data. Compared to algorithms like ALAMO or ALVEN that restrict the search to a specific set of functions, general SR approaches make use of symbolic expression trees that can represent a very large number of plausible algebraic surrogate models (Cozad & Sahinidis, 2018). Here, the best model in the symbolic tree can be identified following different approaches and applying some fitting criteria. These include the formulation and solution of an MINLP problem (Cozad & Sahinidis, 2018), where binary variables encode the model structure and continuous ones its parameters, or the application of stochastic search approaches (Cranmer et al., 2020; Diveev & Shmalko, 2021; Guimerà et al., 2020). For example, Cranmer et al. (2020) created the open-source algorithm PySR, a multi-population evolutionary algorithm, which is freely available in Python (Cranmer, 2020, 2023). There are also algorithms that are available as proprietary software, such as Eureqa (Schmidt & Lipson, 2009) or TuringBot (2023). To build the surrogate models in this work, however, we use an SR method developed by Guimerà et al. (2020), based on a Markov-chain Monte Carlo (MCMC) approach to identify the most suitable closed-form expression to represent the available data. One of the advantages of SR is that it does not assume a predetermined model structure or a reduced set of alternative model structures (e.g., like in the ALAMO approach (Wilson & Sahinidis, 2017) or the above mentioned HDMR approach). The user

only defines some allowable mathematical operations (i.e., addition, multiplication, subtraction, etc.) that are used in a symbolic tree to build plausible expressions to explain given data. This symbolic tree can be seen as a superstructure of mathematical expressions from which the most suitable one and its associated parameters must be identified using specific algorithms. SR was successfully applied in many different fields, such as distillation (Ferreira, Pedemonte, & Torres, 2019; Ferreira, Torres, & Pedemonte, 2019; McKay et al., 1997), food extrusion process (McKay et al., 1999), process control (Keane et al., 1993), or the discovery of physical laws (Cranmer et al., 2020; Schmidt & Lipson, 2009). Moreover, the BMS was also previously applied by some of us to approximate process simulations of carbon capture plants (Negri et al., 2022), to model the link between energy-related impacts and socioeconomic drivers in macro-economic studies (Vázquez et al., 2022), and for surrogate-based global optimization of process units and flowsheets by coupling SR with deterministic global optimization (Forster, Vázquez, & Guillén-Gosálbez, 2023a).

Our proposed approach represents an alternative way to handle complicating constraints in flexibility problems that does not rely on any discretization technique, like those applied to differential equations, thereby avoiding the use of auxiliary variables that increase the dimensionality of the optimization problem. Additionally, no pre-defined model structure is assumed for the surrogate model replacing the complicating constraints. Instead, an SR algorithm, the BMS, creates an algebraic model from a set of samples of the functions describing the complicating process constraints. We show the advantages of this approach in two case studies covering a chromatographic column of an antibody production process and a bioethanol production in fed-batch operation mode. To the best of our knowledge, this is the first work that combines SR with the initially defined flexibility index problem, giving rise to a hybrid optimization problem where some constraints are replaced with algebraic surrogates. In the end, the most appropriate approach to quantify flexibility performance in the presence of complicating constraints will depend on the problem at hand and the goal and scope of the analysis, including the selection of the flexibility metric to be evaluated.

The remainder of this chapter is organized as follows: First, the problem statement is described, followed by the methodology. Afterward, two case studies are introduced, and the results are subsequently discussed. Finally, the conclusions of the work are drawn.

5.2 Problem statement

Here, without loss of generality, we shall consider an existing process or process unit, where a known and fixed process design (i.e., equipment dimensions) is given by variables d. Additionally, there are K uncertain parameters $\theta_k, k \in K$, which have a given nominal value of θ_k^N. Last, there are N control variables, with a value $z_n, n \in N$, that can be adjusted during the operation to regain feasibility.

Within this process, a set of J process constraints $f_j \ \forall j \in J$ (i.e., material balances, process or product specifications or restrictions, etc.) need to be considered, as stated in equation (5.1):

$$f_j\left(d, z, \theta\right) \leq 0, \quad \forall j \in J \tag{5.1}$$

For such a situation, we want to assess how far the uncertain parameters θ can deviate from the nominal operating point θ^N, such that the process remains feasible, i.e., we are interested in the flexibility index problem as described later in the next section. To quantify the flexibility of a process, the feasibility function given in equation (5.2) must be assessed. To do so, the min-max-optimization problem shown in equation (5.2) must be computed:

$$\psi\left(d, \theta\right) = \min_{z} \max_{j \in J} \left\{ f_j(d, z, \theta) \right\} \tag{5.2}$$

In this expression, $\psi\left(d, \theta\right)$ represents the feasibility function for a given design d and a realization of the uncertain parameters θ. However, some of the process constraints $f_j, j \in J$, might be very challenging to be evaluated, or might not even be directly accessible as closed-form algebraic equations. As a consequence, they cannot be directly included in the formulation given in equation (5.2). Complicating constraints might be encountered in complex systems (i.e., involving complex process dynamics, with complex unit operations hard to model mechanistically).

Hence, we divide the set of constraints J into two proper subsets $G \subset J$ and $H \subset J$, as shown in equation (5.3). Set G contains process constraints $\hat{f}_g, g \in G$ that are non-complicating, i.e., clearly defined by an algebraic equation that can be

easily incorporated into (5.2) and handled numerically in an efficient manner. Set $H \subset J$, on the other hand, contains complicating constraints, denoted by $\widetilde{f}_h, h \in H$, which cannot be incorporated directly into the model in a straightforward manner. Note that whether one constraint should be considered complicating or not might depend on the specific case and the numerical performance of the standard approach.

$$\begin{aligned} \hat{f}_g\left(d, z, \theta\right) &\leq 0, \quad \forall g \in G \\ \widetilde{f}_h\left(d, z, \theta\right) &\leq 0, \quad \forall h \in H \end{aligned} \tag{5.3}$$

The idea here is to replace the complicating constraints in equation (5.2) with algebraic surrogate models that are constructed by solving an SR problem. Herein, we shall identify such a surrogate model without assuming a pre-defined model structure, as discussed next.

5.3 Methodology

For the sake of completeness, we will first present the flexibility index formulation developed by Grossmann et al. (1983), Halemane and Grossmann (1983), and Swaney and Grossmann (1985a, 1985b), which is taken as a basis to derive our approach. The reader is referred to these works for more details and further mathematical insights. For simplicity, during the subsequently shown derivation, we use the set J to describe all the constraints, where we split this set into the two subsets G and H - as shown in Section 5.2 - in the very end of the derivation. After that, we describe how the surrogate models can be incorporated in the flexibility formulation. Last, we discuss how to build these surrogate models and assess their performance.

5.3.1 Fundamentals of feasibility and flexibility

Consider the formulation in equation (5.4) that aims to calculate the feasibility function $\psi(d, z, \theta)$ of a given design d and a specific realization of $\theta_k, k \in K$, where some control variables $z_n, n \in N$ are present (I. E. Grossmann et al., 1983; Halemane & Grossmann, 1983; Swaney & Grossmann, 1985a, 1985b):

$$\psi\left(d, \theta\right) = \min_{z} \max_{j \in J} \left\{ f_j(d, z, \theta) \right\} \tag{5.4}$$

Using an upper bound u for the constraints $f_j, j \in J$, we can reformulate the min-max formulation into the following single-level problem:

$$\begin{aligned} \psi(d,\theta) &= \min_{z,u} u \\ \text{s.t.} \quad & f_j(d,z,\theta) \leq u \quad \forall j \in J \end{aligned} \tag{5.5}$$

Formulation (5.5) seeks the smallest u such that each constraint f_j results in a value less or equal to u. Overall, a value of $\psi(d,\theta) \leq 0$ means the process is feasible for a given realization of d and θ. On the other hand, $\psi(d,\theta) > 0$ implies that the process is infeasible for these specific values of d and θ.

The feasibility formulation seeks the worst value of $\psi(d,\theta)$ over the entire uncertain parameters space $\theta \in T$. This problem can be formulated as the following tri-level optimization model, which provides the feasibility test function $\chi(d)$ given in equation (5.6).

$$\begin{aligned} \chi(d) &= \max_{\theta \in T} \psi(d,\theta) \\ &= \max_{\theta \in T} \min_{z} \max_{j \in J} \{f_j(d,z,\theta)\} \end{aligned} \tag{5.6}$$

In formulation (5.6), if $\chi(d) \leq 0$, the process is feasible for the entire space of the uncertain parameters Θ. Using formulation (5.5) given above for the feasibility function $\psi(d,\theta)$, the feasibility test problem in equation (5.6) can be reformulated as a bilevel optimization problem shown in equation (5.7).

$$\begin{aligned} \chi(d) &= \max_{\theta} \psi(d,\theta) \\ \text{s.t.} & \psi(d,\theta) = \min_{z,u} u \\ & \text{s.t.} = f_j(d,z,\theta) \leq u, \quad \forall j \in J \\ & \theta \in T \end{aligned} \tag{5.7}$$

Grossmann et al. (1983) proposed an approach to quantify and identify the largest possible uncertainty set $\theta \in T$, such that the process is still feasible over the entire range of θ. The authors described this as the flexibility index problem, which is given in equation (5.8).

$$
\begin{aligned}
FI = & \max_{\delta \in \mathbb{R}_{\leq 0}} \delta \\
& \text{s.t.} \chi(d) = \max_{\theta} \psi(d, \theta) \leq 0
\end{aligned}
\tag{5.8}
$$

Where FI represents the flexibility index, and δ should be a nonnegative real number ($\mathbb{R}_{0\leq}$). The newly introduced variable δ scales the uncertainty set T, which is therefore subsequently denoted by $T(\delta)$. In other words, δ can be regarded as a scaled deviation from a nominal point θ^N, such that the realization of θ results in a feasible solution. The goal is to maximize the mentioned set $T(\delta)$, under which there exists the possibility of recovering feasibility through the control variable z. In their original work, Swaney and Grossmann (1985a, 1985b) showed that the bilevel problem given in equation (5.8) can be reformulated. Instead of searching for the largest possible set $T(\delta)$ by maximizing δ, the authors showed that it is equivalent to looking for the minimum δ such that the solution is located precisely on the boundary ($\psi(d, \theta) = 0$). In other words, one is looking for the constraint that is closest to the nominal operating point. This reformulation can therefore be expressed as shown in equation (5.9).

$$
\begin{aligned}
FI = & \min_{\delta \in \mathbb{R}_{0\leq}} \delta \\
& \text{s.t.} \chi(d) = \max_{\theta} \psi(d, \theta) = 0
\end{aligned}
\tag{5.9}
$$

The flexibility index problem shown in equation (5.9) ensures that the feasibility function is precisely zero. Using the definition of the feasibility test problem given in equation (5.7), the flexibility index problem can be reformulated as follows:

$$
\begin{aligned}
FI = & \min_{\delta \in \mathbb{R}_{0\leq}} \delta \\
& \text{s.t.} \chi(d) = \max_{\theta} \psi(d, \theta) = 0 \\
& \quad \text{s.t.} \psi(d, \theta) = \min_{z} u \\
& \quad f_j(d, z, \theta) - u + s_j = 0, \quad \forall j \in J \\
& \quad \theta \in T(\delta)
\end{aligned}
\tag{5.10}
$$

Where the inequality constraints of problem (5.7) are expressed as equality constraints using nonnegative slack variables, s_j. The resulting flexibility index problem is challenging due to the non-differentiability of max-min-max (or min-max-min) functions. To tackle this challenge, we can substitute the innermost optimization problem with its Karush-Kuhn-Tucker (KKT) conditions (I. E. Grossmann et al., 2014). The Lagrange function $\mathcal{L}(d, \theta)$ of this innermost problem can be formulated as follows:

$$\mathcal{L}(d, \theta) = u + \sum_j \left(\lambda_j \left(f_j \left(d, z, \theta\right) - u + s_j\right)\right) \tag{5.11}$$

Where λ_j represents the Lagrange multipliers for constraint f_j. Subsequently, the corresponding stationary (5.12) and complementarity (5.13) conditions for problem (5.10) therefore read as follows:

$$\begin{aligned}
\frac{\partial \mathcal{L}(d, \theta)}{\partial u} &= 0 = 1 - \sum_j \lambda_j \\
\frac{\partial \mathcal{L}(d, \theta)}{\partial z_n} &= 0 = \sum_j \lambda_j \frac{\partial f_j \left(d, z, \theta\right)}{\partial z_n}\ , \quad \forall n \in N \\
\frac{\partial \mathcal{L}(d, \theta)}{\partial \lambda_j} &= 0 = f_j \left(d, z, \theta\right) - u + s_j, \quad \forall j \in J
\end{aligned} \tag{5.12}$$

$$\begin{aligned}
\lambda_j s_j = 0, \quad \forall j \in J \\
\lambda_j, s_j \geq 0, \quad \forall j \in J
\end{aligned} \tag{5.13}$$

In 1987, Grossmann and Floudas (1987) described how problem (5.10) can be reformulated into a mixed-integer nonlinear program (MINLP) by applying an active set strategy where some constraints might be inactive in the optimal solution. The usage of active set methods requires making discrete choices on the complementarity conditions $\lambda_j s_j$. Therefore, it is necessary to introduce binary variables $y_j \in \{0, 1\}$ that establish whether a constraint is active ($y_j = 1$) or not ($y_j = 0$). Furthermore, the KKT complementarity conditions are formulated using the following two inequalities in equation (5.14).

$$\begin{aligned} s_j &\leq M\left(1-y_j\right), \quad \forall j \in J \\ \lambda_j &\leq y_j, \quad \forall j \in J \end{aligned} \tag{5.14}$$

Where M represents a large enough parameter that acts as the upper bound for the slack variables s_j. Properly selecting M is one of the main drawbacks of this method since it is hard to define tight bounds for the Lagrange multipliers. If M is too small, the solution obtained with the reformulation in equation (5.14) will not coincide with the optimum of the original problem, since this value would act as an active constraint. On the other hand, an excessively large M often causes numerical instabilities (Cococcioni & Fiaschi, 2021). Consequently, its value must be selected in accordance with the problem, which might not be easy. In addition to the transformations mentioned above, another constraint could be added that enforces the number of potential sets of active constraints to be lower or equal to $|N|+1$, where $|N|$ stands for the number of control variables z (1987). For specific mathematical details, the reader is referred to the original work of Grossmann and Floudas (1987), and the more recent works by Ochoa and Grossmann (2020) and Pulsipher et al. (2019).

Although there are several options to describe the set $T(\delta)$, in this work, we restrict our approach and the discussed case studies to a rectangular form of $T(\delta)$. Therefore, the constraint $\theta \in T(\delta)$ given in equation (5.10) can be expressed by two inequality constraints shown in equation (5.15). The reader is referred to the work of Pulsipher et al. (2019), which addresses the case of an ellipsoidal form of $T(\delta)$.

$$\begin{aligned} \theta_k^N - \delta\Delta\theta_k^{min} &\leq \theta_k \\ \theta_k &\leq \theta_k^N + \delta\Delta\theta_k^{max} \end{aligned} \tag{5.15}$$

Using the above-shown reformulation techniques and assumptions, the reformulated flexibility index problem can be expressed as shown in equation (5.16).

$$
\begin{aligned}
& FI = \min_{\delta} \delta \\
& \text{s.t.} f_j\left(d, z, \theta\right) - u + s_j = 0, \quad \forall j \in J \\
& \quad \sum_j \lambda_j = 1 \\
& \quad \sum_j \lambda_j \frac{\partial f_j\left(d, z, \theta\right)}{\partial z_n} = 0, \quad \forall n \in N \\
& \quad s_j \leq M\left(1 - y_j\right), \quad \forall j \in J \\
& \quad \lambda_j \leq y_j, \quad \forall j \in J \\
& \quad \sum_j y_j \leq |N| + 1 \\
& \quad \theta \in T\left(\delta\right) \\
& \quad \lambda_j \geq 0, \quad \forall j \in J \\
& \quad s_j \geq 0, \quad \forall j \in J \\
& \quad \delta \geq 0
\end{aligned}
\tag{5.16}
$$

5.3.2 Flexibility index formulation with complicating constraints

As already said, here we define as complicating constraints those that are either hard to model explicitly or lead to complex expressions hard to handle numerically. Such a situation might arise, for example, in dynamic systems with constraints on temporal profiles, or in process models with complex unit operations whose behavior is hard to model mechanistically. In the former case, discretization methods such as orthogonal collocation (Carey & Finlayson, 1975) might be applied, but this will result in complex models posing numerical challenges (i.e., convergence problems, entrapment in low-quality local optima, etc.). On the contrary, by non-complicating constraints, we mean constraints that are directly accessible as standard algebraic expressions. To be able to use the flexibility index formulation in equation (5.16), we will follow the approach visualized in Figure 5.1. Therefore, we introduce the two proper subsets $G \subset J$ and $H \subset J$ for the non-complicating and complicating constraints, respectively. With this, the original flexibility index problem given in equation (5.16) is reformulated as given by (5.17), while inheriting the assumptions of (5.16). As stated in Section 5.2, $\hat{f}_g, g \in G$ represent the non-complicating constraints, whereas the complicating constraints are denoted by $\widetilde{f}_h, h \in H$. Due to the introduction of the two subsets

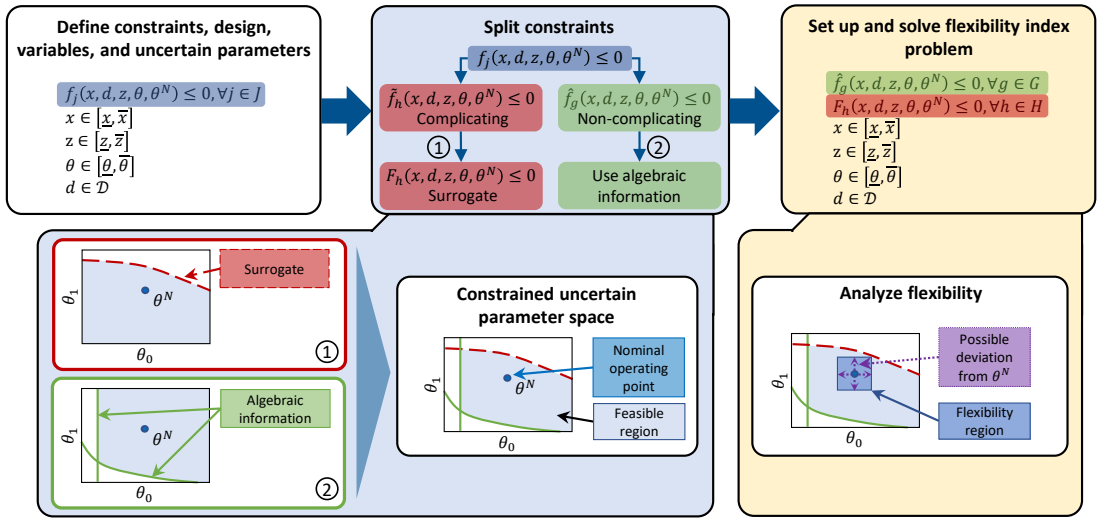

Figure 5.1. Overview of the discussed procedure in Sections 5.3.2 and 5.3.3. After the constraints f_j are defined (top left white box), which are subsequently split into complicating (red, $\tilde{f}_h$) and non-complicating (green, $\hat{f}_g$) constraints (top central box with blue background). In step 1, the complicating constraints are approximated by using surrogate models. In step 2, the available algebraic information about the non-complicating constraints are used (lower boxes in blue background). Last, the information is combined to solve the flexibility index problem (right boxes with yellow background).

G and H, also the slack variables s_j, the Lagrange multipliers λ_j, and the binary variables y_j must be split into the two respective subsets. This requires adjusting the indices in formulations (5.16) and (5.17). It is worth mentioning that this split of J into G and H, does not alter the total number of constraints involved in the problem. As discussed in the introduction, a situation with complicating or unknown constraints was also addressed in the works by Rogers and Ierapetritou (2015a, 2015b), where the authors modelled the feasible region boundaries using surrogate models. These trained surrogates could then be used to approximate the stochastic flexibility index (Straub & Grossmann, 1993), which can consider probabilistic information. The authors overcome the challenge of not having available closed-form expressions for process constraints by using a Kriging binary classification method, which allows to iteratively approximate the feasible region. With the trained classification models, the authors evaluated a range of uncertain parameter combinations and assessed if these realizations were either feasible or infeasible. However, our approach differs from these works. First and foremost, Rogers and Ierapetritou (2015a, 2015b) used their surrogate model to evaluate the stochastic flexibility index (Straub & Grossmann, 1990), which measures the probability of feasible operation, while we use the surrogate to incorporate it into

the originally proposed deterministic flexibility index formulation (I. E. Grossmann et al., 1983; Halemane & Grossmann, 1983; Swaney & Grossmann, 1985a, 1985b). Hence, we quantify the original flexibility index, which measures the maximum allowable perturbation of parameters within which the process remains feasible, so probability information is not considered in the calculations. Second, we do not use any classification approach, but rather a regression approach. The output of the surrogate model in our work is a continuous variable that determines the value of the constraint for given values of the decision variables and parameters. Third, instead of approximating the entire feasible region with the surrogate, we only approximate individual complicating constraints, while keeping the non-complicating constraints in the formulation. To solve formulation (5.17), the complicating constraints $\widetilde{f}_h$ and their respective derivatives will be replaced by algebraic surrogate models, as discussed next. In this manner, the structure of the original flexibility index problem is kept.

$$
\begin{aligned}
& FI = \min_{\delta} \delta \\
& \text{s.t.} \hat{f}_g\left(d, z, \theta\right) - u + s_g = 0, \quad \forall g \in G \\
& \quad \widetilde{f}_h\left(d, z, \theta\right) - u + s_h = 0, \quad \forall h \in H \\
& \quad \sum_g \lambda_g + \sum_h \lambda_h = 1 \\
& \quad \sum_g \lambda_g \frac{\partial \hat{f}_g\left(d, z, \theta\right)}{\partial z_n} + \sum_h \lambda_h \frac{\partial \widetilde{f}_h\left(d, z, \theta\right)}{\partial z_n} = 0, \quad \forall n \in N \\
& \quad s_g \leq M\left(1 - y_g\right), \quad \forall g \in G \\
& \quad s_h \leq M\left(1 - y_h\right), \quad \forall h \in H \\
& \quad \lambda_g \leq y_g, \quad \forall g \in G \\
& \quad \lambda_h \leq y_h, \quad \forall h \in H \\
& \quad \sum_g y_g + \sum_h y_h \leq |N| + 1 \\
& \quad \theta \in T\left(\delta\right) \\
& \quad \lambda_g \geq 0, \quad \lambda_h \geq 0, \quad \forall g \in G, h \in H \\
& \quad s_g \geq 0, \quad s_h \geq 0, \quad \forall g \in G, h \in H \\
& \quad \delta \geq 0
\end{aligned}
\tag{5.17}
$$

5.3.3 Incorporation of algebraic surrogate models for the complicating constraints

We include algebraic models substituting the complicating constraints to solve the flexibility formulation shown in equation (5.16) or (5.17) using state-of-the-art deterministic solvers. To be able to use off-the-shelf optimization solvers, we follow the procedure described in Section 5.2 and graphically summarized in Figure 5.1. The first step is the identification of complicating constraints. These constraints, described by $\widetilde{f}_h, h \in H$, are then separated from the other non-complicating constraints as shown in Figure 5.1. Once separated, we use a surrogate model approximation, F_h, as a simplification for the complicating constraints $\widetilde{f}_h$. The original flexibility index problem in equation (5.17) is therefore reformulated into the hybrid expression (5.18) that combines the main backbone of the flexibility index problem with a data-driven surrogate model defined for the complicating constraints, as shown below.

$$
\begin{aligned}
& FI = \min_{\delta} \delta \\
& \text{s.t.} \hat{f}_g\left(d, z, \theta\right) - u + s_g = 0, \quad \forall g \in G \\
& \quad F_h\left(d, z, \theta\right) - u + s_h = 0, \quad \forall h \in H \\
& \quad \sum_g \lambda_g + \sum_h \lambda_h = 1 \\
& \quad \sum_g \lambda_g \frac{\partial \hat{f}_g\left(d, z, \theta\right)}{\partial z_n} + \sum_h \lambda_h \frac{\partial F_h\left(d, z, \theta\right)}{\partial z_n} = 0, \quad \forall n \in N \\
& \quad s_g \leq M\left(1 - y_g\right), \quad \forall g \in G \\
& \quad s_h \leq M\left(1 - y_h\right), \quad \forall h \in H \\
& \quad \lambda_g \leq y_g, \quad \forall g \in G \\
& \quad \lambda_h \leq y_h, \quad \forall h \in H \\
& \quad \sum_g y_g + \sum_h y_h \leq |N| + 1 \\
& \quad \theta \in T\left(\delta\right) \\
& \quad \lambda_g \geq 0, \quad \lambda_h \geq 0, \quad \forall g \in G h \in H \\
& \quad s_g \geq 0, \quad s_h \geq 0, \quad \forall g \in G, h \in H \\
& \quad \delta \geq 0
\end{aligned}
\tag{5.18}
$$

As visible in equation (5.18), the complicating constraint $\widetilde{f}_h\,(d, z, \theta)$ was replaced by an adequate surrogate model $F_h(d, z, \theta)$. Other than that, expression (5.18) does not differ from expression (5.17).

5.3.4 Surrogate model building

This subsection explains the individual steps involved in the surrogate model generation in detail. The model building follows a similar procedure as described in a previous work by the authors (Forster, Vázquez, & Guillén-Gosálbez, 2023a), where a situation is assumed that a mapping of the uncertain parameters to the process response is possible. First, $\widetilde{f}_h$ is evaluated at different points. Second, SR tools are applied to define a constraint F_h from a closed-form algebraic surrogate model that fits the generated data points precisely (i.e., F_h approximates the given process constraint $\widetilde{f}_h$ accurately). Last, the performance of the obtained surrogate model is assessed by suitable metrics.

Step 1: Data generation

A schematic overview of the data generation and model-building process is given in Figure 5.2. We simulate the desired case study in Python by changing some independent variables (degrees of freedom) and observing the response of the dependent variables. To map these independent variables (also called the features of the model) to the observed response (also called the target of the model), we describe the feature vector $w_i = [z, \theta]$, where $i \in I$ refers to the set of samples. The feature vector ω_i consists of the control variables $z_n, n \in N$ and the uncertain parameters $\theta_k, k \in K$. The target vector is denoted as $\widetilde{f}_h(w_i)$, or $\widetilde{f}_{h,i}$ in short. Therefore, the sampling matrix is generated with the desired number of samples $|I|$ using, without generality loss, the Latin hypercube sampling (LHS).

The resulting dataset I is split into two proper subsets as shown in equation (5.19):

$$\begin{aligned} I &= I^{TR} \cup I^{TE} \\ I^{TR} &\cap I^{TE} = \emptyset \end{aligned} \tag{5.19}$$

Where I^{TR} and I^{TE} represent the training and test subsets, respectively. The training subset is later used for model training, whereas the test subset is used for model testing.

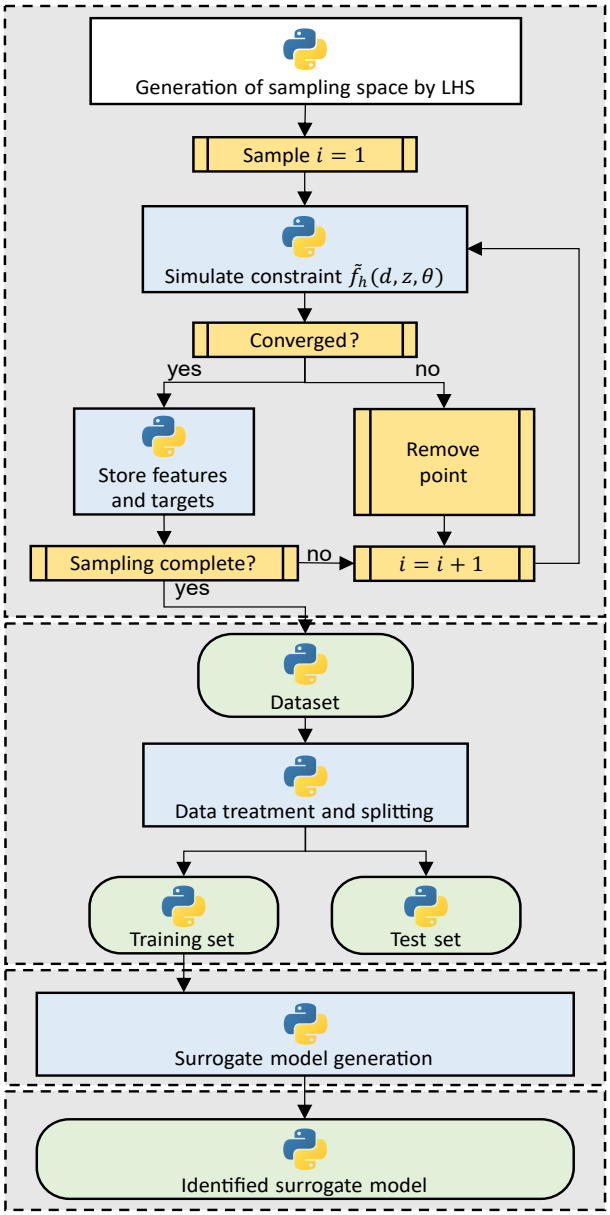

Figure 5.2. Schematic representation of the data generation and surrogate model building procedure.

Step 2: Surrogate model building

After having prepared the data, we want to find an expression in form of a surrogate model $F_h\,(d, z, \theta)$ that accurately maps the above-described feature vector ω_i to the corresponding targets $\widetilde{f}_{h,i}$. Herein, since we apply an SR algorithm, we do not need to define any aprioristic assumption on the structure for $F_h(d, z, \theta)$. As mentioned, SR aims to find a suitable mathematical expression for the observed data by representing the appropriate expressions in a symbolic tree. An example of such a search is schematically shown in Figure 5.3.

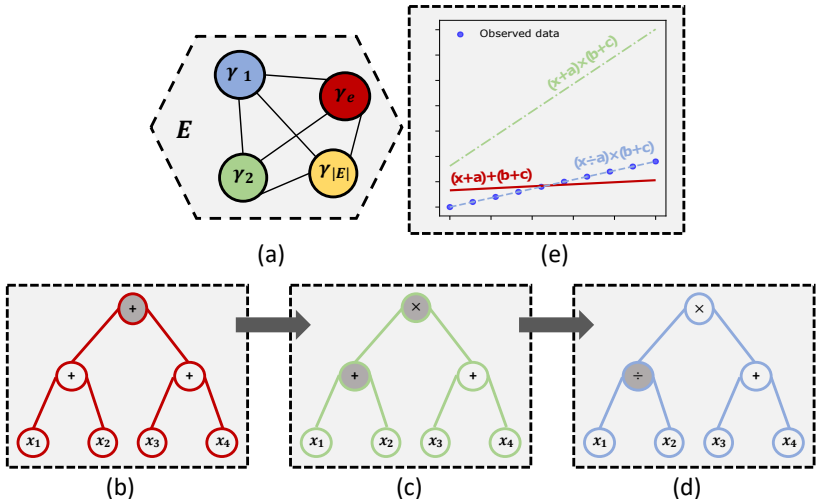

Figure 5.3. (a) The space E of all possible expressions γ_e is schematically shown as a dashed polygon. (b) A representation of an initial mathematical expression $\gamma\,(x) = (x_1 + x_2) + (x_3 + x_4)$ as a symbolic tree (red). (c) A root node replacement is performed (grey node) to reach the green symbolic expression in (c). Performing another node replacement (grey node), the blue symbolic tree is reached (d), representing $\gamma\,(x) = (x_1/x_2) \times (x_3 + x_4)$. The fitting visualization of the three expressions is shown in (e), together with the observed data as circles.

Figure 5.3 (a) visualizes the space of all possible mathematical expressions γ, which is described by E. Starting from one symbolic tree representation $\gamma_e, e \in E$, we perform changes in the tree that lead to different mathematical expressions. One example of such a tree evolution is shown in Figure 5.3 (node replacement). Another adaptation would be the elementary tree replacement (i.e., exchanging the complete sub-tree (x_3+x_4) by another sub-tree). By doing this tree evolution, a defined performance metric can be calculated for each resulting expression. This metric should quantify how well the expression fits the observed data. The SR algorithm then proceeds to search the space of expressions, seeking the expression with the best goodness of fit. This search is stochastic, as in other evolutionary

algorithms (Costa & Oliveira, 2001; Guimerà et al., 2020).

As mentioned in the introduction, several SR algorithms are available to identify algebraic surrogates. Without loss of generality, we use the approach developed by Guimerà et al. (2020), the BMS, to simplify the complicating constraints $\widetilde{f}_h(d, z, \theta)$. The BMS uses statistical prior information about the mathematical operations in the equations, and it is straightforward to implement, working out-of-the-box and allowing interconnection with the Python environment without need of extensive coding. This easy implementation facilitates its application in different fields and case studies. Moreover, we note that the BMS was already successfully applied to build process models (Forster, Vázquez, & Guillén-Gosálbez, 2023a; Jog et al., 2023; Negri et al., 2022). The BMS can provide closed-form algebraic expressions from data based on a set of user-defined mathematical operations (i.e., addition, subtraction, multiplication, etc.). We next provide a high-level overview of how the BMS works. For further information, the reader is referred to the original paper (Guimerà et al., 2020).

A conditional probability $p(\gamma_e|D)$ is assigned to each expression γ_e that is used to fit some data D. This probability is calculated according to Bayes Theorem (Bishop, 2006; Murphy, 2013):

$$p\left(\gamma_e|D\right) = \frac{p\left(D|\gamma_e\right)\ p(\gamma_e)}{p(D)} \tag{5.20}$$

Where $p(D)$ represents the marginal likelihood of some data D. $p(D)$ is independent of γ_e and therefore only acts as a normalization constant. Marginalizing over the parameters ϕ_e associated with expression γ_e (Murphy, 2013), the numerator in equation (5.20) can be expressed as an integral over the space of all possible parameter values Φ_e (Guimerà et al., 2020). This marginalization is then descried by the description length $\mathcal{DL}\left(\gamma_e\right)$ (Guimerà et al., 2020; Hansen & Yu, 2001; Murphy, 2013):

$$\begin{aligned} \mathcal{DL}\left(\gamma_e\right) &= -\log\left[p\left(D|\gamma_e\right)\ p\left(\gamma_e\right)\right] \\ &= -\log\left[\int_{\Phi_e} p\left(D|\gamma_e,\ \phi_e\right)\ p\left(\phi_e|\gamma_e\right)\ p\left(\gamma\right) d\phi\right] \end{aligned} \tag{5.21}$$

The computation of this integral is challenging (Guimerà et al., 2020; Murphy, 2013). In literature it is stated (Grünwald, 2007; Murphy, 2013) that under certain assumptions, the description length can be approximated through the Bayesian information criterion (BIC) and the prior of the corresponding symbolic expression γ_e:

$$\mathcal{DL}\left(\gamma_e\right) \approx \frac{BIC(\gamma_e)}{2} - \log\left(p\left(\gamma_e\right)\right) \tag{5.22}$$

The description length, and, therefore, this final equation can be interpreted as the plausibility of observing an expression γ_e, conditioned on some data D. According to Grünwald (2007), $\mathcal{DL}(\gamma_e)$ can also be understood as an encoded length of the expression γ_e (number of natural units).

In the applied SR approach (Guimerà et al., 2020), a Markov-chain Monte Carlo (MCMC) (Hastings, 1970) algorithm is used to explore the space E of expressions, where the number of MCMC iterations is defined by the user. After evaluating the description length of each expression $\mathcal{DL}\left(\gamma_e\right)$, the BMS keeps the most plausible one, representing the expression with the shortest description length (the best goodness-of-fit).

5.3.5 Surrogate model performance

The performance of the surrogate model is assessed by calculating several metrics for both the training and test data sets, S^{TR} and S^{TE}. Here, to this end, the root mean squared error ($RMSE$), mean absolute error (MAE), and the coefficient of determination (R^2) were used:

$$\begin{aligned} RMSE &= \sqrt{\frac{1}{n}\sum_{a\in A}\left(\widetilde{f}_h\left(w_i\right) - F_h\left(w_i\right)\right)^2} \\ MAE &= \frac{1}{n}\sum_{a\in A}\left|\widetilde{f}_h(w_i) - F_h(w_i)\right| \\ R^2 &= 1 - \frac{SSR}{SST} = 1 - \frac{\sum_{a\in A}\left(F\left(w_i\right) - \widetilde{f}_h\left(w_i\right)\right)^2}{\sum_{a\in A}\left(\widetilde{f}_h\left(w_i\right) - \mu_{\widetilde{f}_h}\right)^2} \end{aligned} \tag{5.23}$$

in equation (5.23), the predictions by the model are described by $F_h(w_i)$ using the

given input vector w_i of one sample i. The observed response $\widetilde{f}_h$ and the mean of the observed process responses are described by $\widetilde{f}_h(w_i)$ and $\mu_{\widetilde{f}_h}$, respectively. As already mentioned, both the model predictions $F_h(w_i)$ and the observed response $\widetilde{f}_h(w_i)$ are calculated by using input data from the training or test set. Variables SSR and SST denote the sum of squares of residuals and the total sum of squares (proportional to the variance of the data), respectively. In addition to these performance metrics, the time required for both the model training and for solving the flexibility index problem is reported as a central processing unit (CPU) time. Lastly, both the solver and model status are reported.

5.3.6 Software implementation

All calculations were carried out on an Intel®Core™i7-8700 CPU and 16 GB of RAM. We used Python v3.10 with NumPy v1.23.5, SciPy v1.9.3, and pyDOE v0.3.8 to construct the sampling dataset. The algorithm provided by Guimerà et al. (2020) was used to train the BMS. The symbolic equation generated by the BMS was incorporated into the flexibility index problem, which was solved using Pyomo (Bynum et al., 2021; Hart et al., 2011) v6.4.4 interfacing with the solver BARON (Sahinidis, 1996) v22.7.23.

5.4 Case studies

Before introducing the case studies (CS), the reader is referred to the supporting information in Section D.1, where three motivational examples are provided. These examples should give an intuition how the flexibility index calculations are performed in easily visualizable examples. The first two motivational examples do not include control variables, where the third one does.

We then apply the hybrid flexibility approach discussed above to two industrially relevant CS. The first covers a protein-A chromatographic process; the second is a bioprocess in fed-batch operation mode. The CS and corresponding data generation processes are described in the following.

5.4.1 Fed-batch bioreactor for ethanol production (CSI)

We consider a bioreactor in a fed-batch operation mode. The model was taken from the dynamic optimization examples demonstrated on APmonitor.com (Hedengren et al., 2014). A schematic representation of the reactor is given in Figure 5.4.

The reactor is equipped with a liquid feed, an air supply (with a submerged aerator), a heating/cooling jacket, and a temperature probe inside the reactor. In the reactor, microorganisms grow and produce ethanol by consuming oxygen and glucose. To describe the dynamic evolution of the species, the system of ODEs given in expressions (5.26)-(5.42) is used together with the corresponding parameters indicated in Tables 5.1 and 5.2. One major goal is to that the final ethanol concentration reaches at least a user-defined lower bound $\underline{E}$. The control variable here is the temperature of the cooling agent, that is, $z = T_c$. Furthermore, the uncertain parameters θ are the glucose concentration in the feed (S_{in}) and the temperature within the reactor (T). The constraints of this problem can therefore be formulated as given in expression (5.24).

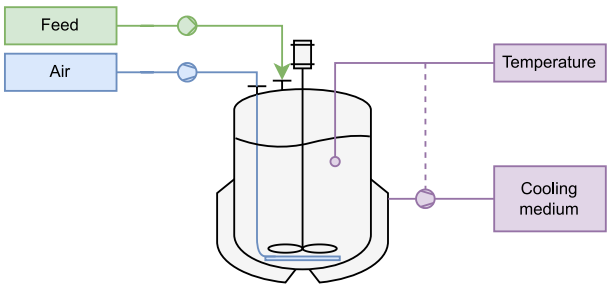

Figure 5.4. Schematic representation of a bioreactor used in case study I.

The ethanol concentration needs to be assessed, which is not straightforward. We add the first constraint to the set of complicating constraints $H = \{1\}$, namely, we define $\widetilde{f_1} = f_1 = \underline{E} - E$. The other constraints are added to the set of non-complicating constraints $G = \{2, 3, 4, 5, 6, 7\}$. As mentioned above, this could for example be done by discretizing the differential equations appropriately (i.e., by applying orthogonal collocation on finite elements). However, one disadvantage is that the dimensionality of the resulting optimization problem would be very large due to the addition of many auxiliary variables (Carey & Finlayson, 1975; Guillén-Gosálbez et al., 2013).

$$
\begin{aligned}
f_1 &: \underline{E} - E \leq 0 & f_2 &: \underline{T_c} - T_c \leq 0 \\
f_3 &: T_c - \overline{T_c} \leq 0 & f_4 &: \underline{S_{in}} - S_{in} \leq 0 \\
f_5 &: S_{in} - \overline{S_{in}} \leq 0 & f_6 &: \underline{T} - T \leq 0 \\
f_7 &: T - \overline{T} \leq 0 & J &:= \{1, 2, 3, 4, 5, 6, 7\}
\end{aligned}
\tag{5.24}
$$

To circumvent such possible limitations, the ethanol concentration at the reactor outlet shall be modeled with the BMS. Therefore, $F(S_{in}, T, T_c)$ represents a trained BMS model that maps the features S_{in}, T, and T_c to the final ethanol concentration E in the reactor. Hence, the constraint $\widetilde{f_1} = \underline{E} - E$ is reformulated by using a closed-form algebraic expression, leading to $F_1 = \underline{E} - F(S_{in}, T, T_c)$. The formulation in equation (5.25) then provides the entire reformulated problem. It is worth mentioning again that $\widetilde{f_1}$ describes the original complicating constraint, whereas F_1 describes the reformulated complicating constraint where a surrogate equation is included to facilitate the calculations.

$$
\begin{aligned}
&f_1 : \underline{E} - F(S_{in}, T, T_c) \leq 0 && f_2 : \underline{T_c} - T_c \leq 0 \\
&f_3 : T_c - \overline{T_c} \leq 0 && f_4 : \underline{S_{in}} - S_{in} \leq 0 \\
&f_5 : S_{in} - \overline{S_{in}} \leq 0 && f_6 : \underline{T} - T \leq 0 \\
&f_7 : T - \overline{T} \leq 0 && H \coloneqq \{1\}, G \coloneqq \{2,3,4,5,6,7\}
\end{aligned}
\tag{5.25}
$$

Specific growth rate

$$\mu = \mu_{max} \frac{S}{K_S X + S} \frac{O_{liq}}{K_{OX} + O_{liq}} \left(1 - \frac{E}{E_{max}}\right) \frac{1}{1 + \exp(-(100 - S))} \tag{5.26}$$

$$\mu_{max} = [(a_1(T - k_1))(1 - \exp(b_1(T - k_2)))]^2$$

$$E_{max} = E_{max,b} + \frac{E_{max,T}}{1 - \exp(-b_2(T - k_3))}$$

$$q_E = a_E \mu + b_E$$

Non-growth ethanol products

$$b_E = c_1 \exp\left(\frac{-A_{P1}}{T}\right) - c_2 \exp\left(\frac{-A_{P2}}{T}\right) \tag{5.27}$$

Ethanol consumption

$$q_S = \frac{\mu}{Y_{XS}} + \frac{q_E}{Y_{ES}} \tag{5.28}$$

Oxygen consumption

$$q_O = \frac{q_{O,max}}{Y_{XO}} \frac{O_{liq}}{K_{OX} + O_{liq}} \tag{5.29}$$

Biomass deactivation

$$K_d = K_{db} + \frac{K_{dT}}{1 + \exp(-b_3(T - k_4))} \tag{5.30}$$

Oxygen saturation

$$O_{sat} = Z \frac{O_{gas} RT}{K_H} \tag{5.31}$$

Oxygen mass transfer

$$k_l a = (k_l a)_0 (1.2)^(T - 20) \tag{5.32}$$

Total volumes

$$V = V_l + V_g \tag{5.33}$$

Liquid volume

$$\frac{dV_l}{dt} = Q \tag{5.34}$$

Total biomass

$$\frac{dX_t}{dt} = \mu X_v + \frac{Q}{V_l}(X_{t,in} - X_t) \tag{5.35}$$

Viable biomass

$$\frac{dX_v}{dt} = (\mu - K_d) X_v + \frac{Q}{V_l}(X_{v,in} - X_v) \tag{5.36}$$

Glucose

$$\frac{dS}{dt} = \frac{Q}{V_l}(S_{in} - S) - q_S X_v \tag{5.37}$$

Ethanol

$$\frac{dE}{dt} = \frac{Q}{V_l}(E_{in} - E) + q_E X_v \tag{5.38}$$

Liquid oxygen

$$\frac{dO_l iq}{dt} = \frac{Q}{V_l}(O_{sat} - O_{liq}) + k_l a(O_{sat} - O_{liq}) - q_O X_v \tag{5.39}$$

Gas oxygen

$$\frac{dO_g as}{dt} = \frac{F_a ir}{V_g}(O_{gas,in} - O_{gas}) - \frac{V_l k_l a}{V_g}(O_s at - O_l iq) + (O_g asQ)/V_g \tag{5.40}$$

Temperature

$$\frac{dT}{dt} = \frac{Q}{V_l(T_{in} - T)} - \frac{T_{ref}}{V_l} Q + q_O X_v \frac{\Delta H}{MW_o \rho C_{p,br}} - \frac{K_T A_T (T - T_c)}{V_l \rho C_{p,br}} \tag{5.41}$$

textCoolingagent

$$\frac{dT_c}{dt} = \frac{F_c}{V_{cj}}(T_{c,in} - T_c) + \frac{K_T A_T (T - T_c)}{V_{cj} \rho_c C_{p,c}} \tag{5.42}$$

Table 5.1. Parameters used in the ordinary differential equation system given in expressions (5.26)-(5.42). Continued in Table 5.2.

Parameter	Physical meaning	Value	Unit
a_1	Ratkowsky parameter	0.05	$^\circ C^{-1}\,h^{-0.5}$
a_E	Growth-associated parameter for ethanol production	4.5	$[-]$
A_{P1}	Activation energy parameter for ethanol production 1	6	$^\circ C$
A_{P2}	Activation energy parameter for ethanol production 2	20.3	$^\circ C$
b_1	Exponential scaling parameter for the maximum specific growth rate	0.035	$^\circ C^{-1}$
b_2	Exponential scaling parameter for the growth inhibitory ethanol concentration	0.15	$^\circ C^{-1}$
b_3	Exponential scaling parameter for the specific death rate	0.4	$^\circ C^{-1}$
c_1	Constant decoupling factor for ethanol production	0.38	$g_E\,g_X^{-1}\,h^{-1}$
c_2	Constant decoupling factor for ethanol production	0.29	$g_E\,g_X^{-1}\,h^{-1}$
k_1	Parameter in the maximum specific growth rate	3	$^\circ C$
k_2	Parameter in the maximum specific growth rate	55	$^\circ C$
k_3	Parameter in the growth-inhibitory ethanol concentration expression	60	$^\circ C$
k_4	Temperature at the inflection point of the specific death rate sigmoid curve	50	$^\circ C$
$E_{max,b}$	Temperature-independent product inhibition constant	90	$g\,L^{-1}$
$E_{max,T}$	Maximum value of product inhibition constant due to temperature	90	$g\,L^{-1}$
K_{db}	Basal specific cellular biomass death rate	0.025	h^{-1}
K_{dT}	Maximum value of specific cellular biomass death rate due to temperature	30	h^{-1}
K_{SX}	Glucose saturation constant for the specific growth rate	5	$g\,L^{-1}$

Table 5.2. Parameters used in the ordinary differential equation system given in expressions (5.26)-(5.42). Continued from Table 5.1.

Parameter	Physical meaning	Value	Unit
K_{OX}	Oxygen saturation constant for the specific growth rate	0.0005	$\mathrm{g\,L^{-1}}$
$q_{O,max}$	Maximum specific oxygen consumption rate	0.05	$\mathrm{h^{-1}}$
Y_{ES}	Theoretical yield of ethanol on glucose	0.51	$\mathrm{g_E\,g_S^{-1}}$
Y_{XO}	Theoretical yield of biomass on oxygen	0.97	$\mathrm{g_X\,g_O^{-1}}$
Y_{XS}	Theoretical yield of biomass on glucose	0.53	$\mathrm{g_X\,g_S^{-1}}$
$C_{p,br}$	Heat capacity of the mass of reaction	4.18	$\mathrm{J\,g^{-1}\,^\circ C^{-1}}$
$C_{p,c}$	Heat capacity of the cooling agent	4.18	$\mathrm{J\,g^{-1}\,^\circ C^{-1}}$
ΔH	Heat of reaction of fermentation	518000	$\mathrm{J\,mol_O^{-1}}$
T_{ref}	Reference temperature	20	$^\circ\mathrm{C}$
K_H	Henry's constant for oxygen in the fermentation broth	200	$\mathrm{atm\,L\,mol^{-1}}$
Z	Oxygen compressibility factor	0.792	[–]
R	Ideas gas constant	0.082	$\mathrm{atm\,L\,mol^{-1}\,^\circ C^{-1}}$
$(k_l a)_0$	Temperature-independent volumetric oxygen transfer coefficient	100	$\mathrm{h^{-1}}$
K_T	Heat transfer coefficient	360000	$\mathrm{J\,h^{-1}\,m^{-2}\,^\circ C^{-1}}$
ρ	Density of the fermentation broth	1080	$\mathrm{g\,L^{-1}}$
ρ_c	Density of the cooling agent	1000	$\mathrm{g\,L^{-1}}$
MW	Molecular weight of oxygen	32	$\mathrm{g\,mol^{-1}}$

The goal of the flexibility analysis is to quantify and identify the largest possible uncertainty set $\theta \in T(\delta)$, such that the process is still feasible over the entire range of θ. In other words, one should assess how far the glucose inlet concentration and the reactor's temperature can deviate from the nominal operating point such that the process is still feasible (all the constraints still hold).

To find $\mathcal{F}$, the ODE system given in expressions (5.26)-(5.42) was solved for different feature vectors $\omega_i = [S_{in}, T, T_c], i \in I$ with $|I| = 250$ samples using the explicit Runge-Kutta method of order 5 (Dormand & Prince, 1980). After simulating for each ω_i, the final ethanol concentration E was obtained. The sampling procedure discussed above (Figure 5.2) was applied, where the upper and lower bounds selected for the LHS are displayed in Table 5.3. The resulting dataset A was randomly split to $|I^{TR}| = 200$ training (80%) and $|I^{TE}| = 50$ testing (20%) samples.

To train the BMS, several unary ($\exp(x)$, $\log(x)$, x^2, x^3, $\sqrt{x}$) and binary ($+$, $-$, $\div$, $\times$, x^y) operators were allowed to be selected. In addition, the number of MCMC steps was fixed to $15 \cdot 10^3$. The model was allowed to contain up to eight parameters.

5.4.2 Protein-A affinity chromatography (CSII)

This case study consists of a loading process of antibodies onto a protein-A affinity chromatographic column. A schematic representation of the different steps in chromatography is given in Figure 5.5. First, the column is packed with the desired material (resin). Second, an equilibration is performed, which makes the column ready to be deployed. During the loading phase, the antibody mixture is added to the top of the column. Depending on the loading time (t_{load}), the antibody concentration in the feed (c_{in}), and the flowrate (Q), some of the product might be lost. Subsequently, the washing step is used to collect the desired product.

Table 5.3. Upper and lower bounds for the features S_{in}, T, and T_c. The bounds were used to create the samples for case study I by applying a Latin hypercube sampling structure.

Feature	Lower bound	Upper bound	Unit
S_{in}	0	20	$\mathrm{g\,L^{-1}}$
T	15	35	°C
T_c	20	40	°C

The elution step terminates the entire operation.

$$\frac{\partial c}{\partial t} = -\frac{Q}{A_{col}\epsilon}\frac{\partial c}{\partial x} + D^{app}\left(\frac{\partial^2 c}{\partial x^2}\right) - \zeta\frac{\partial q}{\partial t} \tag{5.43}$$

$$D^{app} = \hat{V}\left(\frac{d_p}{2}\right)\frac{Q}{A_{col}\epsilon} \tag{5.44}$$

$$\frac{\partial q}{\partial t} = k_m(q^* - q) \tag{5.45}$$

$$q^* = \frac{Hc}{1 + \dfrac{Hc}{q_{sat}}} \tag{5.46}$$

$$k_m = k_{max}\left(C_1 + (1 - C_1)\left(1 - \frac{q}{q_{sat}}\right)_2^C\right) \tag{5.47}$$

$$\left[\frac{\partial c}{\partial x}\right]_{x=L} = 0 \quad \text{and} \quad \left[\frac{\partial q}{\partial x}\right]_{x=L} = 0 \tag{5.48}$$

$$c(t=0) = c_0 \quad \text{and} \quad q(t=0) = q_0 \tag{5.49}$$

$$LR = \frac{\int_0^{t_{load}} c(x,t)dt}{\int_0^{t_{load}} c_{in}dt} \leq \overline{LR} \tag{5.50}$$

$$\underline{Q} \leq Q \leq \overline{Q} \tag{5.51}$$

$$\underline{c_{in}} \leq c_{in} \leq \overline{c_{in}} \tag{5.52}$$

$$\underline{t_{load}} \leq t_{load} \leq \overline{t_{load}} \tag{5.53}$$

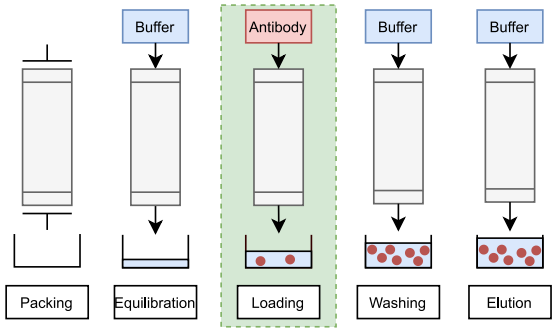

Figure 5.5. Schematic representation of the five different steps in a chromatographic procedure. The loading phase (marked by the dashed green area) is the step of interest for this case study.

We focus exclusively on the loading phase of the entire procedure. The loss ratio (LR) is the relationship between the mass of the leaked product relative to the total amount of proteins fed. With this, the deterministic constraints of the problem can be formulated as given in equations (5.43)-(5.53), which was adapted from (Baur et al., 2016), where the corresponding parameters were taken from the same work (Baur et al., 2016) and Ding and Ierapetritou (2021).

The system given in equations (5.43)-(5.49) describes the partial differential equations (PDE) for the dynamic evolution of the concentration profiles, which can be expressed in terms of concentration in the liquid phase (c) and in the adsorbed phase (q). The parameters of the PDE system are given in Table 5.4.

Q, c_{in}, and t_{load} are the adjustable flow rate, the inlet antibody concentration, and the loading time, respectively. Their lower and upper bounds are indicated by $\underline{Q}$, $\underline{c_{in}}$, $\underline{t_{load}}$, $\overline{Q}$, $\overline{c_{in}}$, and $\overline{t_{load}}$, respectively, which are represented in equation (5.51)-(5.53). LR represents the loss rate, which is the relationship between the mass of leaked product relative to the total amount of product fed during the loading phase. $\overline{LR}$ is a user-defined upper bound for the loss rate.

The entire system in expressions (5.43)-(5.53) can be rewritten more compactly, as shown in equation (5.54).

$$\begin{aligned} f_1 &: LR - \overline{LR} \leq 0 & f_2 &: \underline{Q} - Q \leq 0 \\ f_3 &: Q - \overline{Q} \leq 0 & f_4 &: \underline{c_{in}} - c_{in} \leq 0 \\ f_5 &: c_{in} - \overline{c_{in}} \leq 0 & f_6 &: \underline{t_{load}} - t_{load} \leq 0 \\ f_7 &: t_{load} - \overline{t_{load}} \leq 0 & J &\coloneqq \{1,2,3,4,5,6,7\} \end{aligned} \tag{5.54}$$

This entire differential system in equation (5.43)-(5.49) and the integrals in equation (5.50) are not trivial - and computationally expensive - to be incorporated into the optimization problem.

Again, one option would be to apply an appropriate discretization method to the differential equations, which would increase the problem dimensionality and potentially lead to convergence issues, as discussed in CSI. We, therefore, add the first constraint to the set of complicating constraints $H = \{1\}$, describing it by $\widetilde{f_1} = f_1 = LR - \overline{LR}$. The other constraints are added to the set of non-complicating constraints $G = \{2,3,4,5,6,7\}$. A BMS model is used that maps

the features c_{in}, t_{load}, and Q to LR. Hence, the constraint $\widetilde{f_1} = LR - \overline{LR}$ is reformulated by using a closed-form algebraic expression, leading to $F_1 = \mathcal{F}(c_{in}, t_{load}, Q) - \overline{LR}$. The entire reformulated constraints are then given by the formulations shown in equation (5.55). Again, it is worth mentioning again that $\widetilde{f_1}$ describes the original complicating constraint, whereas F_1 describes the reformulated complicating constraint including the algebraic surrogate equation.

$$\begin{aligned} f_1 :& \mathcal{F}(c_{in}, t_{load}, Q) - \overline{LR} \leq 0 & f_2 :& \underline{Q} - Q \leq 0 \\ f_3 :& Q - \overline{Q} \leq 0 & f_4 :& \underline{c_{in}} - c_{in} \leq 0 \\ f_5 :& c_{in} - \overline{c_{in}} \leq 0 & f_6 :& \underline{t_{load}} - t_{load} \leq 0 \\ f_7 :& t_{load} - \overline{t_{load}} \leq 0 & H \coloneqq& \{1\}, G \coloneqq \{2,3,4,5,6,7\} \end{aligned} \tag{5.55}$$

Here, the flexibility analysis aims to assess how far the inlet concentration of the antibody and the loading time of the column can deviate from the nominal operating point such that the process is still feasible (all the constraints still hold).

To find a suitable model for $\mathcal{F}$, the PDE system given in expessions (5.43)-(5.49) was solved for several samples ($|I| = 250$ samples) of the feature vector $\omega_i = [c_{in,a}, t_{load,a}, Q_a], i \in I$. For each run, a spatial discretization along the column length with 100 grid points was performed using a first-order central finite differences method. Subsequently, the resulting system of ordinary differential equations (ODE) was solved at each spatially discretized point using the explicit Runge-Kutta method of order 5 (Dormand & Prince, 1980). After simulating for each ω_i, the concentration profile was obtained integrating expression (5.50), and therefore a value for LR, could be numerically calculated. The number of spatial discretization points was fixed at 100. The sampling procedure discussed above (Figure 5.2) was applied, where upper and lower bounds for the LHS are displayed in Table 5.5. The resulting dataset A was randomly split into $|I^{TR}| = 200$ training (80%) and $|I^{TE}| = 50$ testing (20%) samples.

To train the BMS, several unary ($\exp(x)$, $\log(x)$, x^2, x^3, $\sqrt{x}$) and binary ($+$, $-$, $\div$, $\times$, x^y) operators were allowed to be selected. In addition, the number of MCMC steps was fixed to $20 \cdot 10^3$. The model was allowed to contain up to three parameters.

Table 5.4. Parameters used for the chromatography model discussed in case study II. The corresponding model equations shown in equation (5.43)-(5.49) were adapted from Baur et al. (2016). The parameters were taken from Baur et al. (2016) and Ding and Ierapetritou (2021).

Parameter	Physical meaning	Value	Unit
L_{col}	Column length	10	cm
A_{col}	Crossectional area of the column	0.2	cm^2
d_p	Average particle diameter	0.0044	cm
ϵ	Void fraction	0.368	[–]
$\hat{V}$	Intercept of reduced Van-Deemter equation	35.13	[–]
H	Partition coefficient	246.8	[–]
q_{sat}	Saturation concentration in the adsorbed phase	94.72	$mg\,mL^{-1}$
k_{max}	Maximum mass transfer rate	0.18	min^{-1}
C_1	Pore blockage coefficient 1	0.6245	[–]
C_2	Pore blockage coefficient 2	2.071	[–]
c_0 and q_0	Initial values of the liquid and adsorbed phases	0	$mg\,mL^{-1}$

Table 5.5. Upper and lower bounds for the features c_{in}, t_{load}, and Q. The bounds were used to create the samples for case study II by applying a Latin hypercube sampling structure.

Feature	Lower bound	Upper bound	Unit
c_{in}	0.5	2.2	$mg\,mL^{-1}$
t_{load}	Jan 60	20	min
Q	0.001	20	$mg\,mL^{-1}$

5.5 Results

5.5.1 Surrogate model generation

The results of the surrogate model training and testing for CSI and CSII are given in Table 5.6. In addition, visualizations of the model performances are shown in Figure 5.6, where predicted values are plotted against observed ones. The corresponding closed-form expressions with the highest plausibility (lowest description length), and their estimated parameters are shown in Table 5.7 and Table 5.8.

In general, both trained models can explain the variance in the data sufficiently well when considering R^2 values greater than 0.9 as acceptance criteria, which was taken as an orientative criterion based on earlier works (Forster, Vázquez, & Guillén-Gosálbez, 2023a). The BMS was run using the maximum number of MCMC iterations as the stopping criterion, as indicated in Section 5.4. This led

Table 5.6. The training performance criteria are summarized for the Bayesian machine scientist (BMS). Each row represents one case study (CS). The CPU time (in hours) needed for the model training is shown in the left part of the table. The error metrics (root mean squared error, mean absolute error, coefficient of determination) are shown for the training and testing data (format: training/testing). The identified algebraic expressions are indicated in Table 5.7, whereas the corresponding model parameters are reported in Table 5.8.

CS	CPU training	$RMSE$	MAE	R^2
I	0.8 h	0.467 / 1.811 $\mathrm{g\,L^{-1}}$	0.383 / 0.656 $\mathrm{g\,L^{-1}}$	0.996 / 0.913 [–]
II	2.7 h	0.014 / 0.012 [–]	0.009 / 0.008 [–]	0.998 / 0.998 [–]

to CPU times of 0.8 h for CSI and 2.7 h for CSII. The low discrepancy between the R^2 values of the training and testing results indicates that the BMS is well-regularized and, therefore, less prone to overfitting, which is in line with the authors' expectations (Guimerà et al., 2020).

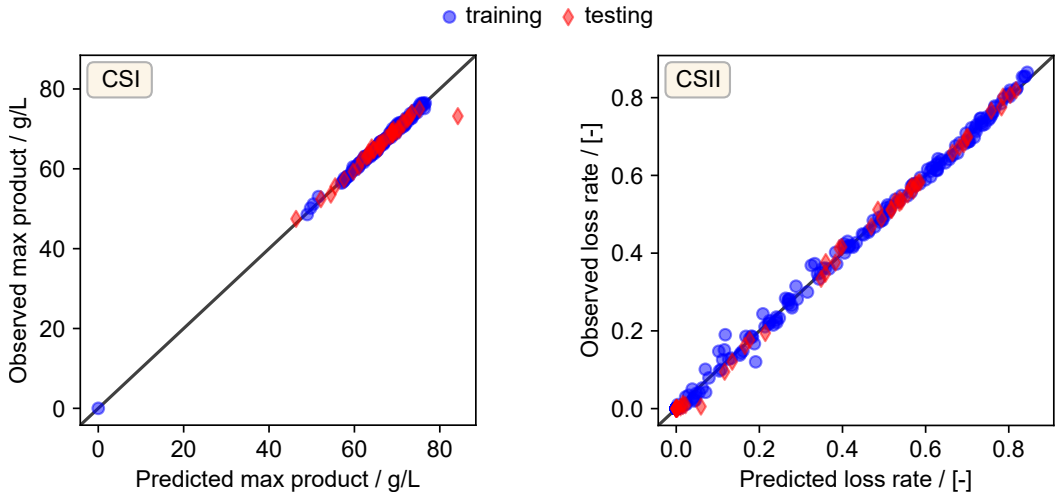

Figure 5.6. Observed vs. predicted (OVP) values for the two different case studies are shown. Blue points represent the training data, whereas red points correspond to the test data. The black line represents the values where the observed value corresponds to the model predictions.

In addition to the previous performance criterion ($R^2 > 0.9$), Figure 5.6 shows that the surrogate models perform satisfactorily both in the training and test sets, where the model responses are very close to the outcome of the theoretical models. However, what can be observed for CSII is that the risk of over or underprediction increases for low values of LR (higher spread of the training and testing points for values below around $LR = 0.5$). For CSI, one can find most data points between 45 $\mathrm{g\,L^{-1}}$ and 80 $\mathrm{g\,L^{-1}}$, where only one training sample was at 0 $\mathrm{g\,L^{-1}}$.

This point resulted from the LHS sampling and was not removed for training the BMS.

Regarding the surrogate models in Table 5.7, the BMS identified nonlinear expressions with all variables included as features. We recall that the model training considers the control variables and the uncertain parameters as features (inputs for the surrogates). This is required to adjust the control variables depending on the realization of the uncertain parameters, as done in the flexibility index problem.

The identified surrogate expressions were then incorporated into the hybrid formulation given in problem (5.18), as already discussed.

5.5.2 Incorporation of surrogate models in the flexibility index problem

The results of the case studies CSI and CSII are summarized in Table 5.9. Schematic representations of these solutions are given in Figure 5.8.

Table 5.9 shows that in both case studies, the optimal control variable z^* will be chosen at one of the bounds ($z^* = 23.0\,°\text{C}$ for CSI and $z^* = 4.0\,\text{mL/min}$ for CSII). Additionally, the first constraint F_1 was active in both cases. These are the constraints that were modeled using the BMS surrogates. Active surrogate constraints were expected, since the control variable influences those constraints. In other words, the optimizer tries to maximize the distance from the nominal operating point to a constraint. The F_1 constraints (surrogates) are influenced by the control variable. The optimizer adapts the control variable to shift the surrogate constraint away from the nominal operating point. This is done until the control variable cannot be adjusted anymore when it reaches its bound. In the chosen scenarios, the control variable impacts only the surrogate constraints with the relationship $\mathcal{F}_{1,CSI} \propto \mathcal{F}(S_{in}, T, T_c)$ in CSI and $\mathcal{F}_{1,CSII} \propto \mathcal{F}(c_{in}, t_{load}, Q)$ in CSII. A visualization of how the control variable influences the surrogate constraints is schematically given in Figure 5.7.

The resulting flexibility index δ^* can for example be used to compare two process designs in order to elucidate which one is more flexible. For example, a comparison of two different process designs for CSII is shown in the supplementary information Section D.2. By using a longer and narrower column (design d_2) compared to the

Table 5.7. The most plausible closed-form expressions for each case study (CS) identified by the Bayesian machine scientist (BMS) are shown. The corresponding estimated parameter values are reported in Table 5.8. The variable descriptions for each case study are given in Section 5.4.

CS	Prediction target	Identified expression
I	$E = E(T_c, S_{in}, T)$ $z = [T_c]$ $\theta = [S_{in}, T]$	$a_1 + \frac{a_4 + T_c}{T_c + a_7}\frac{a_3}{T}(S_{in} + a_1) + a_7 T_c^2 \frac{a_0}{((-S_{in} + a_4 a_5)a_1}$
II	$LR = LR(c_{in}, Q, t_{load})$ $z = [Q]$ $\theta = [c_{in}, t_{load}]$	$a_0 \left(\frac{t_{load}}{\left(a_0^{\frac{t_{load}/c_{in}}{a_1}}\right) t_{load} + \frac{a_1}{\exp(c_{in}^{a_0})}} \right)^{-\frac{a_2 (a_0 Q)^{a_2}}{a_0}\left(\frac{Q}{c_{in}} + a_1\right)}$

Table 5.8. Parameter values of the most plausible surrogate model identified by the Bayesian machine scientist (BMS) for each case study (CS). The corresponding model equations are given in Table 5.7.

Parameter	CS	
	I	II
a_0	1.411	0.894
a_1	66.686	24.458
a_2	1	-1.844
a_3	4.123	-
a_4	-17.508	-
a_5	-1.922	-
a_6	1	-
a_7	-17.503	-

one given in Section 5.4.2 (design d_1), the flexibility is reduced ($\delta^*_{d_1} = 0.811$ vs. $\delta^*_{d_2} = 0.389$). The result is visualized in Figure D.4. Although such visualizations as in Figure 5.7 cannot be done for higher dimensional case studies, the entire procedure can be applied in the same manner.

For both case studies, decreasing the control variable - the cooling temperature in Figure 5.7 (a) and the flow rate in Figure 5.7 (b) - will increase the size of the feasible region. Considering for example CSII, increasing the flow rate would decrease the time the antibodies would require to reach the column outlet. Therefore, a larger amount of product will be lost, which increases the loss rate during the loading phase. Keeping this fact in mind, one can observe that for lower flow rates, a higher loading time and higher antibody concentration would be possi-

Table 5.9. Results summary of the case studies CSI and CSII.

	CSI	**CSII**
θ^N	$[10.0\,\mathrm{g\,L^{-1}},\ 30.0\,^\circ\mathrm{C}]$	$[1.5\,\mathrm{mg\,mL^{-1}},\ 8.0\,\mathrm{min}]$
M	20	500
$\underline{z}$ and $\overline{z}$	$23.0\,^\circ\mathrm{C}$ and $28\,^\circ\mathrm{C}$	$4.0\,\mathrm{mL/min}$ and $12.0\,\mathrm{mL/min}$
$\Delta\theta_k^{min}, k \in K$	$[1\,\mathrm{g\,L^{-1}},\ 1\,^\circ\mathrm{C}]$	$\theta_k^N - \underline{\theta}_k$
$\Delta\theta_k^{max}, k \in K$	$[1\,\mathrm{g\,L^{-1}},\ 1\,^\circ\mathrm{C}]$	$\overline{\theta}_k - \theta_k^N$
δ^*	3.228	0.811
θ^*	$[13.23\,\mathrm{g\,L^{-1}},\ 33.23\,^\circ\mathrm{C}]$	$[2.07\,\mathrm{mg\,mL^{-1}},\ 13.67\,\mathrm{min}]$
z^*	$23.0\,^\circ\mathrm{C}$	$4.0\,\mathrm{mL/min}$
Active constraints	F_1, f_2	F_1, f_4
CPU	0.9 s	1.5 s

ble, meaning these uncertain parameters (t_{load} and c_{in}) can deviate more from a nominal operating point, making the process feasible. This manifests in the larger feasible region given in Figure 5.7 (b). Similar behavior can be observed for CSI. The surrogate model predicts a higher ethanol production with a decreased jacket temperature T_c. Therefore, the deviation on the reactor temperature and the feed concentration can be larger such that the process remains feasible, which again manifests in the higher feasible region visible in Figure 5.7 (a).

Figure 5.8 visualizes the results given in Table 5.9, where the surrogate and control constraints are active. Having chosen a nominal operating point θ^N, the optimal value of theta in the optimum is called critical theta θ^c (red circles in Figure 5.8), which indicates the scaled distance at which the process will hit the first bound. In other words, going beyond the set of parameters values θ^c ($[13.23\,\mathrm{g\,L^{-1}},\ 33.23\,^\circ\mathrm{C}]$ for CSI and $[2.07\,\mathrm{mg\,mL^{-1}},\ 13.67\,\mathrm{min}]$ for CSII), will lead to the violation of the surrogate constraint, resulting in an infeasible process.

First, the flexibility problems were solved quickly, namely, 0.9 s and 1.5 s, for CSI and CSII, respectively. Another advantage of having the algebraic surrogate comes into play when the entire problem must be adapted for any reason. If, for example, the nominal operating point has to be changed, no retraining of the surrogate model is required since the training is decoupled from the flexibility index problem. This makes the adjustment of nominal operating points or bounds very simple because the solution time of the optimization problem is within seconds.

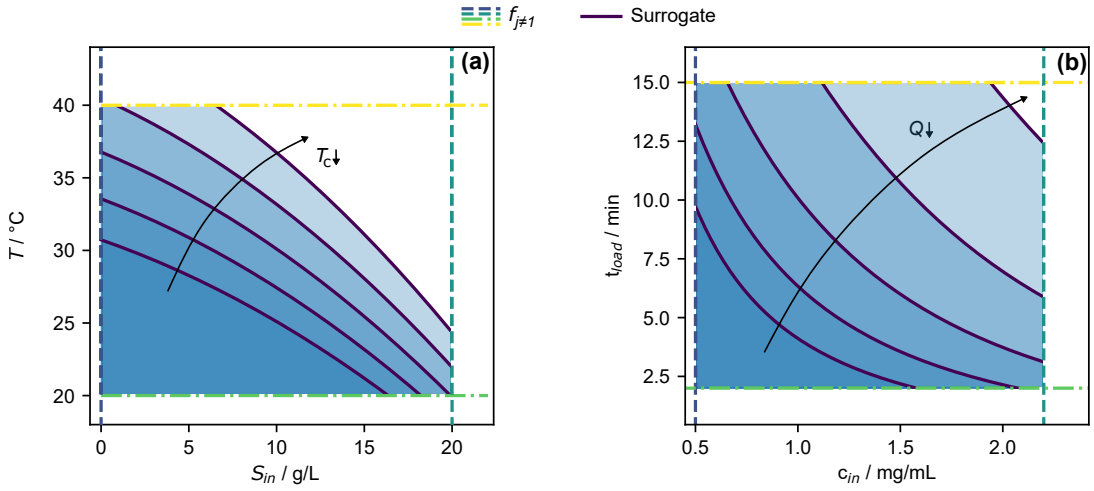

Figure 5.7. Projection of the constraints onto the uncertain parameter plane for case studies CSI (a) and CSII (b). The feasible region is shown in shaded light blue color. The constraints in dashed lines represent the bounds of the un certain parameters. The solid lines represent the surrogate constraint which can be influenced by the control variable z. Decreasing the value of z increases the size of the feasible region.

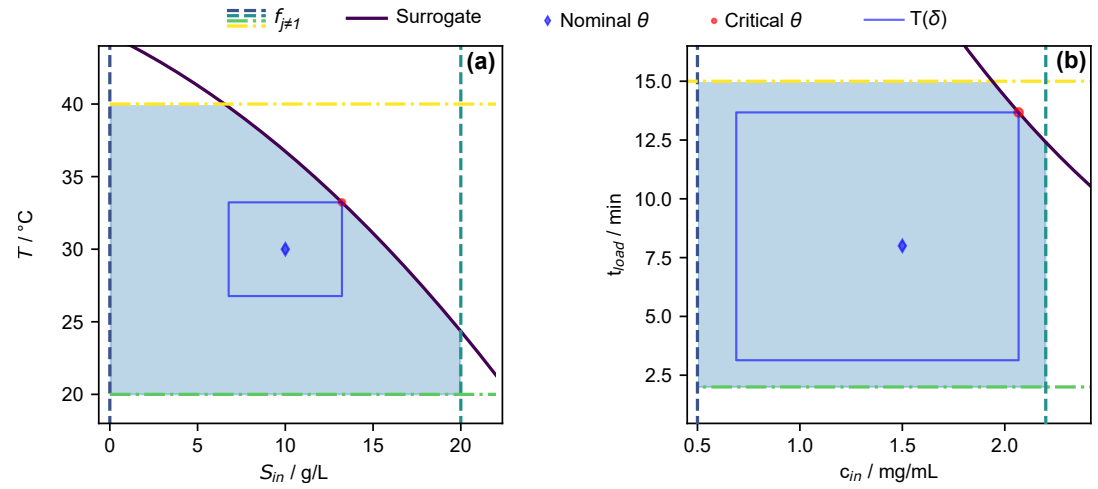

Figure 5.8. Graphical representation of the solution for the flexibility index problem for CSI (a) and CSII (b). The feasible region is shown in shaded light blue color. The constraints in dashed lines represent the bounds of the uncertain parameters. The solid lines represent the surrogate constraint which can be influenced by the control variable z. The chosen nominal operating point θ^N (blue diamond) lies within the set $T(\delta)$ (blue box). As shown in Table 5.9, the surrogate constraints F_1 are the active constraints, which is why $T(\delta)$ touches F_1 constraint (red circle).

5.6 Conclusion

This work introduced a new approach to compute the flexibility index in problems with complicating constraints. Our approach combines the originally described deterministic formulation of the flexibility index problem with a symbolically regressed surrogate model that simplifies the modeling of the complicating constraints. The symbolic regression algorithm, the BMS, assumes no aprioristic model structure, thereby enabling the accurate representation of process constraints hard to model and/or handle numerically. The resulting hybrid flexibility approach was applied to protein-A chromatography and an ethanol production process in fed-batch operation mode. The surrogate equations could accurately reproduce the complicating constraints, as evidenced by their ability to explain the data variance, making them suitable for simplifying such equations in the deterministic flexibility formulation. One drawback of the applied regression tool is the significant training time required for model building, which might be improved in the future as faster SR algorithms become available. Nevertheless, having a closed-form expression at hand pays off in several aspects: The first is that global solvers can be used, which can guarantee global optimality compared to heuristics or stochastic solvers. Additionally, the surrogate model training is decoupled from the flexibility index problem. This makes the study of different process conditions very simple because the solution time of the optimization problem is often within seconds using existing approaches to compute the flexibility index of fully analytical process models. However, our method focuses on the traditional flexibility index, so more complex flexibility metrics would require alternative methods. In the end, the most suitable approach for a given flexibility problem will depend on its features and the goal and scope of the analysis. Future work will focus on exploring alternative symbolic regression algorithms and a wider range of applications within chemical engineering and beyond.

Chapter 6

Conclusion and future research

This chapter summarizes the main contributions and insights of the thesis and reflects on the objectives given in Section 1.4. Overall, this work represents alternative methods and procedures that support advancements in the Process Systems Engineering field and its application in the context of digitalization within an industrial environment. This thesis has addressed the modeling and optimization challenges encountered in the chemical and biological sector, by leveraging advancements in machine learning, mathematical modeling, and optimization techniques. The research presented herein contributes to bridging the gap between purely mechanistic and data-driven approaches, offering hybrid frameworks that combine available knowledge with surrogate modeling and optimization techniques. Through comprehensive case studies and analyses, the effectiveness and applicability of the proposed methodologies were demonstrated, showcasing their potential to support process modeling, optimization, and flexibility analysis tasks. While the presented approaches exhibit promising results, there remains a subsequently discussed avenue for future investigations.

6.1 Conclusion and reflection on the objectives

In Chapter 2, the model building procedure for chemical reactions and bioprocesses was investigated, where a mechanistic backbone based on a canonical formalism was used to create mass balances. A mixed-integer nonlinear programming formulation to automatically identify the model structure and the values of its parameters was developed. Binary variables allowed to change the model topology and therefore made it possible to tune the complexity of the resulting system of ordinary differential equations. The model training was performed following a two-stage approach. This allowed avoiding the iterative integration of differential equations, which is usually a major reason for high computational costs in such sequential parameter estimation problems. It was found that the presented approach performed similarly to models based on artificial neural networks, even outperforming such purely data-driven methods in some cases. The presented approach further had the additional advantage of leading to models containing fewer parameters, which would also simplify a model interpretation. Hence, the proposed approach combines model identification with the ability to analyze the underlying system more easily. Building up on Chapter 2, in Chapter 3 the identification of kinetic models for bioprocesses was investigated, where no pre-defined model structure was assumed to be available. A symbolic regression algorithm

was employed to generate suitable models for the rate expressions. Similarly to Chapter 2, the model training was performed following a two-step approach, thus avoiding the iterative integration of differential equations. In this part of the thesis, different derivative approximation methods were investigated as well. Also, the influence of the number of available sampling points and the present noise was studied. It was found that, even in the case of having scarce noisy datasets available, a derivative approximation was well possible, which allowed to develop appropriate kinetic models for the systems under study. Further, the proposed method performed slightly better than artificial neural networks in most cases. One advantage – similarly as in Chapter 2 – was that the presented method led to analytical expressions that could be analyzed in detail. Such an ability to analyze the model comes, however, with the high computational cost to train the symbolically regressed closed-form expressions. The proposed approach was shown to be general enough to be applied to a wide range of applications.

The first objective of this thesis given in Section 1.4 aimed to design computational models based on available knowledge while allowing for flexible model structures and parameter estimations. As mentioned throughout Chapters 2 and 3, the development of purely mechanistic models and the subsequent parameter estimation steps for chemical and biological systems is a significant challenge due to several reasons. As summarized above, it could be shown that, by combining data, mathematical programming, and machine learning approaches, it was indeed possible to tackle these challenges. The mixed-integer nonlinear programming approach in Chapter 2 helped to combine available knowledge about a system with a parameter and model identification procedure that can be flexibly adjusted by the user. The symbolic regression algorithm in Chapter 3 generated suitable models for the rate expressions based on the statistical prior information about mathematical structures. Therefore, both methodologies tackle the mentioned objective of combining existing knowledge with flexible modeling frameworks that allow to simultaneously identify the model structure and the involved parameters. The tools provide a good balance between interpretability – due to the identified closed-form expressions – and performance. Furthermore, the presented methodologies are very flexible, and can therefore be applied to a wide range of systems, meaning, the core parts of the model identification steps – the optimization formulation for Chapter 2 and the symbolic regression for Chapter 3 – do

not significantly change from case study to case study. It can be concluded that the proposed methodologies in Chapter 2 and Chapter 3 are hence representing promising approaches that tackle the discussed objective. Linking the mentioned findings back to the ongoing digitalization of the chemical and biological production industry, the presented methodologies could address the first objective of this thesis, and are valuable assets to support the understanding and development of process systems.

Chapter 4 introduced a method for globally optimizing surrogate process models using deterministic optimization frameworks. Symbolic regression allowed developing closed-form mathematical expressions, where the degrees of freedom represent the independent variables of the resulting algebraic equation. The obtained models were found to reveal a similar level of accuracy as those constructed by a purely data-driven approach with a Gaussian process. A major advantage of the closed-form surrogate models was shown to be the fact that they can easily be incorporated into optimization problems. The proposed workflow allowed the user to solve a surrogate model for global optimality using state-of-the-art solvers. Solving the resulting surrogate-based optimization problem was shown to be more straightforward and faster than globally optimizing a trained Gaussian process. One major drawback was found in the relation of the model accuracy and the accuracy of the optimal solution. The algebraic surrogate model identified by the symbolic regression was obtained to often be slightly less accurate compared to the trained Gaussian process, due to the usually better interpolation capabilities of the Gaussian process. Since the optimization is performed using the trained surrogate model, the global optimum of a symbolically regressed model might be less accurate and outperformed by multi-start local optimization strategies with Gaussian processes. Another drawback of the symbolic regression was shown in a very high computational time needed for training. However, having a closed-form expression at hand was shown to be useful, since the surrogate model could be easily incorporated into deterministic optimization frameworks, where also off-the-shelf solvers could be used.

The second objective of this thesis given in Section 1.4 aimed to develop frameworks for global optimization using surrogate models. Specifically, simple, accurate, and subsequently usable models should be developed purely from data. The concept developed in Chapter 4 tackled this objective by designing a decoupled modeling and optimization pipeline. Furthermore, since the methodology only

relies on input-output data, the model-based optimization procedure was proven to be highly flexible, where it was shown to be applicable to a variety of process systems that range from a meso scale (single process units) up to macro scale (full flowsheets). Hence, the objective could also be addressed regarding the transferability of the proposed framework. Since the optimization can be solved fast due to the available off-the-shelf deterministic solvers, such a framework could also be helpful in cases where the optimization problem has to be iteratively solved many times. This finding makes the methodology also applicable for conceptually different applications than investigated in Chapter 4, such as real-time optimization problems. In conclusion, the presented workflow could address the second objective, and might be valuable in supporting the optimization of process systems for which mechanistic models are hard or time-intensive to develop.

Chapter 5 introduced a new approach to compute the flexibility index in problems with complicating constraints. Due to the presence of such complicating constraints, it was not possible to directly include those constraints in the originally described deterministic flexibility index problem, since they would need to be available as closed-form expressions. The proposed methodology combined the original formulation of the flexibility index problem with a symbolically regressed surrogate model. The symbolic regression algorithm assumed no predefined model structure to approximate the process constraints. This enabled the accurate representation of constraints that are hard to model or handle numerically. The resulting hybrid flexibility approach was applied to industrially relevant processes, where the surrogate equations could accurately reproduce the complicating constraints. One drawback of the applied regression tool was found to be the high training time required for model building. Nevertheless – as it was described in the objectives above – having a closed-form expression at hand pays off in several aspects.

The last objective of this thesis given in Section 1.4 aimed to analyze process flexibility in case some process constraints are hard to model or even inaccessible. The framework developed in Chapter 5 supports decision-making in systems where process constraints are hard to describe or where they are only describable by measured input-output data. Chapter 5 contributed to this third objective by introducing a method that allows to include surrogate models into the flexibility index problem formulation. This way, a hybrid approach was generated that allows assessing process flexibility even in cases with complicating process constraints.

Furthermore, since the model training was decoupled from solving the flexibility index, the proposed framework allows studying a variety of process conditions in a short amount of time. Additionally, since the optimization problem formulation does not need to be significantly altered when studying a new process, the framework is transferrable to a wide range of different case studies. Hence, this objective was successfully addressed by the presented methodology, which could potentially be a valuable asset for practitioners in the chemical and biological industry that allows to bypass the challenges of modeling hard-to-describe process constraints in situations that require the assessment of a system's flexibility.

Overall, this thesis provides solutions to support the advancement of digitalization through modeling, optimization, and analysis of chemical and biological processes using surrogate and hybrid frameworks. Each chapter showcased the capabilities of the proposed methodologies and pipelines through comprehensive case studies. The effectiveness of the frameworks could be proven in diverse applications. This thesis bridges the gap between purely mechanistic and purely data-driven approaches by offering hybrid frameworks that integrate available knowledge with surrogate modeling and optimization techniques. These hybrid approaches not only allow to bypass existing challenges that were discussed, but they also offer alternative approaches in case existing methods are hard or not possible to be applied. While the presented approaches exhibit promising results, future research endeavors could build upon the present work, which will be discussed subsequently.

6.2 Limitations and future research directions

Data availability The presented methodologies were evaluated using synthetic data, as it is done very often in literature. The advantage is that a wide range of case studies can be produced and assessed in a short amount of time. However, the available data might not fully represent the complexity and variability of real-world processes. Future research should focus on assessing the performance of the proposed models and frameworks using industrial or real experimental data. This would provide further insights into the applicability and robustness of the developed methodologies. Additionally, the influence of data scarcity should be further explored. As mentioned in Chapter 2 and Chapter 3, there are many industrially relevant scenarios where data is very limited. Although considered and mentioned

above in this thesis, the impact of such a data-scarcity on the developed models and frameworks should be further assessed in future research.

Mechanistic backbone models In Chapter 2, the S-system was used as a mechanistic backbone for the model identification. While the S-system was found to be suitable for the investigated case studies, alternative mechanistic backbones for different systems were not investigated. Depending on the system under study, adapting this core formalism could increase the accuracy and applicability of the developed computational models. Future research should therefore focus on exploring alternative mechanistic backbones for different chemical and biological systems. Additionally, those models should be compared to other benchmarks that were mentioned throughout this thesis, such as ALAMO, BIDSAM, ALVEN, and others. Also, since the noise approximation method presented in Chapter 3 was not applied yet to the methodological framework given in Chapter 2 (MINLP approach), this should be explored in future research in order to improve the performance of the developed models in Chapter 2.

Hybrid framework comparison This thesis considered specific ways how hybrid models or frameworks might be developed. In Chapter 2, the expert knowledge was included via a known formalism, such as the S-system. In Chapter 3, prior mathematical/statistical information was included in the symbolic regression, and in Chapter 5, a hybrid framework was developed using surrogate models and a deterministic optimization scheme. However, there are many other possibilities to develop hybrid frameworks, where for example physics-informed neural networks, transformer models, and other regression or classification approaches could be used for predicting the time-dependent concentration profiles. Future research should therefore focus on exploring such alternative hybrid frameworks that can subsequently be compared to the methodologies presented in this thesis.

Surrogate-based optimization In Chapter 4, the algebraic surrogate models were globally optimized using deterministic solvers. Although compared to a proven methodology, the presented framework should be assessed in more detail, where for example the influence of higher dimensional inputs than presented in the last case study of Chapter 4 should be explored. Another aspect that was not investigated in this chapter was the use of multi-start local optimization in combination with the identified algebraic surrogate models. Future research should

therefore focus on the comparison of the identified global optimal solution with multi-start local optimization strategies, where a variety of solvers could be used. In addition to a more in-depth study of the proposed methodology with deterministic solvers, alternative optimization approaches – for example derivative-free optimization or meta heuristic methods such as simulated annealing, particle swarm optimization, or genetic algorithms – could be compared to the performance of the presented methodologies.

Integration into optimization and control frameworks The models identified in Chapters 2 and 3 were just used for prediction studies of time series concentration profiles. However, future research should focus on integrating the identified models into comprehensive optimization, model-based control, or similar frameworks. This would allow to further exploit the potential of the developed models and to support decision-making in steps that come after modeling the kinetic behavior. With this, it would be possible to further analyze the efficacy of the identified models. Furthermore, it would allow to assess the general applicability of such models for industrially relevant frameworks. Moreover, in Chapter 5, it was not investigated how the hybrid flexibility index calculation presented could be implemented in a larger optimization framework. Therefore, in future studies it would be interesting to include the presented flexibility assessment approach in an optimization problem, to, for example, calculate the best possible operating point or to find the optimum performance of a process, while guaranteeing the largest possible flexibility.

Symbolic regression algorithms In Chapters 3, 4, and 5, the Bayesian machine scientist was used as symbolic regression algorithm. This tool uses prior knowledge that was gained from a corpus of mathematical equations, and therefore represents statistical information about the mathematical structure of the obtained models. Future research should focus on comparing the used tool to other symbolic regression algorithms that might or might not consider different prior knowledge. Insights from such a study might be useful to decide in which situation which tool is more suitable. Furthermore, this thesis did not investigate possible algorithmic improvements of the Bayesian machine scientist (i.e., CPU times, search efficiency, memory efficiency, etc.). Future research should therefore also focus on further developing this tool, not only in terms of computational speed, but also considering specific chemical or biological information about the system under study. This could help to guide the algorithm more efficiently to-

wards equations that are more likely to explain the data precisely. One possibility to achieve this would be to, for example, use tailored kinetic equations during the training of the symbolic models, which would restrict the search space efficiently, allowing for faster model identification. Moreover, what was not investigated in depth in the assessments of Chapter 2 and Chapter 3, was the the concept of stiffness of the identified algebraic models. Since in those chapters the model building was performed in the derivative space, stiff ordinary differential equations might be encountered. Therefore, it would be necessary to include stiffness and stability analyses in such a procedure, which might be the focus of future work. In addition to these aspects, this thesis did not assess any improvements of the early stopping criteria of the Bayesian machine scientist. As shown for example in Chapter 3, the description length could be significantly reduced in the beginning of the training, where it was stagnating after that. Therefore, future research should focus on further comparing how the models identified along the training change in performance. This could support the development of early stopping criteria and hence improve the model identification speed. Furthermore, since the Bayesian machine scientist is stochastic in nature, the identified models might be different every time the training is performed. Running the training multiple times for the same training set would yield a distribution of model predictions, therefore generating some kind of ensemble model, allowing to compute a prediction interval. In this thesis, it was not investigated whether such a prediction interval might be useful or not, which might be interesting for future research. Last, it was not investigated whether the identified parameters of a symbolic regression tool could be re-estimated or improved by other methods (i.e., global optimization algorithms). Such a parameter improvement might also be interesting to investigate and should therefore be considered in future research.

By additionally addressing the above-mentioned considerations, future research endeavors have the potential to further enhance the efficacy and versatility of the methodologies proposed, ultimately contributing to the advancement of process modeling, optimization, and control in the chemical and biological industry.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., ... Zheng, X. (2015). Tensorflow: Large-scale machine learning on heterogeneous systems. *Google Research White Paper*.

ABB. (2018). *Digitalization and the chemical plant of the future* (white paper).

Accenture & WEF. (2017). *Digital transformation initiative - chemistry and advanced materials industry* (Rep.).

Adeyemo, S., & Bhattacharyya, D. (2024). Optimal nonlinear dynamic sparse model selection and bayesian parameter estimation for nonlinear systems. *Computers and Chemical Engineering*, *180*, 108502.

Agharafeie, R., Oliveira, R., Ramos, J. R. C., & Mendes, J. M. (2023). Application of hybrid neural models to bioprocesses: A systematic literature review.

Alcántara Avila, J. R., Kong, Z. Y., Lee, H.-Y., & Sunarso, J. (2021). Advancements in optimization and control techniques for intensifying processes. *Processes*, *9*(12), 2150.

Androulakis, I. P., Maranas, C. D., & Floudas, C. A. (1995). Alpha-bb: A global optimization method for general constrained nonconvex problems. *Journal of Global Optimization*, *7*(4), 337–363.

Applequist, G., Pekny, J., & Reklaitis, G. (2000). Risk and uncertainty in managing chemical manufacturing supply chains. *Computers and Chemical Engineering*, *24*(9-10), 2211–2222.

Arora, J. S. (2012). *Introduction to optimum design* (3rd ed.). Elsevier.

Azarpour, A., N.G. Borhani, T., R. Wan Alwi, S., A. Manan, Z., & I. Abdul Mutalib, M. (2017). A generic hybrid model development for process analysis of industrial fixed-bed catalytic reactors. *Chemical Engineering Research and Design*, *117*, 149–167.

Azevedo, C. R., Díaz, V. G., Prado-Rubio, O. A., Willis, M. J., Préat, V., Oliveira, R., & Stosch, M. (2019). Hybrid semiparametric modeling: A modular process systems engineering approach for the integration of available knowledge sources. In *Systems engineering in the fourth industrial revolution* (pp. 345–373). Wiley.

Badejo, O., & Ierapetritou, M. (2022). Integrating tactical planning, operational planning and scheduling using data-driven feasibility analysis. *Computers and Chemical Engineering*, *161*, 107759.

Banerjee, I., & Ierapetritou, M. G. (2005). Feasibility evaluation of nonconvex systems using shape reconstruction techniques. *Industrial and Engineering Chemistry Research*, *44*(10), 3638–3647.

Bansal, V., Sakizlis, V., Ross, R., Perkins, J. D., & Pistikopoulos, E. N. (2003). New algorithms for mixed-integer dynamic optimization. *Computers and Chemical Engineering*, *27*(5), 647–668.

Barbosa-Póvoa, A. P. (2012). Progresses and challenges in process industry supply chains optimization. *Current Opinion in Chemical Engineering*, *1*(4), 446–452.

Baur, D., Angarita, M., Müller-Späth, T., & Morbidelli, M. (2016). Optimal model-based design of the twin-column capturesmb process improves capacity utilization and productivity in protein a affinity capture. *Biotechnology Journal*, *11*(1), 135–145.

Beier, G., Niehoff, S., & Xue, B. (2018). More sustainability in industry through industrial internet of things? *Applied Sciences*, *8*(2), 219.

Bellman, R. (2010). *Dynamic programming* (First Princeton landmarks in mathematics edition). Princeton University Press.
OCLC: 1264090370.

Bellman, R., Jacquez, J., Kalaba, R., & Schwimmer, S. (1967). Quasilinearization and the estimation of chemical rate constants from raw kinetic data. *Mathematical Biosciences*, *1*(1), 71–76.

Belotti, P., Lee, J., Liberti, L., Margot, F., & Wächter, A. (2009). Branching and bounds tighteningtechniques for non-convex minlp. *Optimization Methods and Software*, *24*(4-5), 597–634.

Ben-Tal, A., El Ghaoui, L., & Nemirovski, A. S. (2009). *Robust optimization*. Princeton University Press.
OCLC: ocn318672208.

Ben-Tal, A., & Nemirovski, A. (2002). Robust optimization – methodology and applications. *Mathematical Programming*, *92*(3), 453–480.

Bergamini, M. L., Aguirre, P., & Grossmann, I. (2005). Logic-based outer approximation for globally optimal synthesis of process networks. *Computers and Chemical Engineering*, *29*(9), 1914–1933.

Berning, G., Brandenburg, M., Gürsoy, K., Kussi, J. S., Mehta, V., & Tölle, F.-J. (2004). Integrating collaborative planning and supply chain optimization for the chemical process industry (i)—methodology. *Computers and Chemical Engineering*, *28*(6-7), 913–927.

Berthold, M. R., Cebron, N., Dill, F., Gabriel, T. R., Kötter, T., Meinl, T., Ohl, P., Sieb, C., Thiel, K., & Wiswedel, B. (2007). Knime: The konstanz information miner. *Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007)*.

Beykal, B., Boukouvala, F., Floudas, C. A., Sorek, N., Zalavadia, H., & Gildin, E. (2018). Global optimization of grey-box computational systems using surrogate functions and application to highly constrained oil-field operations. *Computers and Chemical Engineering*, *114*, 99–110.

Bhosekar, A., & Ierapetritou, M. G. (2018). Advances in surrogate based modeling, feasibility analysis, and optimization: A review. *Computers and Chemical Engineering*, *108*, 250–267.

Biegler, L. T., & Grossmann, I. E. (2004). Retrospective on optimization. *Computers and Chemical Engineering*, *28*(8), 1169–1192.

Biegler, L. T., Grossmann, I. E., & Westerberg, A. W. (1997). *Systematic methods of chemical process design*. Prentice Hall PTR.

Birge, J. R., & Louveaux, F. (2011). *Introduction to stochastic programming* (2nd ed). Springer.

Bishnu, S. K., Alnouri, S. Y., & Al-Mohannadi, D. M. (2023). Computational applications using data driven modeling in process systems: A review. *Digital Chemical Engineering*, *8*, 100111.

Bishop, C. M. (2006). *Pattern recognition and machine learning*.

Bongartz, D., & Mitsos, A. (2019). Deterministic global flowsheet optimization: Between equation-oriented and sequential-modular methods. *AIChE Journal*, *65*(3), 1022–1034.

Bongartz, D., Najman, J., Sass, S., & Mitsos, A. (2020). Mccormick-based algorithm for mixed-integer nonlinear global optimization, technical report.

Bonvin, D., Georgakis, C., Pantelides, C. C., Barolo, M., Grover, M. A., Rodrigues, D., Schneider, R., & Dochain, D. (2016). Linking models and experiments. *Industrial and Engineering Chemistry Research*, *55*(25), 6891–6903.

Bonyadi, M. R., & Michalewicz, Z. (2017). Particle swarm optimization for single objective continuous space problems: A review. *Evolutionary Computation*, *25*(1), 1–54.

Boukouvala, F., & Floudas, C. A. (2017). Argonaut: Algorithms for global optimization of constrained grey-box computational problems. *Optimization Letters*, *11*(5), 895–913.

Boukouvala, F., Hasan, M. M. F., & Floudas, C. A. (2017). Global optimization of general constrained grey-box models: New method and its application to constrained pdes for pressure swing adsorption. *Journal of Global Optimization*, *67*(1-2), 3–42.

Boukouvala, F., & Ierapetritou, M. G. (2012). Feasibility analysis of black-box processes using an adaptive sampling kriging-based method. *Computers and Chemical Engineering*, *36*(1), 358–368.

Boukouvala, F., Muzzio, F. J., & Ierapetritou, M. G. (2010). Design space of pharmaceutical processes using data-driven-based methods. *Journal of Pharmaceutical Innovation*, *5*(3), 119–137.

Boukouvala, F., Muzzio, F. J., & Ierapetritou, M. G. (2011). Feasibility analysis of black-box processes using an adaptive sampling kriging based method. In *Computer aided chemical engineering* (pp. 432–436, Vol. 29). Elsevier B.V.

Bradford, E., Schweidtmann, A. M., & Lapkin, A. (2018). Efficient multiobjective optimization employing gaussian processes, spectral sampling and a genetic algorithm. *Journal of Global Optimization*, *71*(2), 407–438.

Brendel, M., Bonvin, D., & Marquardt, W. (2006). Incremental identification of kinetic models for homogeneous reaction systems. *Chemical Engineering Science*, *61*(16), 5404–5420.

Brunton, S. L., Proctor, J. L., Kutz, J. N., & Bialek, W. (2016). Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences of the United States of America*, *113*(15), 3932–3937.

Burnak, B., Diangelakis, N. A., & Pistikopoulos, E. N. (2020). Integrated process design and operational optimization via multiparametric programming. *Synthesis Lectures on Engineering, Science, and Technology*, *2*(7), 1–258.

Bussieck, M. R., & Drud, A. S. (2001). Sbb: A new solver for mixed integer nonlinear programming.

Bynum, M. L., Hackebeil, G. A., Hart, W. E., Laird, C. D., Nicholson, B. L., Siirola, J. D., & Watson, J.-P. (2021). *Pyomo - optimization modeling in python* (Third edition). Springer.

Canova, C. T., Inguva, P. K., & Braatz, R. D. (2023). Mechanistic modeling of viral particle production. *Biotechnology and Bioengineering*, *120*(3), 629–641.

Carey, G., & Finlayson, B. A. (1975). Orthogonal collocation on finite elements. *Chemical Engineering Science*, *30*(5-6), 587–596.

Cavazzuti, M. (2013). *Optimization methods: From theory to design scientific and technological aspects in mechanics*. Springer Berlin Heidelberg.

Ceccon, F., Jalving, J., Haddad, J., Thebelt, A., Tsay, C., Laird, C. D., & Misener, R. (2022). Omlt: Optimization and machine learning toolkit. *Journal of Machine Learning Research*, *23*(349), 1–8.

Chehade, G., & Dincer, I. (2021). Advanced kinetic modelling and simulation of a new small modular ammonia production unit. *Chemical Engineering Science*, *236*, 116512.

Chen, X., Eder, M. A., Shihavuddin, A., & Zheng, D. (2021). A human-cyber-physical system toward intelligent wind turbine operation and maintenance. *Sustainability*, *13*(2), 561.

Chen, Y., & Ierapetritou, M. (2020). A framework of hybrid model development with identification of plant-model mismatch. *AIChE Journal*, *66*(10), e16996.

Cococcioni, M., & Fiaschi, L. (2021). The big-m method with the numerical infinite m. *Optimization Letters*, *15*(7), 2455–2468.

Collodi, G., Azzaro, G., Ferrari, N., & Santos, S. (2017). Techno-economic evaluation of deploying ccs in smr based merchant h2 production with ng as feedstock and fuel. *Energy Procedia*, *114*, 2690–2712.

Communication-Promoters-Group-of-the-Industry-Science (Ed.). (2013). *Recommendations for implementing the strategic initiative industrie 4.0* (tech. rep.).

Cornish-Bowden, A. (2015). One hundred years of michaelis–menten kinetics. *Perspectives in Science*, *4*, 3–9.

Costa, L., & Oliveira, P. (2001). Evolutionary algorithms approach to the solution of mixed integer non-linear programming problems. *Computers and Chemical Engineering*, *25*(2), 257–266.

Cozad, A., & Sahinidis, N. V. (2018). A global minlp approach to symbolic regression. *Mathematical Programming*, *170*(1), 97–119.

Cozad, A., Sahinidis, N. V., & Miller, D. C. (2014). Learning surrogate models for simulation-based optimization. *AIChE Journal*, *60*(6), 2211–2227.

Cranmer, M. (2020). Pysr: Fast and parallelized symbolic regression in python/julia.

Cranmer, M. (2023). Interpretable machine learning for science with pysr and symbolicregression.jl.

Cranmer, M., Sanchez-Gonzalez, A., Battaglia, P., Xu, R., Cranmer, K., Spergel, D., & Ho, S. (2020). Discovering symbolic models from deep learning with inductive biases. *Advances in Neural Information Processing Systems*, *2020-Decem*(NeurIPS), 1–14.

Dantzig, G. B., & Wolfe, P. (1960). Decomposition principle for linear programs. *Operations Research*, *8*(1), 101–111.

Datahow AG. (2024).

DataModeler. (2023). Datamodeler.

De Carvalho Servia, M. Á., & Del Rio Chanona, E. A. (2023a). Interpretable machine learning for kinetic rate model discovery. In *Machine learning and hybrid modelling for reaction engineering: Theory and applications*.

De Carvalho Servia, M. Á., & Del Rio Chanona, E. A. (2023b). Model structure identification. In *Machine learning and hybrid modelling for reaction engineering: Theory and applications*.

De Carvalho Servia, M. Á., Sandoval, I. O., Hii, K. K., Hellgardt, K., Zhang, D., & Del Rio Chanona, E. A. (2024). The automated discovery of kinetic rate models – methodological frameworks. *Digital Discovery*.

Del Rio Chanona, E. A., Cong, X., Bradford, E., Zhang, D., & Jing, K. (2019). Review of advanced physical and data-driven models for dynamic bioprocess simulation: Case study of algae–bacteria consortium wastewater treatment. *Biotechnology and Bioengineering*, *116*(2), 342–353.

Del Rio Chanona, E. A., rashid Ahmed, N., Zhang, D., Lu, Y., & Jing, K. (2017). Kinetic modeling and process analysis for desmodesmus sp. lutein photo-production. *AIChE Journal*, *63*(7), 2546–2554.

Del Rio Chanona, E. A., Wagner, J. L., Ali, H., Fiorelli, F., Zhang, D., & Hellgardt, K. (2019). Deep learning-based surrogate modeling and optimization for microalgal biofuel production and photobioreactor design. *AIChE Journal*, *65*(3), 915–923.

De-Luca, R., Pupo-Correia, M., Feldhofer, M., Martins, D. L., Umprecht, A., Shahmohammadi, A., Corona, D., & Von Stosch, M. (2024). Hybrid modeling of an ultracentrifugation process for separation of full and empty adeno-associated virus particles. *Bioprocess and Biosystems Engineering*, *47*(6), 877–890.

Díaz-Madroñero, M., Mula, J., & Peidro, D. (2014). A review of discrete-time optimization models for tactical production planning. *International Journal of Production Research*, *52*(17), 5171–5205.

Ding, C., & Ierapetritou, M. (2021). A novel framework of surrogate-based feasibility analysis for establishing design space of twin-column continuous chromatography. *International Journal of Pharmaceutics*, *609*, 121161.

Diveev, A., & Shmalko, E. (2021). *Machine learning control by symbolic regression*. Springer International Publishing.

Diwekar, U. M. (2020). *Introduction to applied optimization* (Vol. 22). Springer International Publishing.

Dormand, J. R., & Prince, P. J. (1980). A family of embedded runge-kutta formulae. *Journal of Computational and Applied Mathematics*, *6*(1), 19–26.

Duran, M. A., & Grossmann, I. E. (1986a). Simultaneous optimization and heat integration of chemical processes. *AIChE Journal*, *32*(1), 123–138.

Duran, M. A., & Grossmann, I. E. (1986b). An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, *36*(3), 307–339.

Dutta, S. (2016). *Optimization in chemical engineering*. Cambridge University Press, is part of the University of Cambridge.

Edgar, T. F., Himmelblau, D. M., & Himmelblau, D. M. (1988). *Optimization of chemical processes*. McGraw-Hill.

Ehrenstein, M., Wang, C.-H., & Guillén-Gosálbez, G. (2019). Strategic planning of supply chains considering extreme events: Novel heuristic and application to the petrochemical industry. *Computers and Chemical Engineering*, *125*, 306–323.

Elble, J. M., & Sahinidis, N. V. (2012). Scaling linear optimization problems prior to application of the simplex method. *Computational Optimization and Applications*, *52*(2), 345–371.

Elkamel, A., Al-Ajmi, A., & Fahim, M. (1999). Modeling the hydrocracking process using artificial neural networks. *Petroleum Science and Technology*, *17*(9-10), 931–954.

Elnashaie, S. S. E. H., & Elshishini, S. S. (2022). *Modelling, simulation and optimization of industrial fixed bed catalytic reactors* (1st ed.). Routledge.

Esposito, W. R., & Floudas, C. A. (2002). Deterministic global optimization in isothermal reactor network synthesis. *Journal of Global Optimization*, *22*(1/4), 59–95.

Esposito, W. R., & Floudas, C. A. (2000). Global optimization for the parameter estimation of differential-algebraic systems. *Industrial and Engineering Chemistry Research*, *39*(5), 1291–1310.

Fantke, P., Cinquemani, C., Yaseneva, P., De Mello, J., Schwabe, H., Ebeling, B., & Lapkin, A. A. (2021). Transition to sustainable chemistry through digitalization. *Chem*, *7*(11), 2866–2882.

FDA. (2010). *International conference on harmonisation (ich) q8 guidance for industry on pharmaceutical development* (tech. rep.).

Ferreira, J., Pedemonte, M., & Torres, A. I. (2019). A genetic programming approach for construction of surrogate models. In *Computer aided chemical engineering* (pp. 451–456, Vol. 47). Elsevier.

Ferreira, J., Torres, A. I., & Pedemonte, M. (2019). A comparative study on the numerical performance of kaizen programming and genetic programming for symbolic regression problems. *2019 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, 1–6.

Filho, P. I. O., Carmalt, C. J., Angeli, P., & Fraga, E. S. (2020). Mathematical modeling for the design and scale-up of a large industrial aerosol-assisted chemical vapor deposition process under uncertainty. *Industrial and Engineering Chemistry Research*, *59*(3), 1249–1260.

Fletcher, R. (2000). *Practical methods of optimization*. John Wiley; Sons, Ltd.

Floudas, C. A., Gümüş, Z. H., & Ierapetritou, M. G. (2001). Global optimization in design under uncertainty: Feasibility test and flexibility index problems. *Industrial and Engineering Chemistry Research*, *40*(20), 4267–4282.

Floudas, C. A., Pardalos, P. M., Adjiman, C. S., Esposito, W. R., Gümüş, Z. H., Harding, S. T., Klepeis, J. L., Meyer, C. A., & Schweiger, C. A. (1999). *Handbook of test problems in local and global optimization* (Vol. 33). Springer US.

Foresee, F. D., & Hagan, M. (1997). Gauss-newton approximation to bayesian learning. *Proceedings of International Conference on Neural Networks (ICNN'97)*, *3*, 1930–1935.

Forster, T., Vázquez, D., Cruz-Bournazou, M. N., Butté, A., & Guillén-Gosálbez, G. (2023). Modeling of bioprocesses via minlp-based symbolic regression of s-system formalisms. *Computers and Chemical Engineering*, *170*, 108108.

Forster, T., Vázquez, D., & Guillén-Gosálbez, G. (2023a). Algebraic surrogate-based process optimization using bayesian symbolic learning. *AIChE Journal*, e18110.

Forster, T., Vázquez, D., & Guillén-Gosálbez, G. (2023b). Global optimization of symbolic surrogate process models based on bayesian learning. In A. C. Kokossis, M. C. Georgiadis, & E. Pistikopoulos (Eds.), *Computer aided chemical engineering* (pp. 1241–1246, Vol. 52). Elsevier.

Forster, T., Vázquez, D., Moreno-Palancas, I. F., & Guillén-Gosálbez, G. (2024a). Algebraic surrogate-based flexibility analysis of process units with complicating process constraints. *Computers and Chemical Engineering*, *184*, 108630.

Forster, T., Vázquez, D., Moreno-Palancas, I. F., & Guillén-Gosálbez, G. (2024b). Flexibility analysis using surrogate models generated via symbolic regression. In *Computer aided chemical engineering* (pp. 2791–2796, Vol. 53). Elsevier.

Forster, T., Vázquez, D., Müller, C., & Guillén-Gosálbez, G. (2024). Machine learning uncovers analytical kinetic models of bioprocesses. *Chemical Engineering Science*, *300*, 120606.

Gábor, A., & Banga, J. R. (2015). Robust and efficient parameter estimation in dynamic models of biological systems. *BMC Systems Biology*, *9*(1), 74.

Gabrielli, P., Fürer, F., Mavromatidis, G., & Mazzotti, M. (2019). Robust and optimal design of multi-energy systems with seasonal storage through uncertainty analysis. *Applied Energy*, *238*, 1192–1210.

GAMS Development Corporation. (2020). Gams development corporation. general algebraic modeling system (gams) release 35.1.0.

GAMS Development Corporation. (2022). Gams development corporation. general algebraic modeling system (gams) release 40.2.0.

Garcia, D. J., & You, F. (2015). Supply chain design and optimization: Challenges and opportunities. *Computers and Chemical Engineering*, *81*, 153–170.

Gherman, I. M., Abdallah, Z. S., Pang, W., Gorochowski, T. E., Grierson, C. S., & Marucci, L. (2023). Bridging the gap between mechanistic biological models and machine learning surrogates (M. H. Schulz, Ed.). *PLOS Computational Biology*, *19*(4), e1010988.

Glassey, J., & Von Stosch, M. (2018). *Hybrid modeling in process industries*. Taylor; Francis Group, LLC.

Gnoth, S., Jenzsch, M., Simutis, R., & Lübbert, A. (2007). Product formation kinetics in a recombinant protein production process. *IFAC Proceedings Volumes*, *40*(4), 201–206.

Gnoth, S., Jenzsch, M., Simutis, R., & Lübbert, A. (2008a). Control of cultivation processes for recombinant protein production: A review. *Bioprocess and Biosystems Engineering*, *31*(1), 21–39.

Gnoth, S., Jenzsch, M., Simutis, R., & Lübbert, A. (2008b). Product formation kinetics in genetically modified e. coli bacteria: Inclusion body formation. *Bioprocess and Biosystems Engineering*, *31*(1), 41–46.

Gnoth, S., Simutis, R., & Lübbert, A. (2010). Selective expression of the soluble product fraction in escherichia coli cultures employed in recombinant protein production processes. *Applied Microbiology and Biotechnology*, *87*(6), 2047–2058.

Gonzalez-Garay, A., & Guillen-Gosalbez, G. (2018). Suscape: A framework for the optimal design of sustainable chemical processes incorporating data envelopment analysis. *Chemical Engineering Research and Design*, *137*, 246–264.

González-Garay, A., Frei, M. S., Al-Qahtani, A., Mondelli, C., Guillén-Gosálbez, G., & Pérez-Ramírez, J. (2019). Plant-to-planet analysis of co2-based methanol processes. *Energy and Environmental Science*, *12*(12), 3425–3436.

Götz, J., Bodea, L.-G., & Goedert, M. (2018). Rodent models for alzheimer disease. *Nature Reviews Neuroscience*, *19*(10), 583–598.

Goyal, V., & Ierapetritou, M. G. (2002). Determination of operability limits using simplicial approximation. *AIChE Journal*, *48*(12), 2902–2909.

Goyal, V., & Ierapetritou, M. G. (2003). Framework for evaluating the feasibility/operability of nonconvex processes. *AIChE Journal*, *49*(5), 1233–1240.

Grossmann, I. E., Apap, R. M., Calfa, B. A., Garcia-Herreros, P., & Zhang, Q. (2017). Mathematical programming techniques for optimization under uncertainty and their application in process systems engineering. *Theoretical Foundations of Chemical Engineering*, *51*(6), 893–909.

Grossmann, I. E., & Floudas, C. A. (1987). Active constraint strategy for flexibility analysis in chemical processes. *Computers and Chemical Engineering*, *11*(6), 675–693.

Grossmann, I. E., & Halemane, K. P. (1982). Decomposition strategy for designing flexible chemical plants. *AIChE Journal*, *28*(4), 686–694.

Grossmann, I. E., Halemane, K. P., & Swaney, R. E. (1983). Optimization strategies for flexible chemical processes. *Computers and Chemical Engineering*, *7*(4), 439–462.

Grossmann, I. (2005). Enterprise-wide optimization: A new frontier in process systems engineering. *AIChE Journal*, *51*(7), 1846–1857.

Grossmann, I. E. (1996). *Global optimization in engineering design (nonconvex optimization and its applications)* (Vol. 9).

Grossmann, I. E. (2012). Advances in mathematical programming models for enterprise-wide optimization. *Computers and Chemical Engineering*, *47*, 2–18.

Grossmann, I. E. (2014). Challenges in the application of mathematical programming in the enterprise-wide optimization of process industries. *Theoretical Foundations of Chemical Engineering*, *48*(5), 555–573.

Grossmann, I. E. (2021). *Advanced optimization for process systems engineering*. Cambridge University Press.

Grossmann, I. E., Apap, R. M., Calfa, B. A., García-Herreros, P., & Zhang, Q. (2016). Recent advances in mathematical programming techniques for the optimization of process systems under uncertainty. *Computers and Chemical Engineering*, *91*, 3–14.

Grossmann, I. E., Calfa, B. A., & Garcia-Herreros, P. (2014). Evolution of concepts and models for quantifying resiliency and flexibility of chemical processes. *Computers and Chemical Engineering*, *70*, 22–34.

Grossmann, I. E., & Harjunkoski, I. (2019). Process systems engineering: Academic and industrial perspectives. *Computers and Chemical Engineering*, *126*, 474–484.

Grossmann, I. E., & Westerberg, A. W. (2000). Research challenges in process systems engineering. *AIChE Journal*, *46*(9), 1700–1703.

Grünwald, P. D. (2007). *The minimum description length principle*. The MIT Press.

Guillén-Gosálbez, G., Badell, M., Espuña, A., & Puigjaner, L. (2006). Simultaneous optimization of process operations and financial decisions to enhance the integrated planning/scheduling of chemical supply chains. *Computers and Chemical Engineering*, *30*(3), 421–436.

Guillén-Gosálbez, G., Mele, F. D., Espuña, A., & Puigjaner, L. (2006). Addressing the design of chemical supply chains under demand uncertainty. *Industrial and Engineering Chemistry Research*, *45*(22), 7566–7581.

Guillén-Gosálbez, G., Miró, A., Alves, R., Sorribas, A., & Jiménez, L. (2013). Identification of regulatory structure and kinetic parameters of biochemical networks via mixed-integer dynamic optimization. *BMC Systems Biology*, *7*(1), 113.

Guimerà, R., Reichardt, I., Aguilar-Mogas, A., Massucci, F. A., Miranda, M., Pallarès, J., & Sales-Pardo, M. (2020). A bayesian machine scientist to aid in the solution of challenging scientific problems. *Science Advances*, *6*(5).

Guyon, I., Sun-Hosoya, L., Boulle, M., Escalante, H., Escalera, S., Liu, Z., Jajetic, D., Ray, B., Saeed, M., & Sebag, M. (2017). Analysis of the automl challenge series 2015-2018. In *Automated machine learning*. Springer, Cham.

Hahn, T., Trunzer, T., Rusly, F., Zolyomi, R., Shekhawat, L. K., Malmquist, G., Hesslein, A., & Tjandra, H. (2023). Predictive scaling of fiber-based protein a capture chromatography using mechanistic modeling. *Biotechnology and Bioengineering*, bit.28434.

Halemane, K. P., & Grossmann, I. E. (1983). Optimal process design under uncertainty. *AIChE Journal*, *29*(3), 425–433.

Hansen, M. H., & Yu, B. (2001). Model selection and the principle of minimum description length. *Journal of the American Statistical Association*, *96*(454), 746–774.

Hart, W. E., Watson, J.-P., & Woodruff, D. L. (2011). Pyomo: Modeling and solving mathematical programs in python. *Mathematical Programming Computation*, *3*(3), 219–260.

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, *57*(1), 97.

Haydary, J. (2019). *Chemical process design and simulation: Aspen plus and aspen hysys applications*. Wiley.

Hedengren, J. D., Shishavan, R. A., Powell, K. M., & Edgar, T. F. (2014). Nonlinear modeling, estimation and predictive control in apmonitor. *Computers and Chemical Engineering*, *70*, 133–148.

Helleckes, L. M., Hemmerich, J., Wiechert, W., Von Lieres, E., & Grünberger, A. (2023). Machine learning in bioprocess development: From promise to practice. *Trends in Biotechnology*, *41*(6), 817–835.

Henriques, D., Rocha, M., Saez-Rodriguez, J., & Banga, J. R. (2015). Reverse engineering of logic-based differential equation models using a mixed-integer dynamic optimization approach. *Bioinformatics*, *31*(18), 2999–3007.

Hewson, J. T., Strittmatter, Y., Marinescu, I., Williams, C. C., & Musslick, S. (2023). Bayesian machine scientist for model discovery in psychology. *NeurIPS AI for Science Workshop*.

Hillier, F. S., & Lieberman, G. J. (2010). *Introduction to operations research* (9. ed., internat. ed). McGraw-Hill.

Holland, J. H. (1992). Genetic algorithms. *Scientific American*, *267*(1), 66–73.

Horst, R., & Tuy, H. (1996). *Global optimization*. Springer Berlin Heidelberg.

Hüllen, G., Zhai, J., Kim, S. H., Sinha, A., Realff, M. J., & Boukouvala, F. (2020). Managing uncertainty in data-driven simulation-based optimization. *Computers and Chemical Engineering*, *136*, 106519.

Huš, M., Kopač, D., Štefančič, N. S., Jurković, D. L., Dasireddy, V. D. B. C., & Likozar, B. (2017). Unravelling the mechanisms of co2 hydrogenation to methanol on cu-based catalysts using first-principles multiscale modelling and experiments. *Catalysis Science and Technology*, *7*(24), 5900–5913.

Hwang, C.-R. (1988). Simulated annealing: Theory and applications. *Acta Applicandae Mathematica*, *12*, 108–111.

Ierapetritou, M. G. (2001). New approach for quantifying process feasibility: Convex and 1-d quasi-convex regions. *AIChE Journal*, *47*(6), 1407–1417.

Ierapetritou, M. G., & Pistikopoulos, E. N. (1994). Novel optimization approach of stochastic planning models. *Industrial and Engineering Chemistry Research*, *33*(8), 1930–1942.

Ioannou, I., Carlo D'Angelo, S., Galán-Martin, A., Pozo, C., Pérez-Ramirez, J., & Guillén-Gosálbez, G. (2021). Process modelling and life cycle assessment coupled with experimental work to shape the future sustainable production of chemicals and fuels. *Reaction Chemistry and Engineering*, *6*(7), 1179–1194.

Jog, S., Vázquez, D., Santos, L. F., Caballero, J. A., & Guillén-Gosálbez, G. (2023). Hybrid analytical surrogate-based process optimization via bayesian symbolic regression. *Computers and Chemical Engineering*, 108563.

Johnson, K. A., & Goody, R. S. (2011). The original michaelis constant: Translation of the 1913 michaelis–menten paper. *Biochemistry*, *50*(39), 8264–8269.

Jones, D. R., Schonlau, M., & Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, *13*, 455–492.

Kahrs, O., & Marquardt, W. (2007). The validity domain of hybrid models and its application in process optimization. *Chemical Engineering and Processing: Process Intensification*, *46*(11), 1054–1066.

Kay, S., Sanchez Medina, E. I., Sundmacher, K., & Zhang, D. (2024). A symbolic regression based methodology for the construction of interpretable and predictive thermodynamic models. In *Computer aided chemical engineering* (pp. 2701–2706, Vol. 53). Elsevier.

Keane, M. A., Koza, J. R., & Rice, J. P. (1993). Finding an impulse response function using genetic programming. *1993 American Control Conference*, (1), 2345–2350.

Keith, D. W., Holmes, G., St. Angelo, D., & Heidel, K. (2018). A process for capturing co2 from the atmosphere. *Joule*, *2*(8), 1573–1594.

Kennedy, J., & Eberhart, R. (2006). Particle swarm optimization. *Proceedings of ICNN'95 - International Conference on Neural Networks*, *4*, 1942–1948.

Kesavan, P., Allgor, R. J., Gatzke, E. P., & Barton, P. I. (2004). Outer approximation algorithms for separable nonconvex mixed-integer nonlinear programs. *Mathematical Programming*, *100*(3), 517–535.

Kim, I.-W., Liebman, M., & Edgar, T. (1991). A sequential error-in-variables method for nonlinear dynamic systems. *Computers and Chemical Engineering*, *15*(9), 663–670.

Klatt, K.-U., & Marquardt, W. (2009). Perspectives for process systems engineering—personal views from academia and industry. *Computers and Chemical Engineering*, *33*(3), 536–550.

Klipp, E., & Liebermeister, W. (2006). Mathematical modeling of intracellular signaling pathways. *BMC Neuroscience*, *7*(1), S10.

Koh, L., Orzes, G., & Jia, F. ((2019). The fourth industrial revolution (industry 4.0): Technologies disruption on operations and supply chain management. *International Journal of Operations and Production Management*, *39*(6/7/8), 817–828.

Koza, J. R. (1994). Genetic programming as a means for programming computers by natural selection. *Statistics and Computing*, *4*(2), 87–112.

Krige, D. G. (1951). A statistical approach to some basic mine valuation problems on the witwatersrand. *J. of the Chem., Metal. and Mining Soc. of South Africa*, 119–139.

Kroll, P., Hofer, A., Stelzer, I. V., & Herwig, C. (2017). Workflow to set up substantial target-oriented mechanistic process models in bioprocess engineering. *Process Biochemistry*, *62*, 24–36.

Kroll, P., Hofer, A., Ulonska, S., Kager, J., & Herwig, C. (2017). Model-based methods in the biopharmaceutical process lifecycle. *Pharmaceutical Research*, *34*(12), 2596–2613.

Kubic, W. L., & Stein, F. P. (1988). A theory of design reliability using probability and fuzzy sets. *AIChE Journal*, *34*(4), 583–601.

Kyriakopoulos, S., Ang, K. S., Lakshmanan, M., Huang, Z., Yoon, S., Gunawan, R., & Lee, D.-Y. (2018). Kinetic modeling of mammalian cell culture bioprocessing: The quest to advance biomanufacturing. *Biotechnology Journal*, *13*(3), 1700229.

Lasi, H., Fettke, P., Kemper, H.-G., Feld, T., & Hoffmann, M. (2014). Industry 4.0. *Business and Information Systems Engineering*, *6*(4), 239–242.

Lasschuit, W., & Thijssen, N. (2004). Supporting supply chain planning and scheduling decisions in the oil and chemical industry. *Computers and Chemical Engineering*, *28*(6-7), 863–870.

Lastrucci, G., Theisen, M. F., & Schweidtmann, A. M. (2024). Physics-informed neural networks and time-series transformer for modeling of chemical reactors. In *Computer aided chemical engineering* (pp. 571–576, Vol. 53). Elsevier.

Lee, J., Sun, W., Lee, J. H., & Braatz, R. D. (2024). Learning first-principles knowledge from data. In *Artificial intelligence in manufacturing* (pp. 39–62). Elsevier.

Lee, J. H., Shin, J., & Realff, M. J. (2018). Machine learning: Overview of the recent progresses and implications for the process systems engineering field. *Computers and Chemical Engineering*, *114*, 111–121.

Li, C., & Grossmann, I. E. (2021). A review of stochastic programming methods for optimization of process systems under uncertainty. *Frontiers in Chemical Engineering*, *2*, 1–20.

Li, C., & Samulski, R. J. (2020). Engineering adeno-associated virus vectors for gene therapy. *Nature Reviews Genetics*, *21*(4), 255–272.

Li, Z., Ding, R., & Floudas, C. A. (2011). A comparative theoretical and computational study on robust counterpart optimization: I. robust linear optimization and robust mixed integer linear optimization. *Industrial and Engineering Chemistry Research*, *50*(18), 10567–10603.

Li, Z., & Ierapetritou, M. (2008). Process scheduling under uncertainty: Review and challenges. *Computers and Chemical Engineering*, *32*(4), 715–727.

Li, Z., & Ierapetritou, M. G. (2008). Robust optimization for process scheduling under uncertainty. *Industrial and Engineering Chemistry Research*, *47*(12), 4148–4157.

Li, Z., & Ierapetritou, M. G. (2012). Capacity expansion planning through augmented lagrangian optimization and scenario decomposition. *AIChE Journal*, *58*(3), 871–883.

Lillacci, G., & Khammash, M. (2010). Parameter estimation and model selection in computational biology (A. R. Asthagiri, Ed.). *PLoS Computational Biology*, *6*(3), e1000696.

Lin, X., Janak, S. L., & Floudas, C. A. (2004). A new robust optimization approach for scheduling under uncertainty:: I. bounded uncertainty. *Computers and Chemical Engineering*, *28*(6), 1069–1085.

Liu, M.-L., Sahinidis, N. V., & Shectman, J. P. (1996). Planning of chemical process networks via global concave minimization. In I. E. Grossmann (Ed.), *Global optimization in engineering design* (pp. 195–230). Springer US.

Locatelli, M., & Schoen, F. (2013). *Global optimization - theory, algorithms and applications*. SIAM.

MacKay, D. J. C. (1992). Bayesian interpolation. *Neural Computation*, *4*(3), 415–447.

Mahanty, B. (2023). Hybrid modeling in bioprocess dynamics: Structural variabilities, implementation strategies, and practical challenges. *Biotechnology and Bioengineering*, *120*(8), 2072–2091.

Marti, K., & Kall, P. (Eds.). (1995). *Stochastic programming: Numerical techniques and engineering applications: Proceedings of the 2nd gamm/ifip-workshop on "stochastic optimization: Numerical methods and technical applications", held at the federal armed forces university munich, neubiberg/münchen, germany, june 15-17, 1993*. Springer.

McKay, B., Willis, M., & Barton, G. (1997). Steady-state modelling of chemical process systems using genetic programming. *Computers and Chemical Engineering*, *21*(9), 981–996.

McKay, B., Willis, M., Searson, D., & Montague, G. (1999). Non-linear continuum regression using genetic programming. *Proc. of the Genetic and Evolutionary Computation Conf. (GECCO)-99*, *2*, 1106–1111.

Mercier, S. M., Diepenbroek, B., Wijffels, R. H., & Streefland, M. (2014). Multivariate pat solutions for biopharmaceutical cultivation: Current progress and limitations. *Trends in Biotechnology*, *32*(6), 329–336.

Merck KGaA. (2024).

Metta, N., Ramachandran, R., & Ierapetritou, M. (2021). A novel adaptive sampling based methodology for feasible region identification of compute intensive models using artificial neural network. *AIChE Journal*, *67*(2), e17095.

Michaelis, L., & Menten, M. L. (1913). Die kinetik der invertinwirkung. *Biochemische Zeitschrift*, *49*, 333–3669.

Michalik, C., Chachuat, B., & Marquardt, W. (2009). Incremental global parameter estimation in dynamical systems. *Industrial and Engineering Chemistry Research*, *48*(11), 5489–5497.

Migdalas, A., Pardalos, P. M., Värbrand, P., Pardalos, P. M., & Horst, R. (Eds.). (1998). *Multilevel optimization: Algorithms and applications* (Vol. 20). Springer US.

Min, Q., Lu, Y., Liu, Z., Su, C., & Wang, B. (2019). Machine learning based digital twin framework for production optimization in petrochemical industry. *International Journal of Information Management*, *49*, 502–519.

Mirjalili, S. (2019). Genetic algorithm. In *Soft computing and intelligent systems* (pp. 43–55). Elsevier.

Miró, A., Pozo, C., Guillén-Gosálbez, G., Egea, J. A., & Jiménez, L. (2012). Deterministic global optimization algorithm based on outer approximation for the parameter estimation of nonlinear dynamic biological systems. *BMC Bioinformatics*, *13*(1), 90.

Miró, A. (2014). *Dynamic mathematical tools for the identification of regulatory structures and kinetic parameters in systems biology (doctoral dissertation)* [Doctoral dissertation, Rovira I Virgili University, Tarragona, Spain].

Misener, R., & Floudas, C. A. (2014). Antigone: Algorithms for continuous / integer global optimization of nonlinear equations. *Journal of Global Optimization*, *59*(2-3), 503–526.

Mišković, L., & Hatzimanikatis, V. (2011). Modeling of uncertainties in biochemical reactions. *Biotechnology and Bioengineering*, *108*(2), 413–423.

Mitsos, A., Chachuat, B., & Barton, P. I. (2009). Mccormick-based relaxations of algorithms. *SIAM Journal on Optimization*, *20*(2), 573–601.

Močkus, J. (1975). On bayesian methods for seeking the extremum. In G. Goos, J. Hartmanis, P. Brinch Hansen, D. Gries, C. Moler, G. Seegmüller, N. Wirth, & G. I. Marchuk (Eds.), *Optimization techniques ifip technical conference novosibirsk, july 1–7, 1974* (pp. 400–404, Vol. 27). Springer Berlin Heidelberg.

Morari, M., Grimm, W., Oglesby, M. J., & Prosser, I. D. (1985). Design of resilient processing plants—vii. design of energy management system for unstable reactors—new insights. *Chemical Engineering Science*, *40*(2), 187–198.

Mosavi, A., Shamshirband, S., Salwana, E., Chau, K.-w., & Tah, J. H. M. (2019). Prediction of multi-inputs bubble column reactor using a novel hybrid model of computational fluid dynamics and machine learning. *Engineering Applications of Computational Fluid Mechanics*, *13*(1), 482–492.

Mowbray, M., Savage, T., Wu, C., Song, Z., Cho, B. A., Del Rio Chanona, E. A., & Zhang, D. (2021). Machine learning for biochemical engineering: A review. *Biochemical Engineering Journal*, *172*(February), 108054.

Mowbray, M. R., Wu, C., Rogers, A. W., Rio-Chanona, E. A. D., & Zhang, D. (2023). A reinforcement learning-based hybrid modeling framework for bioprocess kinetics identification. *Biotechnology and Bioengineering*, *120*(1), 154–168.

Müller, A. C., & Guido, S. (2017). *Introduction to machine learning with python*. O'Reilly Media Inc.

Müller, J. M., Buliga, O., & Voigt, K.-I. (2018). Fortune favors the prepared: How smes approach business model innovations in industry 4.0. *Technological Forecasting and Social Change*, *132*, 2–17.

Murphy, K. P. (2013). *Machine learning: A probabilistic perspective*. The MIT Press.

Narayanan, H., Luna, M., Sokolov, M., Arosio, P., Butté, A., & Morbidelli, M. (2021). Hybrid models based on machine learning and an increasing degree of process knowledge: Application to capture chromatographic step. *Industrial and Engineering Chemistry Research*, *60*(29), 10466–10478.

Narayanan, H., Luna, M. F., von Stosch, M., Cruz Bournazou, M. N., Polotti, G., Morbidelli, M., Butté, A., & Sokolov, M. (2020). Bioprocessing in the digital age: The role of process models. *Biotechnology Journal*, *15*(1), 1–10.

Narayanan, H., Seidler, T., Luna, M. F., Sokolov, M., Morbidelli, M., & Butté, A. (2021). Hybrid models for the simulation and prediction of chromatographic processes for protein capture. *Journal of Chromatography A*, *1650*, 462248.

Narayanan, H., Sokolov, M., Morbidelli, M., & Butté, A. (2019). A new generation of predictive models: The added value of hybrid models for manufacturing processes of therapeutic proteins. *Biotechnology and Bioengineering*, *116*(10), 2540–2549.

Narayanan, H., von Stosch, M., Feidl, F., Sokolov, M., Morbidelli, M., & Butté, A. (2023). Hybrid modeling for biopharmaceutical processes: Advantages, opportunities, and implementation. *Frontiers in Chemical Engineering*, *5*.

Negri, V., Vázquez, D., Sales-Pardo, M., Guimerà, R., & Guillén-Gosálbez, G. (2022). Bayesian symbolic learning to build analytical correlations from rigorous process simulations: Application to co $_2$ capture technologies. *ACS Omega*, *7*(45), 41147–41164.

Nelder, J. A., & Mead, R. (1965). A simplex method for function minimization. *The Computer Journal*, *7*(4), 308–313.

Nestler, F., Schütze, A., Ouda, M., Hadrich, M., Schaadt, A., Bajohr, S., & Kolb, T. (2020). Kinetic modelling of methanol synthesis over commercial catalysts: A critical assessment. *Chemical Engineering Journal*, *394*, 124881.

Neumann, P., Cao, L., Russo, D., Vassiliadis, V. S., & Lapkin, A. A. (2020). A new formulation for symbolic regression to identify physico-chemical laws from experimental data. *Chemical Engineering Journal*, *387*(June 2019), 123412.

Nikolopoulou, A., & Ierapetritou, M. G. (2012). Optimal design of sustainable chemical processes and supply chains: A review. *Computers and Chemical Engineering*, *44*, 94–103.

Nocedal, J., & Wright, S. (2006). *Numerical optimization*. Springer, New York, NY.

Ochoa, M. P., & Grossmann, I. E. (2020). Novel minlp formulations for flexibility analysis for measured and unmeasured uncertain parameters. *Computers and Chemical Engineering*, *135*, 106727.

Olson, G. B. (1997). Computational design of hierarchically structured materials. *Science*, *277*(5330), 1237–1242.

Ostertagová, E. (2012). Modelling using polynomial regression. *Procedia Engineering*, *48*, 500–506.

Ostrovsky, G. M., Volin, Y. M., Barit, E. I., & Senyavin, M. M. (1994). Flexibility analysis and optimization of chemical plants with uncertain parameters. *Computers and Chemical Engineering*, *18*(8), 755–767.

Ovalle, D., Vyas, J., Laird, C. D., & Grossmann, I. E. (2024). Integration of plant scheduling feasibility with supply chain network under disruptions using machine learning surrogates. In *Computer aided chemical engineering* (pp. 1489–1494, Vol. 53). Elsevier.

Pal, S., Cho, B. A., Chanona, A. D. R., Zhang, D., & Mowbray, M. (2024). Bayesian hybrid models for simulation of microbial biohydrogen photo-production processes. In *Computer aided chemical engineering* (pp. 85–90, Vol. 53). Elsevier.

Pantelides, C. C., & Pereira, F. E. (2024). The future of digital applications in pharmaceutical operations. *Current Opinion in Chemical Engineering*, *45*, 101038.

Papageorgiou, L. G., Rotstein, G. E., & Shah, N. (2001). Strategic supply chain optimization for the pharmaceutical industries. *Industrial and Engineering Chemistry Research*, *40*(1), 275–286.

Parkinson, B., Balcombe, P., Speirs, J. F., Hawkes, A. D., & Hellgardt, K. (2019). Levelized cost of co2 mitigation from hydrogen production routes. *Energy and Environmental Science*, *12*(1), 19–40.

Pasupa, K., & Sunhem, W. (2016). A comparison between shallow and deep architecture classifiers on small dataset. *2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE)*, 1–6.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., ... Chintala, S. (2019). *Pytorch: An imperative style, high-performance deep learning library*. Curran Associates, Inc.

Paulson, J., & Lu, C. (2021). The cobalt algorithm for constrained grey-box optimization of computationally expensive models.

Paulson, J. A., & Lu, C. (2022). Cobalt: Constrained bayesian optimization of computationally expensive grey-box models exploiting derivative information. *Computers and Chemical Engineering*, *160*, 107700.

Pavlišič, A., Huš, M., Prašnikar, A., & Likozar, B. (2020). Multiscale modelling of co2 reduction to methanol over industrial cu/zno/al2o3 heterogeneous catalyst: Linking ab initio surface reaction kinetics with reactor fluid dynamics. *Journal of Cleaner Production*, *275*, 122958.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Édouard, D. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, *12*(85), 2825–2830.

Perera, A., Wickramasinghe, P., Nik, V. M., & Scartezzini, J.-L. (2019). Machine learning methods to assist energy system optimization. *Applied Energy*, *243*, 191–205.

Petkov, S. B., & Maranas, C. D. (1997). Multiperiod planning and scheduling of multiproduct batch plants under demand uncertainty. *Industrial and Engineering Chemistry Research*, *36*(11), 4864–4881.

Petrillo, A., Felice, F. D., Cioffi, R., & Zomparelli, F. (2018). Fourth industrial revolution: Current practices, challenges, and opportunities. In A. Petrillo, R. Cioffi, & F. D. Felice (Eds.), *Digital transformation in smart manufacturing*. InTech.

Petsagkourakis, P., Sandoval, I. O., Bradford, E., Zhang, D., & Del Rio Chanona, E. A. (2020). Constrained reinforcement learning for dynamic optimization under uncertainty. *IFAC-PapersOnLine*, *53*(2), 11264–11270.

Petsagkourakis, P., Sandoval, I., Bradford, E., Zhang, D., & Del Rio Chanona, E. A. (2020). Reinforcement learning for batch bioprocess optimization. *Computers and Chemical Engineering*, *133*, 106649.

Philbeck, T., & Davis, N. (2018). The fourth industrial revolution: Shaping a new era. *Journal of International Affairs*, *72*(1), 17–2.

Pistikopoulos, E. N. (1995). Uncertainty in process design and operations. *Computers and Chemical Engineering*, *19*, 553–563.

Pistikopoulos, E. N., Barbosa-Povoa, A., Lee, J. H., Misener, R., Mitsos, A., Reklaitis, G. V., Venkatasubramanian, V., You, F., & Gani, R. (2021). Process systems engineering – the generation next? *Computers and Chemical Engineering*, *147*, 107252.

Pistikopoulos, E. N., & Ierapetritou, M. G. (1995). Novel approach for optimal process design under uncertainty. *Computers and Chemical Engineering*, *19*(10), 1089–1110.

Pistikopoulos, E. N., & Mazzuchi, T. A. (1990). A novel flexibility analysis approach for processes with stochastic parameters. *Computers and Chemical Engineering*, *14*(9), 991–1000.

Potvin, G., Ahmad, A., & Zhang, Z. (2012). Bioprocess engineering aspects of heterologous protein production in pichia pastoris: A review. *Biochemical Engineering Journal*, *64*, 91–105.

Prekopa, A. (1995). *Stochastic programming*. Springer Netherlands.

Price, K. V. (2013). Differential evolution. In I. Zelinka, V. Snášel, & A. Abraham (Eds.), *Handbook of optimization* (pp. 187–214, Vol. 38). Springer Berlin Heidelberg.

Psichogios, D. C., & Ungar, L. H. (1992). A hybrid neural network-first principles approach to process modeling. *AIChE Journal*, *38*(10), 1499–1511.

Pulsipher, J. L., Rios, D., & Zavala, V. M. (2019). A computational framework for quantifying and analyzing system flexibility. *Computers and Chemical Engineering*, *126*, 342–355.

Puranik, Y., & Sahinidis, N. V. (2017). Domain reduction techniques for global nlp and minlp optimization. *Constraints*, *22*(3), 338–376.

Qin, S. J. (2014). Process data analytics in the era of big data. *AIChE Journal*, *60*(9), 3092–3100.

Quaghebeur, W., Torfs, E., De Baets, B., & Nopens, I. (2022). Hybrid differential equations: Integrating mechanistic and data-driven techniques for modelling of water systems. *Water Research*, *213*, 118166.

Quirante, N., & Caballero, J. A. (2016). Large scale optimization of a sour water stripping plant using surrogate models. *Computers and Chemical Engineering*, *92*, 143–162.

Quirante, N., Javaloyes, J., & Caballero, J. A. (2015). Rigorous design of distillation columns using surrogate models based on kriging interpolation. *AIChE Journal*, *61*(7), 2169–2187.

Quirante, N., Javaloyes-Antón, J., & Caballero, J. A. (2018). Hybrid simulation-equation based synthesis of chemical processes. *Chemical Engineering Research and Design*, *132*, 766–784.

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian processes for machine learning*. MIT Press.
OCLC: ocm61285753.

Reis, M. S., & Saraiva, P. M. (2021). Data-centric process systems engineering: A push towards pse 4.0. *Computers and Chemical Engineering*, *155*, 107529.

Renardy, M., Yi, T.-M., Xiu, D., & Chou, C.-S. (2018). Parameter uncertainty quantification using surrogate models applied to a spatial model of yeast mating polarization (M. Johnson, Ed.). *PLOS Computational Biology*, *14*(5), e1006181.

Rivera, E. C., Costa, A. C., Andrade, R. R., Atala, D. I. P., Maugeri, F., & Maciel Filho, R. (2007). Development of adaptive modeling techniques to describe the temperature-dependent kinetics of biotechnological processes. *Biochemical Engineering Journal*, *36*(2), 157–166.

Rodriguez-Fernandez, M., Egea, J. A., & Banga, J. R. (2006). Novel metaheuristic for parameter estimation in nonlinear dynamic biological systems. *BMC Bioinformatics*, *7*(1), 483.

Rogers, A., & Ierapetritou, M. (2015a). Feasibility and flexibility analysis of black-box processes part 1: Surrogate-based feasibility analysis. *Chemical Engineering Science*, *137*, 986–1004.

Rogers, A., & Ierapetritou, M. (2015b). Feasibility and flexibility analysis of black-box processes part 2: Surrogate-based flexibility analysis. *Chemical Engineering Science*, *137*, 1005–1013.

Rosa, J. P. S., Guerra, D. J. D., Horta, N. C. G., Martins, R. M. F., & Lourenço, N. C. C. (2020). Overview of artificial neural networks. In *Springerbriefs in applied sciences and technology* (pp. 21–44). Springer, Cham.

Rosafalco, L., Conti, P., Manzoni, A., Mariani, S., & Frangi, A. (2024). Ekf-sindy: Empowering the extended kalman filter with sparse identification of nonlinear dynamics.

Rosen, R., Von Wichert, G., Lo, G., & Bettenhausen, K. D. (2015). About the importance of autonomy and digital twins for the future of manufacturing. *IFAC-PapersOnLine*, *28*(3), 567–572.

Ryoo, H., & Sahinidis, N. V. (1995). Global optimization of nonconvex nlps and minlps with applications in process design. *Computers and Chemical Engineering*, *19*(5), 551–566.

Sachio, S., Ward, A., Pini, R., & Papathanasiou, M. M. (2023). Operability-economics trade-offs in adsorption-based co2 capture process.

Sachio, S., Ward, A., Pini, R., & Papathanasiou, M. M. (2024). Operability-economics trade-offs in adsorption-based co2 capture processes. *Communications Engineering*, *3*(1), 94.

Sadino-Riquelme, M. C., Rivas, J., Jeison, D., Hayes, R. E., & Donoso-Bravo, A. (2020). Making sense of parameter estimation and model simulation in bioprocesses. *Biotechnology and Bioengineering*, *117*(5), 1357–1366.

Sahinidis, N. V. (1996). Baron: A general purpose global optimization software package. *Journal of Global Optimization*, *8*(2), 201–205.

Sahinidis, N. V. (2004). Optimization under uncertainty: State-of-the-art and opportunities. *Computers and Chemical Engineering*, *28*(6-7), 971–983.

Sansana, J., Joswiak, M. N., Castillo, I., Wang, Z., Rendall, R., Chiang, L. H., & Reis, M. S. (2021). Recent trends on hybrid modeling for industry 4.0. *Computers and Chemical Engineering*, *151*, 107365.

Sargent, R. W. H. (1983). Advances in modelling and analysis of chemical process systems. *Computers and Chemical Engineering*, *7*(4), 219–237.

Sargent, R. (1967). Integrated design and optimization of processes. *Chemical Engineering Progress*, *63*(9), 71–78.

Sargent, R. (2004). Introduction: 25 years of progress in process systems engineering. *Computers and Chemical Engineering*, *28*(4), 437–439.

Sartorius AG. (2024).

Savageau, M. A. (1969a). Biochemical systems analysis. *Journal of Theoretical Biology*, *25*(3), 365–369.

Savageau, M. A. (1969b). Biochemical systems analysis. *Journal of Theoretical Biology*, *25*(3), 370–379.

Savageau, M. A. (1970). Biochemical systems analysis. *Journal of Theoretical Biology*, *26*(2), 215–226.

Savitzky, A., & Golay, M. (1964). Smoothing and differentiation. *Anal. Chem*, *36*(8), 1627–1639.

Scearce-Levie, K., Sanchez, P. E., & Lewcock, J. W. (2020). Leveraging preclinical models for the development of alzheimer disease therapeutics. *Nature Reviews Drug Discovery*, *19*(7), 447–462.

Schmidt, M., & Lipson, H. (2009). Distilling free-form natural laws from experimental data. *Science*, *324*(5923), 81–85.

Schoeberl, B., Eichler-Jonsson, C., Gilles, E. D., & Müller, G. (2002). Computational modeling of the dynamics of the map kinase cascade activated by surface and internalized egf receptors. *Nature Biotechnology*, *20*(4), 370–375.

Schweidtmann, A. M., Bongartz, D., Grothe, D., Kerkenhoff, T., Lin, X., Najman, J., & Mitsos, A. (2021). Deterministic global optimization with gaussian processes embedded. *Mathematical Programming Computation*, *13*(3), 553–581.

Schweidtmann, A. M., Huster, W. R., Lüthje, J. T., & Mitsos, A. (2019). Deterministic global process optimization: Accurate (single-species) properties via artificial neural networks. *Computers and Chemical Engineering*, *121*, 67–74.

Schweidtmann, A. M., & Mitsos, A. (2019). Deterministic global optimization with artificial neural networks embedded. *Journal of Optimization Theory and Applications*, *180*(3), 925–948.

Schweidtmann, A. M., Netze, L., & Mitsos, A. (2020). Melon: Machine learning models for optimization.

Scotti, L., Basoalto, H., Moffat, J., & Cogswell, D. (2023). Review of material modeling and digitalization in industry: Barriers and perspectives. *Integrating Materials and Manufacturing Innovation*, *12*(4), 397–420.

Seber, P., & Braatz, R. D. (2024). Lcen: A novel feature selection algorithm for nonlinear, interpretable machine learning models.

Sha, S., Huang, Z., Wang, Z., & Yoon, S. (2018). Mechanistic modeling and applications for cho cell culture development and production. *Current Opinion in Chemical Engineering*, *22*, 54–61.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & de Freitas, N. (2016). Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, *104*(1), 148–175.

Shapiro, A., Dentcheva, D., & Ruszczynski, A. P. (2021). *Lectures on stochastic programming: Modeling and theory* (Third edition). Society for Industrial; Applied Mathematics ; Mathematical Optimization Society.

Sharifian, S., Sotudeh-Gharebagh, R., Zarghami, R., Tanguy, P., & Mostoufi, N. (2021). Uncertainty in chemical process systems engineering: A critical review. *Reviews in Chemical Engineering*, *37*(6), 687–714.

Shimoni, Y., Goudar, C., Jenne, M., & Srinivasan, V. (2014). Qualification of scale-down bioreactors. *Bioprocess International*.

Silva, B. M. D., Champion, K., Quade, M., Loiseau, J.-C., Kutz, J. N., & Brunton, S. L. (2020). Pysindy: A python package for the sparse identification of nonlinear dynamical systems from data. *Journal of Open Source Software*, *5*(49), 2104.

Sinnott, R., & Towler, G. (2020). *Chemical engineering design* (6th Edition). Butterworth-Heinemann.

Slotboom, Y., Bos, M. J., Pieper, J., Vrieswijk, V., Likozar, B., Kersten, S. R. A., & Brilman, D. W. F. (2020). Critical assessment of steady-state kinetic models for the synthesis of methanol over an industrial cu/zno/al2o3 catalyst. *Chemical Engineering Journal*, *389*, 124181.

Smith, E. M., & Pantelides, C. C. (1999). A symbolic reformulation/spatial branch-and-bound algorithm for the global optimisation of nonconvex minlps. *Computers and Chemical Engineering*, *23*(4-5), 457–478.

Smith, E. M. B., & Pantelides, C. C. (1997). Global optimisation of nonconvex minlps. *Computers and Chemical Engineering*, *21*, S791–S796.

Smith, R. (2005). *Chemical process design and integration*. Wiley.

Smith, R. (2016). *Chemical process design and integration* (Second edition). Wiley.

Sorribas, A., Hernández-Bermejo, B., Vilaprinyo, E., & Alves, R. (2007). Cooperativity and saturation in biochemical networks: A saturable formalism using taylor series approximations. *Biotechnology and Bioengineering*, *97*(5), 1259–1277.

Sorribas, A., Pozo, C., Vilaprinyo, E., Guillén-Gosálbez, G., Jiménez, L., & Alves, R. (2010). Optimization and evolution in metabolic pathways: Global optimization techniques in generalized mass action models. *Journal of Biotechnology*, *149*(3), 141–153.

Srisuma, P., Barbastathis, G., & Braatz, R. D. (2024). Mechanistic modeling and analysis of thermal radiation in conventional, microwave-assisted, and hybrid freeze drying for biopharmaceutical manufacturing. *International Journal of Heat and Mass Transfer*, *221*, 125023.

Stephanopoulos, G., & Reklaitis, G. V. (2011). Process systems engineering: From solvay to modern bio- and nanotechnology. *Chemical Engineering Science*, *66*(19), 4272–4306.

Steurtewagen, B., & Van Den Poel, D. (2020). Machine learning refinery sensor data to predict catalyst saturation levels. *Computers and Chemical Engineering*, *134*, 106722.

Straub, D. A., & Grossmann, I. E. (1990). Integrated stochastic metric of flexibility for systems with discrete state and continuous parameter uncertainties. *Computers and Chemical Engineering*, *14*(9), 967–985.

Straub, D. A., & Grossmann, I. E. (1993). Design optimization of stochastic flexibility. *Computers and Chemical Engineering*, *17*(4), 339–354.

Sun, J., Garibaldi, J. M., & Hodgman, C. (2012). Parameter estimation using metaheuristics in systems biology: A comprehensive review. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *9*(1), 185–202.

Sun, W., & Braatz, R. D. (2020). Alven: Algebraic learning via elastic net for static and dynamic nonlinear model identification. *Computers and Chemical Engineering*, *143*, 107103.

Susarla, N., & Karimi, I. A. (2011). Integrated campaign planning and resource allocation in batch plants. *Computers and Chemical Engineering*, *35*(12), 2990–3001.

Swaney, R. E., & Grossmann, I. E. (1985a). An index for operational flexibility in chemical process design. part i: Formulation and theory. *AIChE Journal*, *31*(4), 621–630.

Swaney, R. E., & Grossmann, I. E. (1985b). An index for operational flexibility in chemical process design. part ii: Computational algorithms. *AIChE Journal*, *31*(4), 631–641.

Tawarmalani, M., & Sahinidis, N. V. (2002). *Convexification and global optimization in continuous and mixed-integer nonlinear programming* (Vol. 65). Springer US.

Tawarmalani, M., & Sahinidis, N. V. (2005). A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, *103*(2), 225–249.

Taylor, C. J., Booth, M., Manson, J. A., Willis, M. J., Clemens, G., Taylor, B. A., Chamberlain, T. W., & Bourne, R. A. (2021). Rapid, automated determination of reaction models and kinetic parameters. *Chemical Engineering Journal*, *413*, 127017.

The MathWorks Inc. (2020). Matlab r2020a.

The MathWorks Inc. (2024). Matlab r2024a.

Tjoa, I. B., & Biegler, L. T. (1991). Simultaneous solution and optimization strategies for parameter estimation of differential-algebraic equation systems. *Industrial and Engineering Chemistry Research*, *30*(2), 376–385.

Tonner, P. D., Darnell, C. L., Engelhardt, B. E., & Schmid, A. K. (2017). Detecting differential growth of microbial populations with gaussian process regression. *Genome Research*, *27*(2), 320–333.

Tsionas, M. G., & Assaf, A. G. (2020). Symbolic regression for better specification. *International Journal of Hospitality Management*, *91*(June).

Tsopanoglou, A., & Jiménez Del Val, I. (2021). Moving towards an era of hybrid modelling: Advantages and challenges of coupling mechanistic and data-driven models for upstream pharmaceutical bioprocesses. *Current Opinion in Chemical Engineering*, *32*, 100691.

TuringBot. (2023). Symbolic regression software.

Turton, R., Shaeiwitz, J. A., Bhattacharyya, D., & Whiting, W. B. (2018). *Analysis, synthesis and design of chemical processes*. Pearson Prentice Hall.

Udrescu, S. M., & Tegmark, M. (2019). Ai feynman: A physics-inspired method for symbolic regression. *arXiv*, (1).

Van Dam, D., & De Deyn, P. P. (2011). Animal models in the drug discovery pipeline for alzheimer's disease. *British Journal of Pharmacology*, *164*(4), 1285–1300.

Van De Berg, D., Jimbo, R. X. J., Shah, N., & Del Rio Chanona, E. A. (2023). Tractable data-driven solutions to hierarchical planning-scheduling-control. In *Computer aided chemical engineering* (pp. 649–654, Vol. 52). Elsevier.

van de Berg, D., Petsagkourakis, P., Shah, N., & Del Rio Chanona, E. A. (2023). Data-driven coordination of subproblems in enterprise-wide optimization under organizational considerations. *AIChE Journal*, *69*(4), e17977.

van de Berg, D., Shah, N., & Del Rio Chanona, E. A. (2023). Hierarchical planning-scheduling-control – optimality surrogates and derivative-free optimization.

Van Den Heuvel, W.-J., & Tamburri, D. A. (2020). Model-driven ml-ops for intelligent enterprise applications: Vision, approaches and challenges. In B. Shishkov (Ed.), *Business modeling and software design* (pp. 169–181, Vol. 391). Springer International Publishing.

Van Laarhoven, P. J. M., & Aarts, E. H. L. (1987). Simulated annealing. In *Simulated annealing: Theory and applications* (pp. 7–15). Springer Netherlands.

Van-Dal, É. S., & Bouallou, C. (2013). Design and simulation of a methanol production plant from co2 hydrogenation. *Journal of Cleaner Production*, *57*, 38–45.

Vanden Bussche, K. M., & Froment, G. F. (1996). A steady-state kinetic model for methanol synthesis and the water gas shift reaction on a commercial cu/zno/al2o3 catalyst. *Journal of Catalysis*, *161*(1), 1–10.

Vázquez, D., & Guillén-Gosálbez, G. (2021). Process design within planetary boundaries: Application to co2 based methanol production. *Chemical Engineering Science*, *246*, 116891.

Vázquez, D., Guimerà, R., Sales-Pardo, M., & Guillén-Gosálbez, G. (2022). Automatic modeling of socioeconomic drivers of energy consumption and pollution using bayesian symbolic regression. *Sustainable Production and Consumption*, *30*, 596–607.

Venkatasubramanian, V., Rengaswamy, R., & Ka, S. N. (2003). A review of process fault detection and diagnosis part ii: Qualitative models and search strategies. *Computers and Chemical Engineering*.

Venkatasubramanian, V., Rengaswamy, R., Yin, K., & Ka, S. N. (2003). A review of process fault detection and diagnosis part i: Quantitative model-based methods. *Computers and Chemical Engineering*.

Vladislavleva, E., Friedrich, T., Neumann, F., & Wagner, M. (2013). Predicting the energy output of wind farms based on weather data: Important variables and their correlation. *Renewable Energy*, *50*, 236–243.

Voit, E. O. (2000). *Computational analysis of biochemical systems : A practical guide for biochemists and molecular biologists*. Cambridge University Press.

Voit, E. O. (2013). Biochemical systems theory: A review. *ISRN Biomathematics*, *2013*, 1–53.

Voit, E. O., & Almeida, J. (2004). Decoupling dynamical systems for pathway identification from metabolic profiles. *Bioinformatics*, *20*(11), 1670–1681.

von Stosch, M., Oliveira, R., Peres, J., & Feyo de Azevedo, S. (2014). Hybrid semi-parametric modeling in process systems engineering: Past, present and future. *Computers and Chemical Engineering*, *60*, 86–101.

Von Stosch, M., Portela, R. M., & Varsakelis, C. (2021). A roadmap to ai-driven in silico process development: Bioprocessing 4.0 in practice. *Current Opinion in Chemical Engineering*, *33*, 100692.

Walsh, G. (2018). Biopharmaceutical benchmarks 2018. *Nature Biotechnology*, *36*(12), 1136–1145.

Wan, X., Pekny, J. F., & Reklaitis, G. V. (2005). Simulation-based optimization with surrogate models—application to supply chain management. *Computers and Chemical Engineering*, *29*(6), 1317–1328.

Wang, S., Fan, K., Luo, N., Cao, Y., Wu, F., Zhang, C., Heller, K. A., & You, L. (2019). Massive computational acceleration by using neural networks to emulate mechanism-based biological models. *Nature Communications*, *10*(1), 4354.

Wang, Z., & Ierapetritou, M. (2017). A novel feasibility analysis method for black-box processes using a radial basis function adaptive sampling approach. *AIChE Journal*, *63*(2), 532–550.

Weng, B., Song, Z., Zhu, R., Yan, Q., Sun, Q., Grice, C. G., Yan, Y., & Yin, W.-J. (2020). Simple descriptor derived from symbolic regression accelerating the discovery of new perovskite catalysts. *Nature Communications*, *11*(1), 3513.

Wernet, G., Bauer, C., Steubing, B., Reinhard, J., Moreno-Ruiz, E., & Weidema, B. (2016). The ecoinvent database version 3 (part i): Overview and methodology. *The International Journal of Life Cycle Assessment*, *21*(9), 1218–1230.

Westerberg, A. W. (2004). A retrospective on design and process synthesis. *Computers and Chemical Engineering*, *28*(4), 447–458.

Willis, M. J., Montague, G., & Peel, C. (1995). On the application of artificial neural networks to process control. In *Applications of neural networks* (pp. 191–219).

Willis, M. J., & von Stosch, M. (2017). Simultaneous parameter identification and discrimination of the nonparametric structure of hybrid semi-parametric models. *Computers and Chemical Engineering*, *104*, 366–376.

Wilson, Z. T., & Sahinidis, N. V. (2017). The alamo approach to machine learning. *Computers and Chemical Engineering*, *106*, 785–795.

Wilson, Z. T., & Sahinidis, N. V. (2019). Automated learning of chemical reaction networks. *Computers and Chemical Engineering*, *127*, 88–98.

World Economic Forum. (2022). The data-driven journey towards manufacturing excellence.

Yee, T., & Grossmann, I. (1990). Simultaneous optimization models for heat integration—ii. heat exchanger network synthesis. *Computers and Chemical Engineering*, *14* (10), 1165–1184.

Zamora, J. M., & Grossmann, I. E. (1999). A branch and contract algorithm for problems with concave univariate, bilinear and linear fractional terms. *Journal of Global Optimization*, *14* (3), 217–249.

Zhang, D., Dechatiwongse, P., del Rio-Chanona, E. A., Maitland, G. C., Hellgardt, K., & Vassiliadis, V. S. (2015). Modelling of light and temperature influences on cyanobacterial growth and biohydrogen production. *Algal Research*, *9*, 263–274.

Zhang, D., Dechatiwongse, P., Del Rio Chanona, E. A., Hellgardt, K., Maitland, G. C., & Vassiliadis, V. S. (2015). Analysis of the cyanobacterial hydrogen photoproduction process via model identification and process simulation. *Chemical Engineering Science*, *128*, 130–146.

Zhang, D., Del Rio Chanona, E. A., Petsagkourakis, P., & Wagner, J. (2019). Hybrid physics-based and data-driven modeling for bioprocess online simulation and optimization. *Biotechnology and Bioengineering*, *116* (11), 2919–2930.

Zhang, D., Savage, T. R., & Cho, B. A. (2020). Combining model structure identification and hybrid modelling for photo-production process predictive simulation and optimisation. *Biotechnology and Bioengineering*, *117* (11), 3356–3367.

Zhang, Q., & Grossmann, I. E. (2016). Enterprise-wide optimization for industrial demand side management: Fundamentals, advances, and perspectives. *Chemical Engineering Research and Design*, *116*, 114–131.

Zhang, Q., Grossmann, I. E., & Lima, R. M. (2016). On the relation between flexibility analysis and robust optimization for linear systems. *AIChE Journal*, *62* (9), 3109–3123.

Zhang, S., Androulakis, I. P., & Ierapetritou, M. G. (2013). A hybrid kinetic mechanism reduction scheme based on the on-the-fly reduction and quasi-steady-state approximation. *Chemical Engineering Science*, *93*, 150–162.

Zhou, J. (2013). Digitalization and intelligentization of manufacturing industry. *Advances in Manufacturing*, *1* (1), 1–7.

Zimmermann, H.-J. (2001). *Fuzzy set theory—and its applications*. Springer Netherlands.

Appendix A

Supplementary information of Chapter 2

This part of the appendix contains the supplementary material of the article given in Chapter 2. It is organized as follows. Section A.1 summarizes the parameters used for the *in-silico* data generation (ODE integration). Section A.2 shows the procedure how we scaled the data for CSIV. The heuristic approach how we chose the starting values for the solvers are summarized in Section A.3, where the stopping criteria of the solver is given in Section A.4. Additional results and information about the case studies conclusions are given in Sections A.5, A.6, and A.7. For each individual model, the identified model parameters are indicated in Section A.8. Finally, Section A.9 includes an error comparison of the MINLP, ANN, GP, and SINDy approaches.

A.1 Parameters for underlying *in-silico* models

Table A.1. Constant parameters used in CSI.

Parameter	Value	Unit
k_1	5	h^{-1}
k_2	1	h^{-1}
k_3	2	h^{-1}

Table A.2. Reaction rate constants used in CSII.

Parameter	Value	Unit
k_1	10	h^{-1}
k_2	1	h^{-1}
k_3	0.5	$L\,mol^{-1}\,h^{-1}$

Table A.3. Reaction rate constants used in CSIII.

Parameter	Value	Unit
k_1	0.33384	h^{-1}
k_2	0.26687	h^{-1}
k_3	0.1494	h^{-1}
k_4	0.18957	h^{-1}
k_5	0.009598	h^{-1}
k_6	0.29425	h^{-1}
k_7	0.011932	h^{-1}

Table A.4. Parameters used in CSIV.

Parameter	Value	Unit
ϕ_{max}	0.25	h^{-1}
K_S	105.4	$g\,L^{-1}$
A_1	130	-
A_2	$3.8{\cdot}10^{48}$	-
E_1	12.4	$kJ\,mol^{-1}$
E_2	298.6	$kJ\,mol^{-1}$
K_ϕ	121.9	$g\,L^{-1}$
$Y_{B,S}$	0.07	-
$Y_{P,S}$	0.167	-

A.2 Applied scaling for CSIV

For case study IV with noisy data, the sampled data points $X_{i,u}$ were preprocessed before using them for training the mentioned models. The state values of CSI-III were not preprocessed.

First, a Savitzky-Golay filter (Savitzky & Golay, 1964) was used to denoise the sampled training state data $X_{i,u}$, resulting in the filtered state variables $\tilde{X}_{i,u}$. To perform the filtering, we used a frame size of 7, and a polynomial order of 5. The filtered values $\tilde{X}_{i,u}$ were then further used for the training.

Second, we look for the maximum value of each species i in all the training runs:

$$MX_i = \max_{u \in U} \tilde{X}_{i,u}, \quad \forall i \in I \tag{A.1}$$

Third, the filtered state variables of each species $\tilde{X}_{i,u}$ were scaled with the maximum value available (MX_i):

$$X^*_{i,u} = \frac{\tilde{X}_{i,u}}{MX_i} \tag{A.2}$$

Where $X^*_{i,u}$ rpresent ethe filtered and scaled data points of species i, and time point u.

A.3 Heuristics to choose starting values

Using the non-noisy datasets, we only considered if a species is being produced or consumed over time. If a species i is produced, the corresponding reaction rate constant for the production term α_i is initialized with a value of one. If a species is consumed, the corresponding reaction rate constant of the depletion term β_i is initialized with value one. All other parameters are initialized to zero.

The exponents were initialized to one when the corresponding species are involved in a generation or depletion reaction. In CSI, we knew that the reaction starts with species A and B and that both species were consumed, where species C is first produced, then consumed, and D is only produced. Therefore, we chose $\alpha_i = [0, 0, 1, 1]$, and $\beta_i = [1, 1, 1, 0]$. The exponents of the generation term, $g_{C,A}$, $g_{C,B}$, $g_{D,A}$, and $g_{D,B}$ are initialized to one, since we know that species C and D must be originating from A and B (although not via a direct reaction). The depletion exponents were initialized to one assuming a first-order reaction as follows: $h_{A,A} = h_{B,B} = h_{C,C} = 1$ and $h_{D,D} = 0$ since species D is not consumed (according to the observed profiles). This approach was implemented in CSI-III. In CSIV, we initialized all rate constants (α_i and β_i) to one to account for possible generation/depletion reactions. The exponents were initialized to one for

biomass production from substrate ($g_{B,B}$, $g_{B,S}$), product formation from biomass consuming substrate ($g_{P,B}$, $g_{P,S}$), biomass death that might depend on all metabolites ($h_{B,B}$, $h_{B,S}$, $h^{(2)}$), and substrate consumption by biomass accounting for the amount of product already produced ($h_{S,B}$, $h_{S,S}$, $h^{S,P}$).

Using the noisy datasets, the same heuristics as explained above were applied. For CSIII and IV, the solver was unable to find a feasible solution. Therefore, we proceeded as follows. For CSIII, we used the same initialization as provided in the non-noisy case. However, we solved the optimization problem only for one arbitrary training run. We used the found feasible solution as a new initial point to solve the entire MINLP for all six training runs. For CSIV, we first reduced the value of NP to only eight parameters for the first MINLP. Additionally, the starting points were as follows: $\alpha_i = [0, 0, 1]$, $\beta_i = [1, 1, 0]$, $g_{P,B} = h_{S,B} = 1$, and $h_{B,B} = 1$.

All these chosen initial values are given subsequently in Table A.5 and Table A.6.

Table A.5. Initial values chosen for the parameters to be estimated with the proposed method applied to the non-noisy dataset.

Parameter	Species[a]	CSI				CSII				CSIII					CSIV		
		A	B	C	D	A	B	C	D	A	B	C	D	E	B	S	P
α_i	-	0	0	1	1	0	1	1	1	0	0	1	1	1	1	1	1
β_i	-	1	1	1	0	1	1	0	0	1	1	0	0	0	1	1	1
$g_{i,j}$	A	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
	B	0	0	0	0	1	1	0	0	1	1	0	0	0	0	0	0
	C	1	1	0	0	1	0	0	0	1	1	0	0	0	1	1	0
	D	1	1	0	0	1	0	0	0	1	1	0	0	0	-	-	-
	E	-	-	-	-	-	-	-	-	1	0	0	0	0	-	-	-
$h_{i,j}$	A	1	0	0	0	1	0	0	0	0	0	0	0	0	1	1	1
	B	0	1	0	0	0	1	0	0	0	0	0	0	0	1	1	1
	C	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	D	0	0	0	0	0	0	0	0	0	0	0	0	0	-	-	-
	E	-	-	-	-	-	-	-	-	0	0	0	0	0	-	-	-

[a] Column: species i, row: species j.

Table A.6. Initial values chosen for the parameters to be estimated with the proposed method applied to the noisy dataset.

Parameter	Species[a]	CSI				CSII				CSIII					CSIV		
		A	B	C	D	A	B	C	D	A	B	C	D	E	B	S	P
α_i	-	0	0	1	1	0	1	1	1	0	0	1	1	1	0	0	1
β_i	-	1	1	1	0	1	1	0	0	1	1	0	0	0	1	1	0
$g_{i,j}$	A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	B	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
	C	1	1	0	0	1	1	0	0	1	0	0	0	0	1	0	0
	D	1	1	0	0	1	1	0	0	1	0	0	0	0	-	-	-
	E	-	-	-	-	-	-	-	-	1	0	0	0	0	-	-	-
$h_{i,j}$	A	1	0	0	0	1	1	0	0	1	0	0	0	0	1	0	0
	B	0	1	0	0	1	1	0	0	0	1	0	0	0	1	0	0
	C	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	D	0	0	0	0	0	0	0	0	0	0	0	0	0	-	-	-
	E	-	-	-	-	-	-	-	-	0	0	0	0	0	-	-	-

[a] Column: species i, row: species j.

A.4 Stopping criteria of solver

Table A.7. Termination criteria used for the solver in GAMS. The options not mentioned here were chosen to be the default values.

CS	I	II	III	IV
Relative optimality gap	0	0	0	0
Maximum runtime [s]	5000	5000	5000	600
Maximum iterations	$1{\cdot}10^6$	$1{\cdot}10^6$	$1{\cdot}10^8$	$1{\cdot}10^6$
Maximum nodes	$1{\cdot}10^6$	$1{\cdot}10^6$	$1{\cdot}10^6$	$1{\cdot}10^6$

A.5 Tabulated results

In this section we summarize the error metrics of each individual model.

Table A.8. Summary of the model performance considering an unseen non-noisy test set (state-space). The $RMSE$ values are shown for the S-system and the ANN. The Extr-value indicates the percentage of data points outside the training range. The bold values represent the best model performance for this case study. If several metrics show the same performance for the S-system, the one with the lowest number of parameters is chosen as the best model (most little complexity). For the S-system, the number of non-negative parameters is shown in brackets after the error metric, together with the upper bound for the maximum number of parameters in the MINLP (NP in model (2.11) in the manuscript).

CS	**Extr**	**RMSE**[a]								
		MINLP							**NLP**	**ANN**
I	0.34	$4.3{\cdot}10^{-2}$ (12/12)	$\mathbf{3.3{\cdot}10^{-4}}$ **(14/17)**	$3.3{\cdot}10^{-4}$ (14/32)	$3.3{\cdot}10^{-4}$ (14/40)				$1.7{\cdot}10^{-2}$ (16/40)	$2.2{\cdot}10^{-2}$
II	0.16	$\mathbf{2.6{\cdot}10^{-1}}$ **(11/11)**	$3.1{\cdot}10^{-1}$ (13/15)	$3.1{\cdot}10^{-1}$ (13/20)	$3.1{\cdot}10^{-1}$ (13/30)	$3.1{\cdot}10^{-1}$ (13/40)			$3.2{\cdot}10^{-1}$ (19/40)	$7.0{\cdot}10^{-1}$
III	0.28	$5.1{\cdot}10^{-1}$ (10/10)	$2.0{\cdot}10^{-1}$ (16/16)	$1.8{\cdot}10^{-1}$ (21/21)	$1.8{\cdot}10^{-1}$ (21/24)	$1.8{\cdot}10^{-1}$ (21/30)	$1.8{\cdot}10^{-1}$ (21/40)	$1.8{\cdot}10^{-1}$ (21/60)	$1.8{\cdot}10^{0}$ (31/60)	$\mathbf{8.8{\cdot}10^{-4}}$
IV	0.35	$1.5{\cdot}10^{0}$ (15/15)	$1.3{\cdot}10^{0}$ (15/17)	$1.3{\cdot}10^{0}$ (15/19)	$1.3{\cdot}10^{0}$ (15/21)	$1.3{\cdot}10^{0}$ (15/23)	$1.3{\cdot}10^{0}$ (15/24)		$\mathbf{3.7{\cdot}10^{-1}}$ **(24/24)**	$3.8{\cdot}10^{0}$

[a] $\mathrm{mol\,L^{-1}}$ for CSI, II, III, and $\mathrm{g\,L^{-1}}$ for CSIV.

Table A.9. Summary of the model performance considering an unseen non-noisy test set (derivative-space). The $RMSE$ values are shown for the S-system and the ANN. The Extr-value indicates the percentage of data points outside the training range. The bold values represent the best model performance for this case study. If several metrics show the same performance for the S-system, the one with the lowest number of parameters is chosen as the best model (most little complexity). For the S-system, the number of non-negative parameters is shown in brackets after the error metric, together with the upper bound for the maximum number of parameters in the MINLP or NLP (NP in model (2.11) in the manuscript).

CS	**Extr**	**RMSE**[a]								
		MINLP							**NLP**	**ANN**
I	0.34	$4.2{\cdot}10^{-1}$ (12/12)	$1.6{\cdot}10^{-11}$ (14/17)	$1.6{\cdot}10^{-11}$ (14/32)	$1.6{\cdot}10^{-11}$ (14/40)				**$1.8{\cdot}10^{-12}$ (16/40)**	$1.0{\cdot}10^{0}$
II	0.16	$7.1{\cdot}10^{-1}$ (11/11)	**$5.5{\cdot}10^{-1}$ (13/15)**	$5.5{\cdot}10^{-1}$ (13/20)	$5.5{\cdot}10^{-1}$ (13/30)	$5.5{\cdot}10^{-1}$ (13/40)			$5.9{\cdot}10^{-1}$ (19/40)	$6.1{\cdot}10^{0}$
III	0.28	$3.7{\cdot}10^{-1}$ (10/10)	$1.3{\cdot}10^{-1}$ (16/16)	**$1.1{\cdot}10^{-1}$ (21/21)**	$1.1{\cdot}10^{-1}$ (21/24)	$1.1{\cdot}10^{-1}$ (21/30)	$1.1{\cdot}10^{-1}$ (21/40)	$1.1{\cdot}10^{-1}$ (21/60)	$1.2{\cdot}10^{-1}$ (31/60)	$4.0{\cdot}10^{-1}$
IV	0.35	$7.9{\cdot}10^{-2}$ (15/15)	**$8.2{\cdot}10^{-3}$ (15/17)**	$8.2{\cdot}10^{-3}$ (15/19)	$8.2{\cdot}10^{-3}$ (15/21)	$8.2{\cdot}10^{-3}$ (15/23)	$8.2{\cdot}10^{-3}$ (15/24)		$7.9{\cdot}10^{-2}$ (24/24)	$4.0{\cdot}10^{-1}$

[a] $\mathrm{mol\,L^{-1}\,h^{-1}}$ for CSI, II, III, and $\mathrm{g\,L^{-1}\,h^{-1}}$ for CSIV.

Table A.10. Summary of the model performance considering an unseen non-noisy test set (state-space). The R^2-values are shown for the S-system and the ANN. The Extr-value indicates the percentage of data points outside the training range. The bold values represent the best model performance for this case study. If several metrics show the same performance for the S-system, the one with the lowest number of parameters is chosen as the best model (most little complexity). For the S-system, the number of non-negative parameters is shown in brackets after the error metric, together with the upper bound for the maximum number of parameters in the MINLP (NP in model (2.11) in the manuscript).

CS	**Extr**	R^{2a}								
		MINLP							**NLP**	**ANN**
I	0.34	0.9974 (12/12)	**1.0000 (14/17)**	1.0000 (14/32)	1.0000 (14/32)				0.9996 (16/40)	0.9993
II	0.16	**0.9954 (11/11)**	0.9934 (13/15)	0.9934 (13/20)	0.9934 (13/30)	0.9934 (13/40)			0.9929 (19/40)	0.9662
III	0.28	0.9708 (10/10)	0.9957 (16/16)	0.9965 (21/21)	0.9965 (21/24)	0.9965 (21/30)	0.9965 (21/40)	0.9965 (21/60)	0.6265 (31/60)	**1.0000**
IV	0.35	0.9938 (15/15)	0.9949 (15/17)	0.9949 (15/19)	0.9949 (15/21)	0.9949 (15/23)	0.9949 (15/24)		**0.9996 (24/24)**	0.9584

[a] Unitless.

Table A.11. Summary of the model performance considering an unseen non-noisy test set (derivative-space). The R^2-values are shown for the S-system and the ANN. The Extr-valuea indicates the percentage of data points outside the training range. The bold values represent the best model performance for this case study. If several metrics show the same performance for the S-system, the one with the lowest number of parameters is chosen as the best model (most little complexity). For the S-system, the number of non-negative parameters is shown in brackets after the error metric, together with the upper bound for the maximum number of parameters in the MINLP (NP in model (2.11) in the manuscript).

CS	**Extr**	$R^{2\mathbf{a}}$								
		MINLP							**NLP**	**ANN**
I	0.34	0.9794 (12/12)	**1.0000 (14/17)**	1.0000 (14/32)	1.0000 (14/32)				1.0000 (16/40)	0.8725
II	0.16	0.9996 (11/11)	**0.9997 (13/15)**	0.9997 (13/20)	0.9997 (13/30)	0.9997 (13/40)			0.9997 (19/40)	0.9669
III	0.28	0.9680 (10/10)	0.9957 (16/16)	**0.9970 (21/21)**	0.9970 (21/24)	0.9970 (21/30)	0.9970 (21/40)	0.9970 (21/60)	0.9968 (31/60)	0.9628
IV	0.35	0.9905 (15/15)	**0.9999 (15/17)**	0.9999 (15/19)	0.9999 (15/21)	0.9999 (15/23)	0.9999 (15/24)		0.9906 (24/24)	0.7581

[a] Unitless.

Table A.12. Summary of the model performance considering an unseen noisy test set (state-space). The $RMSE$ values are shown for the S-system and the ANN. The Extr-value indicates the percentage of data points outside the training range. The bold values represent the best model performance for this case study. If several metrics show the same performance for the S-system, the one with the lowest number of parameters is chosen as the best model (most little complexity). For the S-system, the number of non-negative parameters is shown in brackets after the error metric, together with the upper bound for the maximum number of parameters in the MINLP (NP in model (2.11) in the manuscript)

CS	**Extr**	***RMSE*[a]**						
		MINLP					**NLP**	**ANN**
I	0.3	$\mathbf{1.5{\cdot}10^{-1}}$ **(12/12)**	$6.7{\cdot}10^{2}$ (17/17)	$2.8{\cdot}10^{-1}$ (32/32)	$3.1{\cdot}10^{-1}$ (35/40)	-	$3.5{\cdot}10^{-1}$ (30/40)	$1.8{\cdot}10^{-1}$
II	0.14	$3.9{\cdot}10^{-1}$ (17/17)[b]	$\mathbf{3.2{\cdot}10^{-1}}$ **(20/20)[b]**	$3.6{\cdot}10^{-1}$ (30/30)[b]	$3.6{\cdot}10^{-1}$ (35/40)[b]		$4.1{\cdot}10^{-1}$ (30/40)	$4.9{\cdot}10^{-1}$
III	0.28	$2.6{\cdot}10^{-1}$ (15/15)	$2.6{\cdot}10^{-1}$ (15/24)	$2.9{\cdot}10^{-1}$ (29/30)	$2.8{\cdot}10^{-1}$ (38/40)	$2.8{\cdot}10^{-1}$ (38/60)	$2.8{\cdot}10^{-1}$ (45/60)	$\mathbf{1.9{\cdot}10^{-1}}$
IV	0.35	$2.0{\cdot}10^{-1}$ (8/8)	$1.6{\cdot}10^{-1}$ (9/9)	$1.6{\cdot}10^{-1}$ (9/15)	$1.6{\cdot}10^{-1}$ (9/20)	$1.6{\cdot}10^{-1}$ (9/24)	$7.6{\cdot}10^{-1}$ (24/24)	$\mathbf{9.4{\cdot}10^{-2}}$

[a] $\mathrm{mol\,L^{-1}}$ for CSI, II, III, and $\mathrm{g\,L^{-1}}$ for CSIV.

[b] For two batches the solver resulted in NaN predictions. Only four batches considered for error calculation.

Table A.13. Summary of the model performance considering an unseen noisy test set (derivative-space). The RMSE values are shown for the S-system and the ANN. The Extr-value indicates the percentage of data points outside the training range. The bold values represent the best model performance for this case study. If several metrics show the same performance for the S-system, the one with the lowest number of parameters is chosen as the best model (most little complexity). For the S-system, the number of non-negative parameters is shown in brackets after the error metric, together with the upper bound for the maximum number of parameters in the MINLP or NLP (NP in model (2.11) in the manuscript).

CS	**Extr**	***RMSE*[a]**						
		MINLP					**NLP**	**ANN**
I	0.3	$5.5{\cdot}10^{-1}$ (12/12)	$5.0{\cdot}10^{-1}$ (17/17)	$\mathbf{4.8{\cdot}10^{-1}}$ **(32/32)**	$4.8{\cdot}10^{-1}$ (35/40)		$5.2{\cdot}10^{-1}$ (30/40)	$9.9{\cdot}10^{-1}$
II	0.14	$3.7{\cdot}10^{0}$ (17/17)	$\mathbf{3.0{\cdot}10^{0}}$ **(20/20)**	$3.4{\cdot}10^{0}$ (30/30)	$3.4{\cdot}10^{0}$ (35/40)		$3.9{\cdot}10^{0}$ (30/40)	$3.6{\cdot}10^{0}$
III	0.28	$2.0{\cdot}10^{-1}$ (15/15)	$2.0{\cdot}10^{-1}$ (15/24)	$1.9{\cdot}10^{-1}$ (29/30)	$1.9{\cdot}10^{-1}$ (38/40)	$1.9{\cdot}10^{-1}$ (38/60)	$\mathbf{1.8{\cdot}10^{-1}}$ **(45/60)**	$4.1{\cdot}10^{-1}$
IV	0.35	$7.6{\cdot}10^{-3}$ (8/8)	$5.4{\cdot}10^{-3}$ (9/9)	$5.4{\cdot}10^{-3}$ (9/15)	$5.4{\cdot}10^{-3}$ (9/20)	$5.4{\cdot}10^{-3}$ (9/24)	$\mathbf{5.3{\cdot}10^{-3}}$ **(24/24)**	$8.5{\cdot}10^{-3}$

[a] $\mathrm{mol\,L^{-1}\,h^{-1}}$ for CSI, II, III, and $\mathrm{g\,L^{-1}\,h^{-1}}$ for CSIV.

Table A.14. Summary of the model performance considering an unseen noisy test set (state-space). The $R2$-values are shown for the S-system and the ANN. The Extr-value indicates the percentage of data points outside the training range. The bold values represent the best model performance for this case study. If several metrics show the same performance for the S-system, the one with the lowest number of parameters is chosen as the best model (most little complexity). For the S-system, the number of non-negative parameters is shown in brackets after the error metric, together with the upper bound for the maximum number of parameters in the MINLP (NP in model (2.11) in the manuscript).

CS	**Extr**	$\boldsymbol{R^{2}}$[a]						
		MINLP					**NLP**	**ANN**
I	0.3	**0.9689 (12/12)**	<0 (17/17)	0.8957 (32/32)	0.8705 (35/40)	-	0.8311 (30/40)	0.9544
II	0.14	**0.9896 (17/17)**[b]	0.9896 (20/20)[b]	0.987 (30/30)[b]	0.9896 (35/40)[b]	-	0.9881 (30/40)	0.9835
III	0.28	0.9928 (15/15)	0.9928 (15/24)	0.9906 (29/30)	0.9915 (38/40)	0.9915 (38/60)	0.9915 (45/60)	**0.9959**
IV	0.35	0.7377 (8/8)	0.8347 (9/9)	0.8347 (9/15)	0.8347 (9/20)	0.8347 (9/24)	<0 (24/24)	**0.9439**

[a] Unitless.

[b] For two batches the solver resulted in NaN predictions. Only four batches considered for error calculation.

Table A.15. Summary of the model performance considering an unseen noisy test set (derivative-space). The $R2$-values are shown for the S-system and the ANN. The Extr-value indicates the percentage of data points outside the training range. The bold values represent the best model performance for this case study. If several metrics show the same performance for the S-system, the one with the lowest number of parameters is chosen as the best model (most little complexity). For the S-system, the number of non-negative parameters is shown in brackets after the error metric, together with the upper bound for the maximum number of parameters in the MINLP (NP in model (2.11) in the manuscript).

CS	**Extr**	$\boldsymbol{R^{2}}$[a]						
		MINLP					**NLP**	**ANN**
I	0.3	0.9518 (12/12)	0.9594 (17/17)	0.9635 (32/32)	**0.9636 (35/40)**		0.9575 (30/40)	0.8433
II	0.14	0.9551 (17/17)	**0.9706 (20/20)**	0.9616 (30/30)	0.9616 (35/40)		0.9509 (30/40)	0.9569
III	0.28	0.9908 (15/15)	0.9908 (15/24)	0.9917 (29/30)	0.9915 (38/40)	0.9915 (38/60)	**0.9921 (45/60)**	0.9609
IV	0.35	0.7655 (8/8)	0.8808 (9/9)	0.8808 (9/15)	0.8808 (9/20)	0.8808 (9/24)	**0.885 (24/24)**	0.7049

[a] Unitless.

A.6 Model sizes and CPU times

In the following, we indicate the number of equations and number of variables that are required to fully describe the models. Additionally, the required CPU time for the parameter estimation is indicated.

Table A.16. The number of equations and variables are shown that result from implementing the model given in expression (2.11) in the manuscript. In the NLP case, all binaries are set to one, which reduces the number of equations and variables compared to the MINLP cases.

Dataset	CS	MINLP		NLP	
		Equations	**Variables**	**Equations**	**Variables**
Non-noisy	I	619	586	506	545
	II	427	394	314	353
	III	1103	1052	932	991
	IV	447	428	380	403
Noisy	I	619	586	506	545
	II	427	394	314	353
	III	1103	1052	932	991
	IV	447	428	380	403

A.7 Results supplementary information

We next provide more detailed information on the iterations of the algorithm. Again, the term NLP approach refers to the case of an MINLP where all binaries are set to one. In this case, the model is solved according to the procedure shown in Figure 2.4 in the manuscript.

Case Study I - Non-noisy dataset We start with non-noisy data. We first solve the ANN and full-space NLP (NP=40), where in the latter formulation, 14 parameters are found to take a non-zero value. We then solve the MINLP starting with $NP_1 = 12$, which is expected to be too few parameters for finding the true underlying model (14 parameters would be needed in total if expressed by the S-system). Therefore, the error in training (slope space) should be decreasing after increasing the bound to at least NP=14, where the true underlying model should be identifiable by the S-system. The training error should decrease until reaching the next bound, which was set to $NP_2 = 17$, where the true underlying model should be identifiable. Since in practice, one would not know that exactly 14 parameters are required, we set $NP_2 = 17$, instead of $NP_2 = 14$ (same for

Table A.17. Required CPU times (in seconds) of the solver are shown which were needed for the parameter estimation. Times are indicated for each MINLP model candidate and the NLP model. For the MINLP candidate models, the upper bound for the binary variables (NP) is indicated in brackets. CPU times less than 10 seconds were reported accordingly.

Dataset	**CS**	**MINLP**							**NLP**
Non-noisy	I	152 (12)	< 10 (17)	< 10 (32)	< 10 (40)				< 10
	II	81 (11)	< 10 (15)	< 10 (20)	< 10 (30)	< 10 (40)			< 10
	III	682 (10)	369 (16)	40 (21)	< 10 (24)	< 10 (30)	< 10 (40)	< 10 (60)	< 10
	IV	81 (15)	< 10 (17)	< 10 (19)	< 10 (21)	< 10 (23)	< 10 (24)		< 10
Noisy	I	41 (12)	275 (17)	< 10 (32)	< 10 (40)				< 10
	II	21 (17)	1397 (20)	26 (30)	< 10 (40)				< 10
	III	193 (15)	5012 (24)	853 (30)	18 (40)	< 10 (60)			< 10
	IV	126 (8)	< 10 (9)	< 10 (15)	< 10 (20)	< 10 (24)			< 10

$NP_1 = 12$ instead of $NP_1 = 14$).

We note that increasing the upper bound on the number of binaries (NP) that can be one in the MINLP does not necessarily imply that the optimal MINLP solution will hit the said bound. In other words, the MINLP may decide to select fewer parameters when increasing their number further does not result in a lower error in the training set (slope space). For this reason, Figure A.1 shows both the maximum allowed number of parameters and the actual number of non-zero parameters. By looking at the kinetic expressions of the *in-silico* model in Table A.1, it follows that the S-system should be able to model the non-noisy data with zero error using 14 parameters in total. In practice, the solver finds the true expression for $NP \geq 14$ (Table A.18 and Table A.19). On the contrary, the ANN architecture requires 55 parameters to be estimated.

After integration of the trained models, the lowest state-space error in the test set is obtained by the MINLP approach (i.e., $3.27{\cdot}10^{-4}$ mol L^{-1}), followed by the NLP approach (i.e., $1.70{\cdot}10^{-2}$ mol L^{-1}) and the ANN approach (i.e., $2.22{\cdot}10^{-2}$ mol L^{-1}). Table 2.2 in the manuscript and Table A.8 summarize these results.

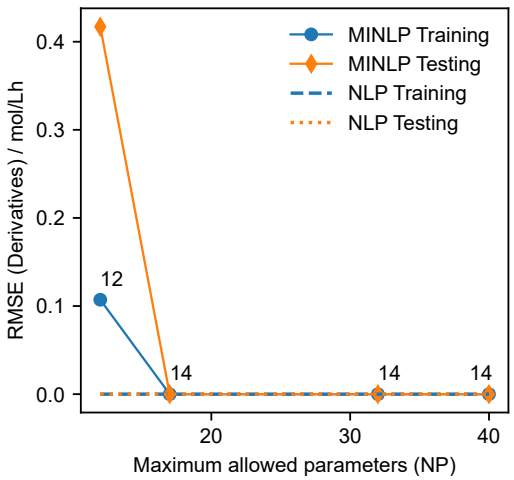

Figure A.1. The $RMSE$ in the slope-space is shown for CSI as a function of different values for NP (maximum allowed non-zero parameters). Non-noisy data were used for model training. On the one hand, the training and test errors of the MINLP are shown as solid blue and orange lines with markers, respectively. The corresponding NLP training and test errors are shown as dashed-dotted lines (overlapping in the plot). The numbers in the plots indicate how many parameters the solver chooses to be non-zero during the training procedure.

Case Study I - Noisy Dataset After integration of the trained models, the

lowest state-space error in the test set is obtained by the MINLP approach (i.e., $1.51 \cdot 10^{-1}$ mol L^{-1}), followed by the ANN approach (i.e., $1.83 \cdot 10^{-1}$ mol L^{-1}) and the NLP approach (i.e., $3.51 \cdot 10^{-1}$ mol L^{-1}). Table 2.3 in the manuscript and Table A.12 summarize these results, where it is emphasized that the MINLP approaches lead to compact model expressions revealing only 12 parameters. The NLP approach requires 35 parameters, while the ANN architecture requires 55 parameters to describe the trained model fully.

In contrast to the non-noisy data, the optimizer can further reduce the training error if more parameters can be non-zero. However, allowing fewer parameters in the model to be non-zero reduces the model complexity, thereby reducing the risk of overfitting. As a conclusion, one can observe that the MINLP approach results in a trained S-system that outperforms the predictive capabilities of the NLP or ANN approach. Figure 2.5 in the manuscript shows a comparison of the MINLP and the ANN model predictions.

In conclusion to this case study, it can be stated that the MINLP approach reveals higher performances if the S-system can fully describe the underlying reaction since the actual model is identified. This is indeed the case for CSI.

Case Study II - Non-Noisy Dataset After performing the integration of the trained models, the lowest state space error in the test set is obtained by the MINLP approach (i.e., $2.58 \cdot 10^{-1}$ mol L^{-1}), followed by the NLP approach (i.e., $3.18 \cdot 10^{-1}$ mol L^{-1}), and the ANN approach (i.e., $6.97 \cdot 10^{-1}$ mol L^{-1}). Table 2 in the manuscript and Table S8 summarize these results. The MINLP and NLP approaches lead to compact model expressions that reveal only 11 and 13 parameters, respectively. The ANN architecture requires again 55 parameters to fully describe the trained model.

For CSII, the S-system will not be able to model the behavior of species A without error. Due to the nature of the model given in equation (2.13) in the manuscript, only one depletion term can be described by an S-system. However, the ODE of species A is characterized by two depletion terms. This is visible in Table A.9, where the training error (slope space) cannot be decreased to zero. It is worth mentioning that this does not directly indicate that the model will not predict this species well. On the other hand, Species B, C, and D can be appropriately modeled by the S-system. To summarize, it can be stated that the ANN approach

achieves better predictive performance in those chemical reaction examples where the S-system cannot fully describe the underlying *in-silico* model.

Case Study II - Noisy Dataset After training and integrating the S-system with the MINLP formulation and the data of CSII, the slope space errors shown in Figure 2.6 could be obtained. The lowest state space error in the test set is obtained by the MINLP approach (i.e., $3.24 \cdot 10^{-1}$ mol L^{-1}), followed by the NLP approach (i.e., $4.13 \cdot 10^{-1}$ mol L^{-1}) and the ANN approach (i.e., $4.87 \cdot 10^{-1}$ mol L^{-1}). Table 2.3 in the manuscript and Table A.12 summarize these results, where it is emphasized that the MINLP approaches lead to compact model expressions revealing only 17 parameters. The NLP approach requires again 35 parameters, where the ANN architecture needs again 55 parameters to describe the trained model fully.

In contrast to the non-noisy data, the optimizer can further reduce the training error if more parameters can be non-zero (Figure 2.6 (a) in the manuscript). However, allowing fewer parameters in the model to be non-zero reduces the model complexity. Additionally, it increases the models' generalization ability and therefore reduces the chance of overfitting. This procedure is visible in Figure 2.6 (b) in the manuscript, where the test error in slope space is first reduced and starts to increase again when allowing more parameters to be non-zero. This clearly fulfills the expectation of overfitting models.

As a conclusion, one can observe that the ANN approach reveals the best performance, where the MINLP approach results in a trained S-system model candidate that outperforms the predictive power of the NLP approach. However, although the ANN approach shows better overall performance, the difference is only marginal, where fewer parameters are required for the MINLP/NLP approaches to model the studied system. Having a compact canonical expression (as it results from the proposed method) at hand makes it easier to perform subsequent tasks, such as an optimization. Since the performance is in a similar range, the availability of a canonical model makes the presented approach more suitable for later usage than the ANN.

Case Study III - Non-noisy dataset Case study III represents a chemical reaction that reveals a slightly higher complexity than before. After model training and integration, the lowest state-space error in the test set is obtained by the

ANN approach (i.e., $8.84 \cdot 10^{-4}$ mol L^{-1}), followed by the MINLP approach (i.e., $1.77 \cdot 10^{-1}$ mol L^{-1}) and the NLP approach (i.e., 1.83 mol L^{-1}). Table 2.2 in the manuscript and Table A.8 summarize these results, where it is emphasized that the MINLP and NLP approaches lead to compact model expressions revealing only 21 parameters each. The ANN architecture needs 83 parameters to fully describe the trained model.

Case Study III - Noisy dataset After training and integrating the models on noisy data, the lowest state space error in the test set is obtained by the ANN approach (i.e., $1.93 \cdot 10^{-1}$ mol L^{-1}), followed by the MINLP approach (i.e.,$2.55 \cdot 10^{-1}$ mol L^{-1}) and the NLP approach (i.e., $2.77 \cdot 10^{-1}$ mol L^{-1}). Table 2.3 in the manuscript and Table A.12 summarize these results, where it is emphasized that the MINLP approaches lead to compact model expressions revealing only 15 parameters. The NLP approach requires again 40 parameters, where the ANN architecture needs again 83 parameters to fully describe the trained model.

Although the ANN approach shows better overall performance, the difference is only marginal, as in CSII. Again, fewer parameters are required for the MINLP approach making such a model more suitable for later applications.

Case Study IV - Non-Noisy Dataset Underlying systems of bioprocesses are usually not described by the same kinetic power-law models used for chemical reactions. However, in this work, the same S-system structure is applied for predicting the concentration profiles of a bioprocess. In this case, the MINLP approach show better predictive performance than the ANN-based method.

After integration of the trained models, the lowest state-space error in the test set is obtained by the NLP approach (i.e., $3.72 \cdot 10^{-1}$ g L^{-1}), followed by the MINLP approach (i.e., 1.34 g L^{-1}) and the ANN approach (i.e., 3.80 g L^{-1}). Table 2.2 in the manuscript and Table A.8 summarizes these results, where it is emphasized that the MINLP/NLP approaches lead to compact model expressions revealing only 17 parameters each. The ANN architecture requires in total 33 parameters to fully describe the trained model.

Case Study IV - Noisy Dataset After training and integrating the S-system with the MINLP formulation in the same manner as before, the lowest state-space error in the test set is obtained by the ANN approach (i.e., $9.46 \cdot 10^{-2}$ g L^{-1}), followed by the MINLP approach (i.e., $1.62 \cdot 10^{-1}$ g L^{-1}) and the NLP

approach (i.e., $7.58 \cdot 10^{-1}$ g L^{-1}). Table 2.3 in the manuscript and Table A.12 summarize these results, where it is emphasized that the MINLP/NLP approaches lead to compact model expressions revealing only 9 parameters each. The ANN architecture again requires in total 33 parameters to fully describe the trained model.

The best performing MINLP model candidate is obtained at $NP = 9$. For two given test batch runs, the corresponding concentration profile predictions of this model candidate and the ANN model are shown in Figure A.2. One can observe that the candidate model of the MINLP solution usually predicts a time lagging profile. However, the shapes of the different phases (growth of cells, stagnation, substrate consumption, etc.) are well predicted.

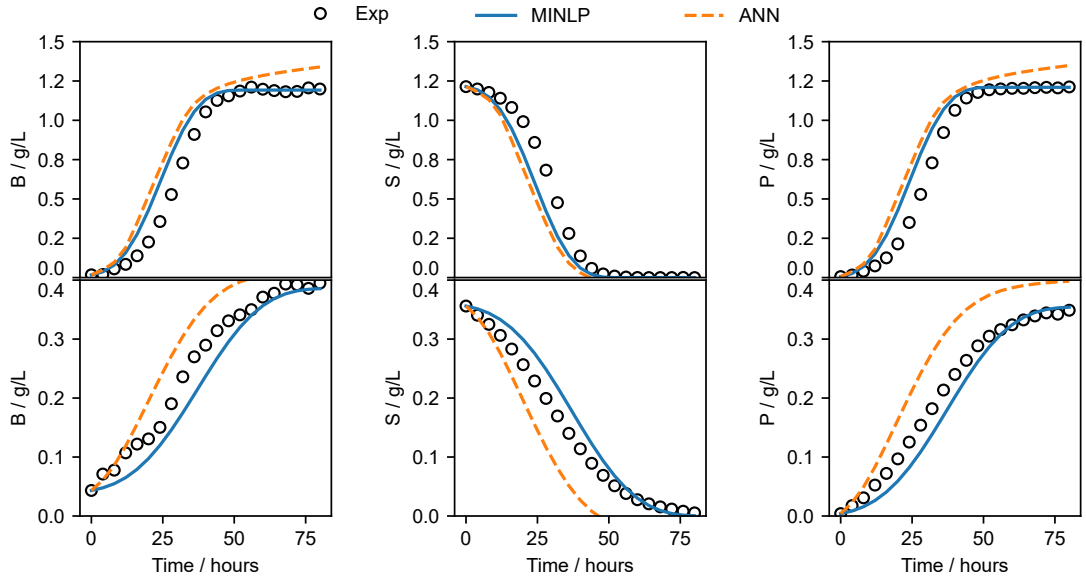

Figure A.2. The concentration profiles of the three species in CSIV (noisy data) are shown: The black circles represent the observed data, the dashed orange line shows the ANN prediction, and the solid blue line displays the predictions by the S-system (at $NP = 9$). The subplots in the top row and bottom row represent two different individual test runs (unseen in model training).

Two critical factors can therefore be read from the obtained solution. First, one can correctly describe the species' behavior and the entire bioprocess. Thus, having such a formalism at hand, a modeler could extract information about the micro- and macroscopic processes after training the system, further improving process knowledge. Second, instead of using the full model resulting from the NLP formalism, one would not reach the same predictive performance and lower

model complexity as if the binary constraints are implemented.

A.8 Identified model parameters

In the following Tables A.18 to A.21, additional information about the identified model parameters are given for each individual method.

Table A.18. Model parameters identified by the MINLP approach at the indicated NP value (corresponding to the best identified model given in Tables 2.2 and 2.3). Non-noisy data were used for model identification.

Parameter and Species[a]		**CSI** $NP = 17$				**CSII** $NP = 11$				**CSIII** $NP = 21$					**CSIV** $NP = 24$		
		A	**B**	**C**	**D**	**A**	**B**	**C**	**D**	**A**	**B**	**C**	**D**	**E**	**B**	**S**	**P**
α_i	-	1	1	5	2	0	10	1	0.500	0.000	0.150	0.200	0.790	0.330	0	55.630	6.400
β_i	-	5	5	3	0	8.430	1	0	0	0.610	0.290	0.000	1.450	0	0	55.660	6.340
$g_{i,j}$	A	0	0	1	0	0	0	0	0	0	0	0	0	0	1.050	0.860	0
	B	0	0	1	0	1	0	0	0	0	0	0	1	0	0.970	1.190	0
	C	1	0	0	0	0	1	0	0	0.050	0	-0.020	0.950	0	1.180	0.640	0
	D	0	0	1	0	2	0	0	0	0.570	0.330	0	0	0.020	-	-	-
	E	-	-	-	-	-	-	-	-	1	0	0	0	0	-	-	-
$f_{i,j}$	A	1	0	0	0	1.480	0	0.010	0	1	0	0	0	0	0	0	0
	B	0	1	0	0	0	1	0	0	0	1	0	0	0	0.970	1.190	0
	C	0	0	1	0	0	0	0	0	0	0	0	0	0	1.180	0.640	0
	D	0	0	0	0	0	0	0	0	0.640	0.230	0.710	-0.650	0	-	-	-
	E	-	-	-	-	-	-	-	-	0	0	0	0	0	-	-	-

[a] Column: species i, row: species j

Table A.19. Model parameters identified by the NLP approach. Non-noisy data were used for model identification.

Parameter and Species[a]		CSI $NP = NP_{max} = 40$				CSII $NP = NP_{max} = 40$				CSIII $NP = NP_{max} = 60$					CSIV $NP = NP_{max} = 24$		
		A	**B**	**C**	**D**	**A**	**B**	**C**	**D**	**A**	**B**	**C**	**D**	**E**	**B**	**S**	**P**
α_i	-	1	1	5	2	0	10.57	1	0.50	0	0.15	0.73	1.29	0.33	0.75	80.05	0.69
β_i	-	5	5	3	0	8.18	1.19	0	0	0.61	0.29	0	1.93	0	0.74	80.10	0.69
$g_{i,j}$	A	0	0	1	0	0	0	0	0	0	0	0	0	0	0.97	1.19	0.00
	B	0	0	1	0	1.11	-0.10	0.01	0.00	0	0	0	0	0	1.17	0.64	0.03
	C	1	0	0	0	0	1	0	0	0.39	0.47	0.80	-0.73	0.00	0.97	1.18	0.00
	D	0	0	1	0	2	0	0	0	0.58	0.32	0.09	-0.08	0.01	-	-	-
	E	-	-	-	-	-	-	-	-	0	0	0	0	0	-	-	-
$f_{i,j}$	A	1	0	0	0	1.47	0.04	0.21	-0.21	1	0	0	0	0	0.97	1.19	0.00
	B	0	1	0	0	0.03	0.82	0.19	-0.01	0	1	0	0	0	1.17	0.64	0.03
	C	0	0	1	0	0	0	0	0	0	0	0	0	0	0.97	1.19	0.00
	D	0.00	0.00	0	0	0	0	0	0	0.63	0.26	0.55	-0.52	-0.01	-	-	-
	E	-	-	-	-	-	-	-	-	0.87	-2	2	-1.89	-1.39	-	-	-

[a] Column: species i, row: species j

Table A.20. Model parameters identified by the MINLP approach at the indicated NP value (corresponding to the best identified model given in Tables 2.2 and 2.3). Noisy data were used for model identification.

Parameter and Species[a]		CSI $NP = 12$				CSI $NP = 20$				CSIII $NP = 15$					CSIV $NP = 9$		
		A	**B**	**C**	**D**	**A**	**B**	**C**	**D**	**A**	**B**	**C**	**D**	**E**	**B**	**S**	**P**
α_i	-	0	0	3.640	1.910	0.047	3.140	1.060	1.680	0	0.160	0.200	0.670	0.340	0.090	0.000	0.100
β_i	-	3.710	3.670	2.830	0	10.080	0.240	0	0	0.630	0.300	0	0.340	0	0	0.100	0
$g_{i,j}$	A	0	0	0	0	3	0	0.120	0	0	0	0	0	0	0	0.700	0.730
	B	0	0	0	0	0.820	0.650	0.020	-0.020	0	0	0	0.940	0	0	0	0
	C	1.040	0	0	0	0	0.980	0	0	0	0	0	0.970	0	0	0.700	0.730
	D	0	0	1	0	1.040	0	0	0	0.480	0.440	0	0	0	-	-	-
	E	-	-	-	-	-	-	-	-	1	0	0	0	0	-	-	-
$f_{i,j}$	A	1.130	0	0	0	1.540	0.140	0.050	0	0.970	0	0	0	0	0	0	0
	B	0	1.270	0	0	0	1.650	0.130	-0.170	0	0.990	0	0	0	0	0.700	0.720
	C	0.260	0	0.520	0.350	0	0	0	0	0	0	0	0	0	0	0	0
	D	0	0	0	0	0	0	0	0	0	0	0	0.910	0	-	-	-
	E	-	-	-	-	-	-	-	-	0	0	0	0	0	-	-	-

[a] Column: species i, row: species j

Table A.21. Model parameters identified by the NLP approach. Noisy data were used for model identification.

Parameter and Species[a]		CSI $NP = NP_{max} = 40$				CSII $NP = NP_{max} = 40$				CSIII $NP = NP_{max} = 60$					CSIV $NP = NP_{max} = 24$		
		A	**B**	**C**	**D**	**A**	**B**	**C**	**D**	**A**	**B**	**C**	**D**	**E**	**B**	**S**	**P**
α_i	-	0	3.76	3.69	1.99	0	3.30	15	0.07	0	0.26	0.20	0.72	0.54	7.57	299.90	11.02
β_i	-	4	5.54	2.96	0	2.75	0.28	14.40	0	0.60	0.40	0	0.42	0.30	7.49	300.00	10.95
$g_{i,j}$	A	0	0	0	0	0	0	0	0	0	0	0	0	0	0.15	0.47	0.13
	B	0.03	0.95	-0.01	0.51	0.80	0.64	0.02	-0.02	0.04	0.43	0.01	0.40	0.21	0.11	0.45	0.14
	C	1.02	0.02	0	0	0.00	0.69	-0.02	0.05	-0.01	0.01	0	0.96	0	0.10	0.48	0.15
	D	0.01	0.01	0.95	0.04	1.45	1.31	0.04	-0.02	0.56	0.35	0.01	0	0.01	-	-	-
	E	-	-	-	-	-	-	-	-	0.84	-0.01	0	0.01	-0.03	-	-	-
$f_{i,j}$	A	0.98	0.20	0.01	-0.01	1.04	0.74	0.02	-0.01	1	0.00	0	0.00	0.04	0.15	0.47	0.13
	B	0.02	0.97	0.00	0.02	0.00	1.59	0.12	-0.16	-0.01	0.95	0.00	0.02	0.07	0.11	0.45	0.14
	C	0.25	0.03	0.49	0.37	-0.01	0.64	-0.02	0.06	0	0	0	0	0	0.10	0.48	0.15
	D	0	0	0	0	1.04	0.74	0.02	-0.01	0.27	-0.02	0.11	0.39	0.29	-	-	-
	E	-	-	-	-	-	-	-	-	0.21	-0.25	0.01	0.36	-0.36	-	-	-

[a] Column: species i, row: species j

A.9 Comparison to Gaussian process and SINDy algorithm

In addition to the MINLP and the ANN approach, a GP was trained to predict the slopes for CSI. The state space errors were assessed for the three different methods and shown in Table A.22.

Table A.22. Error metrics of the different approaches are shown for the first case study based on non-noisy and noisy data. The best-performing approach in terms of state-space error is indicated in bold text. The units for the $RMSE$ are $\mathrm{mol\,L^{-1}}$, where the R^2 is a unitless quantity. For the MINLP model, only the best-performing model candidate of the MINLP approach in terms of state-space test error is listed (as done in the main manuscript). Bold values indicate a the best performing model candidate for each type of dataset.

CS	Method	Non-noisy State-space test $RMSE/R^2$	Noisy State-space test $RMSE/R^2$
I	ANN	$2.2{\cdot}10^{-2}$ / 0.9993	$1.8{\cdot}10^{-1}$ / 0.9544
	MINLP	$3.3{\cdot}10^{-4}$ / 1.0000	$\mathbf{1.5{\cdot}10^{-1}}$ / **0.9689**
	GP	$\mathbf{3.7{\cdot}10^{-5}}$ / **1.0000**	$1.9{\cdot}10^{0}$ / <0

Specifically, the GP performs slightly better than the MINLP or the ANN in the non-noisy state space (bold number in the table above). However, when using noisy data, the GP results in larger $RMSE$ values than the ANN or the MINLP approaches.

Additionally, we applied the sparse identification of nonlinear dynamics (SINDy) algorithm (Brunton et al., 2016) as a reference to all of the case studies used in the manuscript (noisy and also non-noisy datasets). For this, the python implementation PySINDy (Silva et al., 2020) was used. We used a sensitivity analysis around the default values to find suitable values for alpha (regularization parameter) and threshold (minimum magnitude for a coefficient in the weight vector) in the sequentially thresholded least squares algorithm (STLSQ) settings. The combination with the best R^2 was chosen to be used. Table A.23 shows the results of this comparison.

As demonstrated for these specific case studies, the MINLP approach performs better for CSI and CSII in both cases for the non-noisy and the noisy datasets. For CSIII (more species than in CSI-II), SINDy performs better in the absence of noise. It also outperforms the MINLP method in CSIV.

Table A.23. Error metrics of the different approaches are shown for the case studies based on non-noisy and noisy data. The best-performing approach in terms of state-space error is indicated in bold text. The units for the $RMSE$ are $\mathrm{mol\,h^{-1}}$, where the R^2 is a unitless quantity. For the MINLP model, only the best-performing model candidate of the MINLP approach in terms of state-space test error is listed (as done in the main manuscript). The used parameter values for "threshold" and "alpha" are shown in the last two columns. Bold values indicate the best performing model candidate for each type of dataset.

CS	Method	Non-noisy State-space test $\boldsymbol{RMSE}$ and test $\boldsymbol{R^2}$	Noisy State-space test $\boldsymbol{RMSE}$ and test $\boldsymbol{R^2}$	Non-noisy SINDy threshold and alpha	Noisy SINDy threshold and alpha
I	SINDy	$8.1{\cdot}10^{-2}$ / 0.979	$2.7{\cdot}10^{-1}$ / 0.829	10^{-1} / $5{\cdot}10^{-2}$	10^{0} / $5{\cdot}10^{-1}$
	MINLP	$\mathbf{3.3{\cdot}10^{-4}}$ **/ 1.000**	$\mathbf{1.5{\cdot}10^{-1}}$ **/ 0.969**	–	–
II	SINDy	$3.8{\cdot}10^{-1}$ / 0.954	$5.2{\cdot}10^{-1}$ / 0.963	10^{-1} / 10^{1}	10^{-1} / 10^{-1}
	MINLP	$\mathbf{2.6{\cdot}10^{-1}}$ **/ 0.995**	$\mathbf{3.24{\cdot}10^{-1}}$ **/ 0.990**	–	–
III	SINDy	$\mathbf{1.1{\cdot}10^{-3}}$ **/ 1.000**	$\mathbf{2.4{\cdot}10^{0}}$ **/ -0.234**	10^{-3}/ 10^{-3}	10^{-1} / $5{\cdot}10^{-2}$
	MINLP	$1.8{\cdot}10^{-1}$ / 0.997	$2.55{\cdot}10^{-1}$ / 0.993	–	–
IV	SINDy	$\mathbf{4.4{\cdot}10^{-2}}$ **/ 0.985**	$\mathbf{7.1{\cdot}10^{-2}}$ **/ 0.961**	10^{-2} / 10^{-4}	10^{-2} / 10^{-4}
	MINLP	$0.37{\cdot}10^{0}$ / 1.000	$1.62{\cdot}10^{-1}$ / 0.835	–	–

Appendix B

Supplementary information of Chapter 3

This part of the appendix contains the supplementary material of the article given in Chapter 3. It is organized as follows. Sections B.1 and B.2 summarize an in-depth analysis of derivative approximation methods. Section B.3 summarizes details to the neural network architecture and the grid search performed to tune the hyperparameters. Section B.4 shows the parameters used to generate the data of the different case studies. Section B.5 shows additional results and CPU times for model training. Section B.6 summarizes the model equations and values of the parameters that were identified by the BMS. Section B.7 discusses the impact of a different noise distribution (uniform distribution). Section B.8 summarizes background information about the case studies introduced in Chapter 3.

B.1 Comparison of derivative estimation methods

In the following section, we discuss the impact of the noise level and the number of available data points in the time series of one state profile on the accuracy of the derivative calculation. For this analysis, four different approaches to derive noisy data are applied to the sinusoidal test function shown in Figure 3.2 of the main manuscript. These methods include a forward finite difference (FD) approach, a Savitzky-Golay filter followed by forward finite difference (SG-FD) approach, the polynomial (Polynomial) approach, and the symbolic approach (BMS). The exact procedures of the different approaches are discussed in Section 3.3.1 of the main

manuscript (symbolic and polynomial approaches) or below in Section B.2 (FD and SG-FD approaches). The different noise levels considered in this analysis were 1%, 2.5%, 5%, 10%, and 20%, where the noise of these levels were added to the ground truth data. Additionally, the simulated state profiles consisted of either 5, 10, 20, 50, or 100 data points. Since the true derivative of the underlying ground truth is known, the mean squared error (MSE) and coefficient of determination (R^2) were used as performance metrics to check how well the derivatives were estimated. Figure B.1 summarizes the results of the analysis. Each of the heat

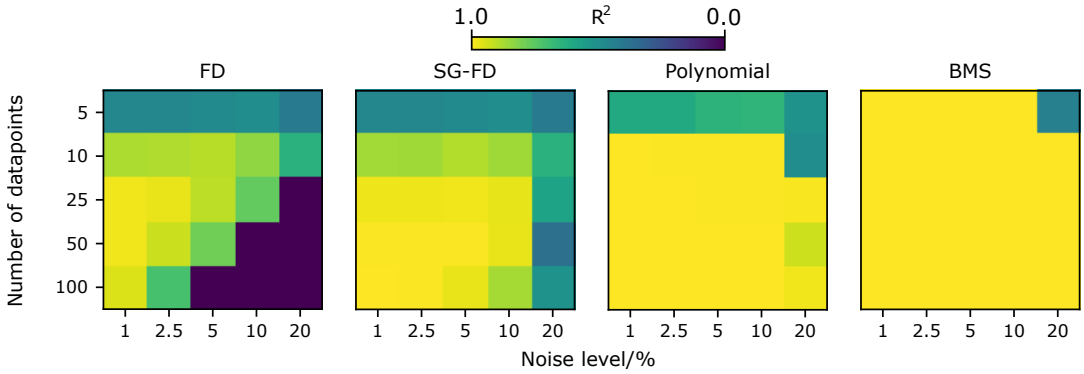

Figure B.1. Performance (coefficient of determination, R^2) of the different approaches to derive noise state profiles. Each subplot represents one method (FD: Forward finite difference, SG-FD: Savitzky-Golay filtered forward finite difference, Polynomial: Polynomial fitting and analytical derivation, BMS: Symoblic fitting and analytical derivation). On the x-axis of the heat maps, the different noise levels are shown, where the y-axis displays the number of data points per state profile. The brighter the color, the higher the performance of the approach (more accurate estimation of the derivative profile).

maps show the results for one method, where the above-mentioned noise levels and number of data points were considered. Starting with the FD approach, one can observe that by increasing the noise level or decreasing the number of data points results in a decreased accuracy of the numerically calculated derivatives. Applying a pretreatment step by including a Savitzky-Golay filter improves the accuracy of the derivatives in case of noise presence (SG-FD). However, the filtering might not be sufficient to estimate the derivatives well enough, even if 100 data points are available. This can be observed in the case of 10-20% noise in the SG-FD approach, where the coefficient of determination was obtained to be around 0.3 to 0.6. On the other hand, a higher accuracy could be achieved by the polynomial or BMS approach. Both methods lead to higher obtained R^2-values compared to the SG-FD approach. The BMS approach reaches very high accuracy, even in the case of a relatively low number of data points and a high noise level present in the measured

state space. In bioprocess applications, time series data of samples might be sparse (i.e., if processes are executed without having online measurements in place). Manually sampling from ongoing processes and prepare the samples for offline measurement takes time and effort. All of these aspects might lead to a relatively low sampling frequency. Considering this analysis summarized in Figure B.1, for cases with high noise levels or scarce data sets, the symbolic derivative estimation approach might therefore simplify the surrogate model identification process and boost the accuracy of the final models by removing noise more effectively.

B.2 Finite difference and Savitzky-Golay filter

In the following, the FD and SG-FD approaches for the derivative estimation are discussed in more detail. As mentioned in the manuscript, the methods are implemented and available in Python.

Forward finite difference
The forward finite difference (FD) approach is implemented by using the following approximation of the derivatives:

$$\dot{X}(t) = \frac{dX(t)}{dt} \approx \frac{\Delta X}{\Delta t} = \frac{X(t+1) - X(t)}{\Delta t} \tag{B.1}$$

Savitzky-Golay filtered forward finite difference
In the SG-FD approach, a Savitzky-Golay filter is used to smooth the state profile X (with time points $u \in U$) before deriving it. Details to the algorithm for the smoothing process can be found elsewhere (Savitzky & Golay, 1964). Here, we discuss how an appropriate combination of the window length w and polynomial order d of the SG filter are identified. A pseudo-code is given in below in algorithm 1. First, sets for the different polynomial orders $d \in D$ and window lengths $w \in W$ are defined together with a weight factor α. The objective J combines the current mean squared error (MSE) between the filtered values and the noisy state profile together with the total variation (VAR) as a weighted sum. For every order $d \in D$ and window length $w \in W$, an SG filter is applied and the objective J is calculated. For the best J identified (J^*), the corresponding smoothed data (X^*_{smooth}) is returned. After this filtering, the forward finite difference method described above is applied.

Algorithm 1: Procedure used to identify appropriate values for the polynomial order and window length of the Savitzky-Golay filter

Data: State profile X_u for every time step $u \in U$

Initialize *Polynomial order* $d \in D$*, window size* $w \in W$*, and weight factor* α
Initialize *Large values for* MSE^* *and* VAR^*
Initialize $J^* = \alpha MSE^* + (1-\alpha)VAR^*$
Initialize X *as* X^*_{smooth}
for *Every* $d \in D$ **do**
  for *Every* $w \in W$ **do**
    if $w > |U|$ *or* $d \geq w$ **then**
      Continue with next w
    end
    Apply SG filter with $d, w \rightarrow X_{smooth}$
    Calculate $MSE = \left[\sum_u^{|U|}(X_{smooth,u} - X_u)^2\right] / |U|$
    Calculate $VAR = \sum_u^{|U|-1}|X_{smooth,u+1} - X_{smooth,u}|$
    Calculate $J = \alpha MSE + (1-\alpha)VAR$
    if $J < J^*$ **then**
      Update J^* and X^*_{smooth}
    end
  end
end

B.3 Neural network architecture

Table B.1. Hyperparameters varied during the grid search for tuning the ANN parameters.

Parameter	**Values**
Hidden layer size	(3,), (10,), (100,), (3,3), (10,10), (100,100), (100, 10), (100, 50), [(i, j) for i in range(3, 10) for j in range(2, 7)]
Activation function	logistic, relu, tanh
Regularization term	10^{1}, 10^{0},10^{-1},10^{-2},10^{-3},10^{-4},10^{-5}
Initial learning rate	10^{-1}, 10^{-2}, 10^{-3}

Table B.2. Details to the fixed hyperparametes in the ANN architecture.

Parameter	**Values**
Maximum iterations	$15{\cdot}10^{3}$
Tolerance	10^{-4}
Random state	0
Number of iterations with no change	10
Maximum function evaluations	$15{\cdot}10^{3}$
Solver	Adam
Learning rate	Constant

B.4 Case studies

Table B.3. Parameters used in CSI.

Parameter	Value	Unit
ϕ_{max}	0.25	h^{-1}
K_S	105.4	$\mathrm{g\,L}^{-1}$
A_1	130	[–]
A_2	$3.8{\cdot}10^{48}$	[–]
E_1	12.4	$\mathrm{kJ\,mol}^{-1}$
E_2	298.6	$\mathrm{kJ\,mol}^{-1}$
K_ϕ	121.9	$\mathrm{g\,L}^{-1}$
$Y_{B,S}$	0.07	[–]
$Y_{P,S}$	0.167	[–]

Table B.4. Parameters used in CSII.

Parameter	Value	Unit
μ	0.109	h^{-1}
μ_d	0.0854	$\mathrm{L\,g}^{-1}\,\mathrm{h}^{-1}$
K_C	0	$\mathrm{mg\,L}^{-1}$
K_N	0.0086	$\mathrm{mg\,L}^{-1}$
K_P	0	$\mathrm{mg\,L}^{-1}$
Y_{C1}	217	$\mathrm{mg\,L}^{-1}$
Y_{C2}	0.839	$\mathrm{mg\,g}^{-1}\,\mathrm{h}^{-1}$
Y_{N1}	5.36	$\mathrm{mg\,L}^{-1}$
Y_{N2}	0.0559	$\mathrm{mg\,g}^{-1}\,\mathrm{h}^{-1}$
Y_{P1}	2.74	$\mathrm{mg\,L}^{-1}$
Y_{P2}	0.00833	$\mathrm{mg\,g}^{-1}\,\mathrm{h}^{-1}$

B.5 Additional error metrics and CPU times

Subsequently, additional results similar to those presented in Table 3.2 of the main manuscript are given. In Table B.5, the root mean squared error ($RMSE$) of the models are shown, where Table B.6 displays the mean absolute error (MAE). Tables B.7 to B.9 summarize the raw data for the calculation of the mean CPU

time displayed in Table 3.2 of the main manuscript.

Table B.5. The root mean squared error ($RMSE$) values are shown for the two case studies and their respective scenarios.

CS	Derivative method	BMS state RMSE	BMS derivative RMSE	ANN state RMSE	ANN derivative RMSE
I	Poly-40	3.444 / 0.493	0.050 / 0.055	7.069 / 2.281	0.064 / 0.064
	SR-40	2.559 / 5.940	0.101 / 0.077	3.428 / 1.840	0.093 / 0.083
	Poly-20	0.810 / 1.419	0.045 / 0.056	1.602 / 1.552	0.052 / 0.054
	SR-20	1.302 / 0.989	0.098 / 0.094	1.727 / 2.256	0.092 / 0.090
II	Poly-40	4.798 / 4.589	0.389 / 0.399	28.486 / 19.548	1.178 / 0.602
	SR-40	4.211 / 2.900	0.396 / 0.296	25.936 / 12.959	0.930 / 0.485
	Poly-20	5.436 / 5.257	0.586 / 0.593	28.186 / 31.662	1.692 / 1.598
	SR-20	2.067 / 2.312	0.232 / 0.366	17.841 / 12.404	0.780 / 0.703

Table B.6. The mean absolute error (MAE) values are shown for the two case studies and their respective scenarios.

CS	Derivative method	BMS state MAE	BMS derivative MAE	ANN state MAE	ANN derivative MAE
I	Poly-40	1.227 / 0.270	0.030 / 0.032	2.377 / 0.964	0.037 / 0.039
	SR-40	1.246 / 2.835	0.050 / 0.045	1.816 / 1.265	0.049 / 0.051
	Poly-20	0.446 / 0.742	0.028 / 0.035	0.720 / 0.803	0.032 / 0.035
	SR-20	0.759 / 0.905	0.054 / 0.051	0.883 / 1.133	0.049 / 0.051
II	Poly-40	2.123 / 2.083	0.159 / 0.163	12.861 / 8.876	0.410 / 0.264
	SR-40	1.332 / 1.261	0.103 / 0.098	11.122 / 5.406	0.267 / 0.160
	Poly-20	2.291 / 2.205	0.261 / 0.266	11.476 / 13.645	0.641 / 0.597
	SR-20	0.758 / 0.947	0.071 / 0.120	7.759 / 5.430	0.257 / 0.227

Table B.7. CPU times in seconds for the BMS training are displayed for the different species (B: Biomass, S: Substrate, P: Product (CSI) and Phosphate (CSII), C: Carbon).

	B	**S**	**P**	**B**	**N**	**C**	**P**	**Mean**
CSI-Poly-40	6754	9581	5477	-	-	-	-	7271
CSI-SR-40	2806	5228	4363	-	-	-	-	4132
CSI-Poly-20	5232	5768	6408	-	-	-	-	5803
CSI-SR-20	6405	9651	8444	-	-	-	-	8167
CSII-Poly-40	-	-	-	13921	895	20725	1995	9384
CSII-SR-40	-	-	-	14786	3373	13498	5443	9275
CSII-Poly-20	-	-	-	3797	499	3123	1060	2120
CSII-SR-20	-	-	-	9947	951	4602	2466	4492

Table B.8. CPU times in seconds for the ANN training are displayed for the different species (B: Biomass, S: Substrate, P: Product (CSI) and Phosphate (CSII), C: Carbon). The mean value column marked with * represents the mean of the training CPU times including the grid search times displayed in Table B.9.

	B	**S**	**P**	**B**	**N**	**C**	**P**	**Mean**	**Mean***
CSI-Poly-40	1	0	0	-	-	-	-	0	88
CSI-SR-40	0	0	0	-	-	-	-	0	88
CSI-Poly-20	1	0	1	-	-	-	-	1	59
CSI-SR-20	1	0	0	-	-	-	-	0	57
CSII-Poly-40	-	-	-	0	0	0	0	0	130
CSII-SR-40	-	-	-	0	0	1	0	0	162
CSII-Poly-20	-	-	-	0	0	0	0	0	155
CSII-SR-20	-	-	-	3	0	0	0	1	151

Table B.9. CPU times in seconds for the ANN grid search are displayed for the different species (B: Biomass, S: Substrate, P: Product (CSI) and Phosphate (CSII), C: Carbon).

	B	**S**	**P**	**B**	**N**	**C**	**P**
CSI-Poly-40	131	259	134	-	-	-	-
CSI-SR-40	128	263	135	-	-	-	-
CSI-Poly-20	88	175	89	-	-	-	-
CSI-SR-20	87	165	88	-	-	-	-
CSII-Poly-40	-	-	-	441	187	222	190
CSII-SR-40	-	-	-	607	216	253	218
CSII-Poly-20	-	-	-	775	144	174	146
CSII-SR-20	-	-	-	777	135	157	135

B.6 Identified BMS models

Table B.10. Most plausible rate equations identified by the BMS for CSI in plain text and Python format (i.e., ** representing 'to-the-power-of'). The corresponding values of the estimated parameters are given in Table B.11.

Scenario	**Rate of species**	**Identified expression in plain text**
Poly-40	dB/dt	(((x2*x1)**(a1+(a2/(a2+x3))))*a0)
	dS/dt	((a0+(-(((x3+(x1**(x2*a1)))*a1))*(x2**(((a2/x3)/(x2*a2))**a0))))+a2)
	dP/dt	-((((a2*((x1*x3)**a1))/(x2**a0))*((a2/x3)+(a0/(x2+(a1**x2)))))))
SR-40	dB/dt	(((x2/(((a2**2)+(a0**(x2+x3)))/x3))**a1)*a2)
	dS/dt	(a2*(((exp(((x3*a0)**2))/a1)+(x1+(x2*a2)))*x2))
	dP/dt	((x2/((x3*x2)+a1))*x3)
Poly-20	dB/dt	(((a1*x3)*x2)**a0)
	dS/dt	((((((x1*(x2**2))**a0)/(x3**a2))*a1)*a2)+a1)
	dP/dt	((((x2**a0)*a1)*(x1+x3))**a0)
SR-20	dB/dt	(((a2*x1)*x2)+a1)
	dS/dt	((((a0+x3)*((x1*x2)*x2))**a0)*a1)
	dP/dt	(((x3*x1)**a0)*((a2*x2)**a1))

Table B.11. Estimated parameter values to the models given in Table B.10. The parameters ignored by the BMS are highlighted with bright grey cells. These parametes are set to value 1 by the BMS and multiplied with the rest of the model. These parameters are therefore not shown in the equations given in Table B.10.

Scenario	Parameter	Rate equation		
		dB/dt	**dS/dt**	**dP/dt**
Poly-40	a_0	$4.164 \cdot 10^{-3}$	$5.343 \cdot 10^{-2}$	$-1.797 \cdot 10^{0}$
	a_1	$8.285 \cdot 10^{-1}$	$2.072 \cdot 10^{-2}$	$4.059 \cdot 10^{-1}$
	a_2	$-1.086 \cdot 10^{-1}$	$2.302 \cdot 10^{-2}$	$4.175 \cdot 10^{-3}$
SR-40	a_0	$1.307 \cdot 10^{0}$	$9.575 \cdot 10^{-2}$	$1.000 \cdot 10^{0}$
	a_1	$8.003 \cdot 10^{-1}$	$7.704 \cdot 10^{-1}$	$3.489 \cdot 10^{2}$
	a_2	$2.460 \cdot 10^{4}$	$-1.664 \cdot 10^{-2}$	$1.000 \cdot 10^{0}$
Poly-20	a_0	$7.882 \cdot 10^{-1}$	$3.609 \cdot 10^{-1}$	$8.955 \cdot 10^{-1}$
	a_1	$4.764 \cdot 10^{-4}$	$1.359 \cdot 10^{-1}$	$1.672 \cdot 10^{-3}$
	a_2	$1.000 \cdot 10^{0}$	$-4.423 \cdot 10^{-1}$	$1.000 \cdot 10^{0}$
SR-20	a_0	$1.000 \cdot 10^{0}$	$4.139 \cdot 10^{-1}$	$3.750 \cdot 10^{-1}$
	a_1	$1.612 \cdot 10^{-2}$	$-3.639 \cdot 10^{-2}$	$8.107 \cdot 10^{-1}$
	a_2	$1.389 \cdot 10^{-3}$	$1.000 \cdot 10^{0}$	$2.394 \cdot 10^{-3}$

Table B.12. Most plausible rate equations identified by the BMS for CSII in plain text and Python format (i.e., ** representing 'to-the-power-of'). The corresponding values of the estimated parameters are given in Table B.13.

Scenario	Rate of species	Identified expression in plain text
Poly-40	dB/dt	(((-(a5)/(a0+((a2**(a1/(a1+(a3+((x1*(a3+x2))+a3)))))/a2)))/a7
	dN/dt	(((a4*x3)*((x1+a0)/(a7+x2)))+a5)
	dC/dt	((((a1*(a7*(x1+a6)))+a7)+(x2*(a6**((a7+x3)*(x1**((x2*a1)+a0))))))*a2)
	dP/dt	(a1+((a0*x2)*((x4+a4)+x2)))
SR-40	dB/dt	(((x1/(x3**2))*(x2+(x3+a6)))*(a6+(a3/(a5+x1))))
	dN/dt	((((x3**a5)*(x4*((((((x2+a2)+(a1+x4))*(x2+a1))**2)/a3)+a4)))*(a1+(((x4**3)**(x4**a0))**2)))+a3)
	dC/dt	(((a3/x2)+(-((a7/x3))*((((x2/-(a7))*a6)*x2)+a2)))+(((x1+(a2+x2))*a2)**a0))
	dP/dt	((a5*((a3+((a4**x2)*(a1**x4)))**2))+((a7*(x4+a7))*a0))
Poly-20	dB/dt	(a2+(a1*(x1**a7)))
	dN/dt	a6
	dC/dt	((a3*(x1**a1))+a2)
	dP/dt	(a4+(a2*(a2/(x3**x4))))
SR-20	dB/dt	(a3*(((x1*a4)**((x2/x1)+((a7/x2)/x4)))+a5))
	dN/dt	a1
	dC/dt	((a4/((x3*(x1+a6))*a5))+a5)
	dP/dt	(((((x3+x2)/x4)*(a3+(((a6/x4)+(((((a5*x4)+x3)**2)/x2)+a2))*((a7**x2)*((x3*a4)*(x3**2)))))))/x4)/(x2**2))

Table B.13. Estimated parameter values to the models given in Table B.12. The parameters ignored by the BMS are highlighted with bright grey cells. These parametes are set to value 1 by the BMS and multiplied with the rest of the model. These parameters are therefore not shown in the equations given in Table B.12.

Scenario	Parameter	Rate equation		
		dB/dt	**dS/dt**	**dP/dt**
Poly-40	a_0	$-1.161 \cdot 10^{0}$	$-1.272 \cdot 10^{3}$	$6.855 \cdot 10^{-5}$
	a_1	$-6.563 \cdot 10^{23}$	$1.000 \cdot 10^{0}$	$7.913 \cdot 10^{-4}$
	a_2	$9.639 \cdot 10^{-1}$	$1.000 \cdot 10^{0}$	$5.722 \cdot 10^{0}$
	a_3	$5.184 \cdot 10^{20}$	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$
	a_4	$1.000 \cdot 10^{0}$	$-2.478 \cdot 10^{-7}$	$1.000 \cdot 10^{0}$
	a_5	$1.707 \cdot 10^{-1}$	$-7.113 \cdot 10^{-2}$	$-1.419 \cdot 10^{0}$
	a_6	$-5.692 \cdot 10^{-1}$	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$
	a_7	$2.732 \cdot 10^{-2}$	$-1.044 \cdot 10^{2}$	$-1.174 \cdot 10^{-1}$
SR-40	a_0	$1.000 \cdot 10^{0}$	$6.443 \cdot 10^{-1}$	$-1.430 \cdot 10^{-1}$
	a_1	$1.000 \cdot 10^{0}$	$-1.011 \cdot 10^{2}$	$1.000 \cdot 10^{0}$
	a_2	$1.000 \cdot 10^{0}$	$-6.896 \cdot 10^{1}$	$-1.408 \cdot 10^{3}$
	a_3	$1.836 \cdot 10^{6}$	$-7.080 \cdot 10^{-2}$	$-1.775 \cdot 10^{2}$
	a_4	$1.000 \cdot 10^{0}$	$4.289 \cdot 10^{6}$	$1.000 \cdot 10^{0}$
	a_5	$3.424 \cdot 10^{3}$	$-2.406 \cdot 10^{1}$	$1.000 \cdot 10^{0}$
	a_6	$-3.905 \cdot 10^{2}$	$1.000 \cdot 10^{0}$	$-2.688 \cdot 10^{-2}$
	a_7	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$	$3.121 \cdot 10^{-1}$
Poly-20	a_0	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$
	a_1	$-3.943 \cdot 10^{-15}$	$1.000 \cdot 10^{0}$	$9.732 \cdot 10^{0}$
	a_2	$3.143 \cdot 10^{1}$	$1.000 \cdot 10^{0}$	$-1.386 \cdot 10^{0}$
	a_3	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$	$1.850 \cdot 10^{-31}$
	a_4	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$
	a_5	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$
	a_6	$1.000 \cdot 10^{0}$	$-7.079 \cdot 10^{-2}$	$1.000 \cdot 10^{0}$
	a_7	$5.119 \cdot 10^{0}$	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$
SR-20	a_0	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$
	a_1	$1.000 \cdot 10^{0}$	$-7.038 \cdot 10^{-2}$	$1.000 \cdot 10^{0}$
	a_2	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$
	a_3	$-4.503 \cdot 10^{1}$	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$
	a_4	$7.940 \cdot 10^{-4}$	$1.000 \cdot 10^{0}$	$3.797 \cdot 10^{3}$
	a_5	$-1.041 \cdot 10^{0}$	$1.000 \cdot 10^{0}$	$-1.228 \cdot 10^{0}$
	a_6	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$	$-1.335 \cdot 10^{3}$
	a_7	$5.460 \cdot 10^{3}$	$1.000 \cdot 10^{0}$	$1.000 \cdot 10^{0}$

B.7 Impact of additive noise distribution

To compare the influence of the nature of the noise (i.e., its distribution), uniformly distributed noisy data was generated for the first case study given in Section 3.4.1. The general procedure to generate the data was kept the same as described in sections 3.4 and 3.4.1 (i.e., number of data points, number of batches for training and testing, etc.), except that for the noise generation, the random number generation was changed from the Python method *numpy.random.normal* (for a normal distribution) to *numpy.random.uniform* (for a uniform distribution). After the data generation, the same data pretreatment and training procedures are followed. This investigation was only carried out for case study 1 (Section 3.4.1), with 20 data points, and a symbolic regression differentiation for the derivative approximation. Therefore, this scenario with the uniform noise distribution can be compared to the scenario described by CSI-SR-20 in the main manuscript. As prediction models, the BMS and a hyperparameter-tuned ANN are used. For the training, all required steps were done as described in the manuscript, except that here, scaled inputs were used for the ANN to further improve the performance of the ANN, which was not required for the data that was displayed in the main manuscript.

Table B.14 summarizes the model training and test prediction results, where the model performances are also visualized in Figure B.2. To compare the impact of the uniform noise to the normally distributed noise, the results of CSI-SR-20 are included in Table B.14 as well, which originate from Table 3.2. Furthermore, Figure B.3 shows the time series profiles of the observed data (black circles) together with the BMS predictions (blue solid line), the ANN predictions (orange dashed line), and the true underlying derivatives (black dotted line) of an unseen test batch.

Table B.14. The coefficients of determination (R^2, unitless) are shown for the training and testing runs (notation: train/test) for the base case study 1 and the two different noise generation possibilities (normal or uniform distribution). The best-performing approach for each case in terms of state-space performance in the test set (unseen in training) is indicated in bold text.

	Noise Distribution	**BMS state R^2**	**BMS derivative R^2**	**ANN state R^2**	**ANN derivative R^2**
CSI-SR-20	normal	0.993 / **0.989**	0.981 / 0.986	0.988 / 0.984	0.983 / 0.988
CSI-SR-20	uniform	0.983 / **0.991**	0.964 / 0.981	0.979 / 0.984	0.966 / 0.971

It is visible that for both noise distributions, the suggested approach using the BMS, but also the ANN benchmark model, were able to capture the effects after an appropriate pretreatment of the data (i.e., the derivative approximation).

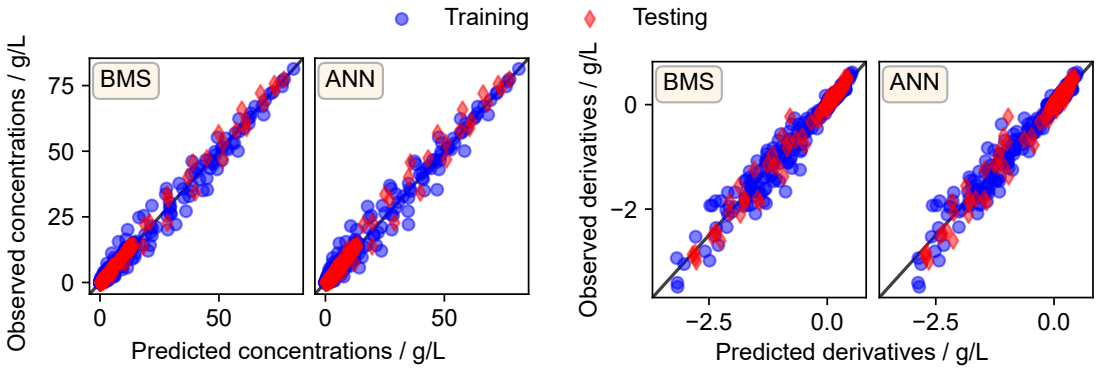

Figure B.2. The observed concentration values (left two figures) and derivative values (right two figures) are plotted against the model predictions for CSI where uniform noise was added to the data. Both, the ANN and BMS performances are displayed. Blue circles represent the training data, whereas red diamonds correspond to the test data. The black line represents the values where the observed value corresponds to the model prediction.

Considering the time series profile of a test batch prediction, the BMS and the ANN were both able to capture the dynamics well, as observable in Figure B.3. The regression is performed in the derivative space, where the data displayed in Figure S3 is in the state space. This means, both, the identified analytical expression by the BMS and the trained ANN could be well incorporated to estimate the right-hand-side of the ODE system from the values of the state variables, and an integration could be successfully performed, even for a test case which was not included in the training of the models.

From these calculations one can conclude that changing from a normally to a uniformly distributed additive noise, no significant change in model performance was observed, where predictions could still be made successfully.

B.8 Additional experimental information about the case study

For CSI, the considered ground truth model given the ODE system (3.16) in Chapter 3 is adapted from the example given by Turton et al. (2018). In this work, the batch production of L-phenylalanine via fermentation with a mutant *Brevibac-*

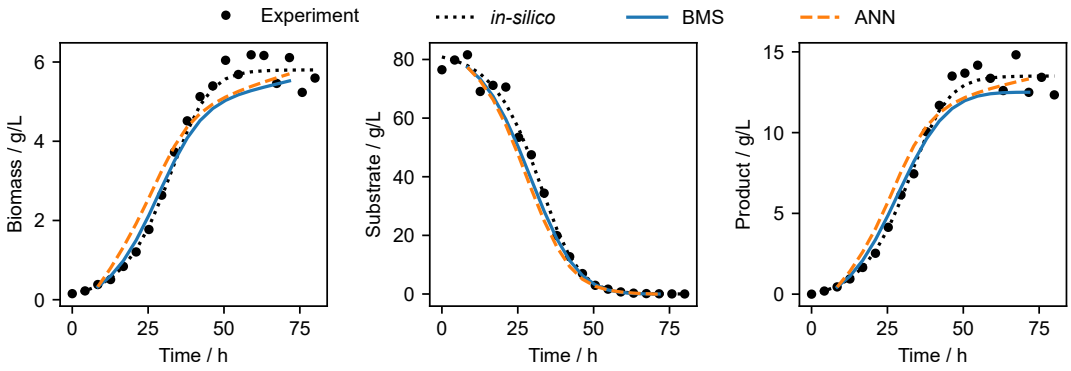

Figure B.3. The concentration profiles of the three species in CSI are shown together with the model predictions. The black circles represent the observed noisy data (uniform noise distribution) of an unseen test batch (not used for model training). The dashed orange line represents the ANN predictions, whereas the blue solid line represents the BMS predictions. It is worth mentioning the model predictions are only shown for the experimental time points that were used for model training, since some initial and last samples were removed from the training, as discussed in the manuscript.

terium lactofermentum 2256 (ATCC No. 13869) was studied. The authors used Monod kinetics to describe the dynamics of the system, which were adapted to create the system given in system (3.16) of Chapter 3. For CSII, the considered ground truth model given the ODE system (3.17) in Chapter 3 is based on the work of Del Rio Chanona et al. (2019). In their work, the researchers described these dynamics to be well suited for the bacterial system they studied. In the following, additional information on the experimental settings are provided, while the reader is also referred to the original work for further details (Del Rio Chanona, Wagner, et al., 2019), in which an algae, bacteria, and algae-bacteria-consortium wastewater treatment was studied:

The researchers conducted experiments using algae and bacteria. *Alga Chlorella vulgaris GY-H4* was sourced from the Institute of Hydrobiology, Chinese Academy of Sciences, China, where the bacterium Bacillus subtilis was obtained from previous research conducted at Xiamen University. Prior to the experiments in synthetic wastewater (SWW), the cells were precultured in BG-11 and Luria-Bertani media, respectively. Subsequently, they were inoculated separately in both high and low concentration SWW mediums.
The high concentration SWW initially contained the following ingredients (per liter of distilled water): 500 mg glucose, 1750 mg $NaHCO_3$, 727 mg $NaNO_3$, 83.3 mg KH_2PO_4, 7 mg NaCl, 4 mg $CaCl_2 \cdot 2\,H_2O$, 75 mg $MgSO_4 \cdot 7\,H_2O$, 2.5 mg $FeSO_4$,

20 mg EDTA, 0.001 25 mg $ZnSO_4$, 0.0025 mg $MnSO_4$, 0.0125 mg H_3BO_3, 0.0125 mg $Co(NO_3)_2$, 0.0125 mg Na_2MoO_4, and $6.25{\cdot}10^6$ mg $CuSO_4$. This resulted in 200 mg L^{-1} dissolved organic carbon (DOC), 120 mg L^{-1} total nitrogen (TN-$NO_3{}^-$), and 19 mg L^{-1} total phosphorus (TP-$PO_4{}^{3-}$). The low concentration SWW contained the following ingredients (per liter of distilled water): 100 mg glucose, 350 mg $NaHCO_3$, 115 mg $NaNO_3$, 13.2 mg KH_2PO_4, 7 mg NaCl, 4 mg $CaCl_2 \cdot 2\,H_2O$, 75 mg $MgSO_4 \cdot 7\,H_2O$, 2.5 mg $FeSO_4$, 20 mg EDTA, 0.001 25 mg $ZnSO_4$, 0.0025 mg $MnSO_4$, 0.0125 mg H_3BO_3, 0.0125 mg $Co(NO_3)_2$, 0.0125 mg Na_2MoO_4, and $6.25{\cdot}10^6$ mg $CuSO_4$. This resulted in 40 mg L^{-1} DOC, 19 mg L^{-1} TN-$NO_3{}^-$, and 3 mg L^{-1} TP-$PO_4{}^{3-}$.
Bacterial experiments were performed in a 500 mL baffled flask containing 100 mL of SWW medium, where a cultivation at 28 °C and 200 rpm was performed for 8 d, with an initial inoculum size of 0.24 g L^{-1}. Algal and algae-bacteria consortium experiments were conducted in a 1 L photobioreactor (PBR) equipped with external light sources. The light intensity and the aeration rate were set to 300 µmol m^{-2} s^{-1} and to 0.1 vvm with 2.5% CO_2, respectively. The initial culture volume was 800 mL of SWW medium, and the incubation was done for 8 days at 25 °C to 28 °C. The initial biomass concentration for the algal experiments was 0.24 g L^{-1}. In the consortium experiments, the same inoculum size of algae and bacteria was added to the PBR, with a combined concentration of 0.48 g L^{-1}. The consortium was also cultivated in sterilized SWW with high and low concentrations of glucose (500 and 100 mg L^{-1}), total nitrogen (TN-$NO_3{}^-$ 120 and 19 mg L^{-1}), and total phosphorus (TP-$PO_4{}^{3-}$ 19 and 3 mg L^{-1}), respectively. The culture pH was maintained between 7 and 8. Liquid samples were collected from the culture broth at specific time intervals to measure cell concentration, DOC, TP, and TN.

To obtain the concentration profiles, the biomass concentration was determined by measuring the optical density at 680 nm and expressed as dry weight (g L^{-1}). The biomass was collected by centrifugation at 5000 rpm for 5 min and then washed three times with water treated by reverse osmosis. During the experiments, carbon concentration was measured using a TOC analyzer (LiquiTOC II, Elementar, Germany) from filtered samples (0.45 µm). Nitrate ($NO_3{}^-$) and phosphate ($PO_4{}^{3-}$) ions in the filtered (0.20 µm) wastewater were analysed using an ion chromatograph (ICS-5000, Dionex, Italy).

Appendix C

Supplementary information of Chapter 4

This part of the appendix contains the supplementary material of the article given in Chapter 4. It is organized as follows. Section C.1 summarizes the CPU times needed to collect the initial samples. Section C.2 describes the case studies for the *in-silico* data generation. Section C.3 shows the a possible application of a linear basis function model (LBF). Section C.4 summarizes the sensitivity analysis for the solver settings (relative optimality gap) of MAiNGO.

C.1 Sampling times

An initial dataset was sampled for each CS, which was split into training and testing datasets. Table C.1 summarizes the time needed to perform this sampling. CSIII shows a significantly higher CPU time since the mentioned MINLP model for an optimum heat exchanger network (SYNHEAT, (Yee & Grossmann, 1990)) is solved iteratively.

C.2 Case study descriptions

CSI - Compressor plant

A constant feed rate of $1000\,\mathrm{kmol\,h^{-1}}$ of a vapor mixture is fed to the mixer. The composition of the feed was set to 78% nitrogen, 21% oxygen, and 1% of argon. The feed temperature was set to $25\,^{\circ}\mathrm{C}$ and $1\,\mathrm{bar}$. The Peng-Robinson

Table C.1. The recorded time needed to perform the initial sampling. Additionally, the number of samples is indicated.

CS	Number of samples	Sampling time [min]
I	200	85
II	1000	10
III	1000	320
IV	1000	40

equation of state was used for the Aspen HYSYS calculations. The compressor curves of the two compressors are given in Table C.2. Both compressors operate in centrifugal mode with a single-MW curve input option. The pressure ratio of both compressors is set to 1.5. The Schultz method is used as the polytropic method.

Table C.2. Compressor curves of the implemented compressors in CSI.

Compressor 1				Compressor 2			
Speed [rpm]	**Volume flow [actm^3/h]**	**Head [m]**	**Efficiency [%]**	**Speed [rpm]**	**Volume flow [actm^3/h]**	**Head [m]**	**Efficiency [%]**
5000	300	3100	74	3000	300	3100	68
	550	2950	78		550	2950	72
	850	2800	80		850	2800	74
	1200	2350	79		1200	2350	73
	1550	1550	68		1550	1550	62
	1750	900	51		1750	900	45
7000	1300	4600	74	10000	1300	4600	68
	1550	4500	78		1550	4500	72
	1850	4250	80		1850	4250	74
	2200	3750	79		2200	3750	73
	2550	2600	68		2550	2600	62
9000	1950	5400	74	15000	1950	5400	68
	2150	5250	78		2150	5250	72
	2500	5000	80		2500	5000	74
	2850	4300	79		2850	4300	73
	3150	3300	68		3150	3300	62
11000	2800	6800	74	25000	2800	6800	68
	3050	6650	78		3050	6650	72
	3350	6400	80		3350	6400	74
	3700	5800	79		3700	5800	73
	4150	4700	68		4150	4700	62

CSII - Ammonia reator

The reactor was modeled as a plug flow reactor in Aspen HYSYS. For the integration, 20 segments with a minimum step fraction of $1 \cdot 10^{-6}$ was chosen. The

catalyst was modeled with a diameter of $1 \cdot 10^{-3}$m, sphericity of 1, solid density of $2500\,\mathrm{kg\,m^{-3}}$, and a solid heat capacity of $250\,\mathrm{kJ\,kg^{-1}\,K}$. The reactor tube length was chosen to be 1.5 m with 0.2 m of diameter. One tube with a wall thickness of $5 \cdot 10^{-3}$m was selected, where the tube packing has a void fraction of 0.33. The feed entering the reactor consists of 100% vapor, where a constant molar flow rate of $2.242 \cdot 10^{-4}\mathrm{kg\,mol\,h^{-1}}$ was chosen.

CSIII - Methanol plant

This case study was adapted from Vázquez et al. (2021), where the process flowsheet is based on the works by VanDal and Bouallou (2013) and González-Garay et al. (2019).

As thermodynamic packages, the Peng-Robinson and NRTL-ideal were applied in Aspen HYSYS. We modeled the plant with a constant CO_2 feed of 2,000 kmol/h available at 25 °C and 1 bar. This stream is compressed to reach the reaction pressure (optimization variable). The desired final pressure is in the range of 45 bar to 55 bar. Hydrogen is fed at 30 bar and is compressed to reach the reaction pressure. After being compressed, the two gases are mixed with the recycled stream, and the resultant stream is heated to reach the desired reaction temperature. Two main reactions occur in the reactor. The first one is the CO hydrogenation to produce methanol ($R1$), which is accompanied by the water-gas shift reaction ($R2$), leading to the global reaction ($R3$), as shown below. The reaction equations are given in Table C.3.

Table C.3. Participating reactions in the methanol production plant.

Reaction name	Reaction equation
$R1$	$CO_2 + 2\,H_2 \rightleftharpoons CH_3OH$
$R2$	$CO_2 + H_2 \rightleftharpoons CO + H_2O$
$R3$	$CO_2 + 3\,H_2 \rightleftharpoons CH_3OH + H_2O$

The reactor was modeled as a plug flow reactor with a $Cu-ZnO-Al_2O_3$ catalyst. The kinetic model developed by Bussche and Froment (1996) was used, whereas alternative kinetic models could also have been used (Huš et al., 2017; Pavlišič et al., 2020; Slotboom et al., 2020). The reactor outlet is cooled to 35 °C and separated in a flash unit. A recycle stream with recompression is implemented to reach the reaction pressure. Part of the recycle stream, mainly containing CO_2, is purged. Before sending the bottom stream to a distillation column, the

stream is depressurized to 2 bar and sent to another flash unit. The distillation column reveals a partial condenser operating at a head pressure of 1 bar. The distillate product is methanol with a 99%, where the overhead, containing residual traces of CO_2 and methanol, is sent to the purge. The bottom product is wastewater.

As done in the work of Vázquez et al. (2021), the HEN design is optimized based on the stream data of the Aspen HYSYS model. High-pressure steam (40 bar, 250 °C) and cooling water (25 °C to 30 °C) are considered as utilities. The minimum temperature difference was set to 10 °C. The equations used to calculate the cost were based on the ones given by Sinnott and Towler (2020) and read as follows. The calculation was performed as shown in the work of Vázquez et al. (2021).

The Annual capital charge ratio ($ACCR$) is considered to be 20%. The capital expense ($CAPEX$) is obtained as follows:

$$CAPEX = ISBL + OSBL + CE + CC + CW$$

where $ISBL$ is the inside battery limit, $OSBL$ is the outside battery limit, CE is the contingency expense, CC is the construction cost, and CW is the working capital. The $ISBL$ is calculated as follows:

$$ISBL = \sum_{u \in U} UC_u$$

With U being the set of process units in the flowsheet, and UC_u is the cost of each unit, defined as follows:

$$UC_u = k(a + bC^n) \cdot \frac{CEPCI^{current}}{CEPCI^{2010}}, \quad \forall u \in U$$

Where the parameters k, a, b, and n depend on the process unit. The design variable C is used in the correlations that estimate the purchase cost of the unit. The Chemical Engineering Plant Cost Index ($CEPCI$) is then used to update the cost data. As done in the work of Vázquez et al. (2021), we consider a value

of 603.1 for the current one, corresponding to 2018, and a value of 532.9 for the 2010 one.

The other variables needed to assess the $CAPEX$ are calculated as percentages of the $ISBL$:

$$\begin{aligned} OSBL &= 0.35 \cdot ISBL \\ CE &= 0.20 \cdot (ISBL + OSBL) \\ CC &= 0.30 \cdot (ISBL + OSBL) \\ CW &= 0.15 \cdot (ISBL + OSBL) \end{aligned}$$

Where the corresponding factors are given by Sinnott and Towler (2020). The $OPEX$ is calculated as the addition of the fixed costs of operation (FOC) and the variable costs of operation (VOC).

$$OPEX = FOC + VOC$$

Where the FOC is calculated as follows:

$$FOC = C^{salary} + C^{superv} + C^{SalOv} + C^{Maint} + C^{Land} + C^{Ins} + C^{GenOv}$$

The costs included in the FOC are the salary, supervision, salary overhead, maintenance, land, taxes and insurance, and general overhead cost, respectively. The salary cost is obtained as follows:

$$C^{salary} = S \cdot NS \cdot OS$$

With S being the yearly salary of an operator, NS the number of shifts considered, and OS is the number of operators per shift. The other costs are obtained as percentages of the previous costs:

$$
\begin{aligned}
C^{superv} &= 0.25 \cdot C^{salary} \\
C^{SalOv} &= 0.45 \cdot (C^{salary} + C^{superv}) \\
C^{Maint} &= 0.03 \cdot ISBL \\
C^{Land} &= 0.01(ISBL + OSBL) \\
C^{Ins} &= 0.06 \cdot C^{work} \\
C^{GenOv} &= 0.64 \cdot (C^{salary} + C^{superv} + C^{SalOv} + C^{Maint})
\end{aligned}
$$

For the variable operation cost, we consider the following correlation:

$$VOC = C^{Feed} + C^{Util} + C^{Res}$$

With C^{Util} being the annual cost of the utilities, C^{Feed} the annual cost of the feeding, and C^{Res} the cost of treating the residues generated.

The equipment data used to calculate the UC is shown in Table C.4. For the fixed operation cost, the cost of the feeding material, and the utilities/residues, the parameters shown in Table C.5 to Table C.7 are used.

Table C.4. Cost data for the process units.

Unit	Design variable	k	a	b	n
Reactor	Volume [m^3]	4	61500	32500	0.8
Heat exchangers	Area [m^2]	1	28000	54	1.2
Process vessel	Mass [kg]	4	17400	79	0.85
Trays	Diameter [m]	1	130	440	1.8
Cooling tower	Water cooled [$L\,s^{-1}$]	2.5	170000	1500	0.9
Furnace	Duty [MW]	2.5	80000	10900	0.8
Compressor	Duty [kW]	2.5	580000	20000	0.6

Table C.5. FOC parameters.

Parameter	Value
S	40000 \$/operator
NS	3 shifts
OS	5 operator/shift

CSIV - Ammonia reactor series

The reactors were modeled as plug flow reactors in Aspen HYSYS. For the inte-

Table C.6. Parameters for C^{Feed}.

Parameter	Value	Reference
Cost of hydrogen from electrolysis	5.240 €$_{2015}$ kg^{-1}	Parkinson et al., 2019
Cost of hydrogen from SMR with CCS	0.165 €$_{2017}$ N^{-1}m^{-3}	Collodi et al., 2017
Cost of CO_2 from DAC	95.50 \$$_{2018}$ t^{-1}	Keith et al., 2018

Table C.7. Parameters for C^{util} and C^{Res}.

Parameter	Value	Reference
Cost of heating	12.1 \$$_{2018}$ GJ^{-1}	Wernet et al., 2016
Cost of cooling water	0.03 \$$_{2018}$ m^{-3}	Wernet et al., 2016
		Gonzalez-Garay and Guillen-Gosalbez, 2018
Cost of electricity	94.5 \$$_{2018}$ MWh^{-1}	Wernet et al., 2016
		Gonzalez-Garay and Guillen-Gosalbez, 2018
Cost of treating waste water	1.50 \$$_{2018}$ m^{-3}	Wernet et al., 2016
		Gonzalez-Garay and Guillen-Gosalbez, 2018

gration, 20 segments with a minimum step fraction of $1 \cdot 10^{-6}$ was chosen. The catalyst was modeled with a diameter of $1 \cdot 10^{-3}$ m, sphericity of 1, solid density of $2500\,\mathrm{kg\,m^{-3}}$, and a solid heat capacity of $250\,\mathrm{kJ\,kg^{-1}\,K}$. The reactor tube length was chosen to be 1.5 m with 0.2 m of diameter. One tube with a wall thickness of $5 \cdot 10^{-3}$ m was selected, where the tube packing has a void fraction of 0.33. The feed entering the plant (before the splitter) consists of 100% vapor, where a constant molar flow rate of $2.242 \cdot 10^4$ $\mathrm{kg\,mol\,h^{-1}}$ was chosen. The temperature and pressure of this stream were used as optimization variables since this stream directly enters the first reactor after the splitter. The composition was chosen to be 74% of hydrogen, 24.52% nitrogen, 1.46% ammonia, and 0.03% argon. The mixers between the reactors set the outlet pressure to the lowest inlet pressure. Additional compressors after mixing (before reactors 2 and 3) adjust the stream's pressure to the desired input pressure. Both compressors operate in centrifugal mode with a single-MW curve input option. The Schultz method is used as the polytropic method. Coolers (after the compressors) allow adjustment of the inlet temperatures for reactors 2 and 3. A pressure drop over the coolers was neglected.

C.3 Comparison of linear basis function model to the Bayesian machine scientist

This section summarizes the results of a sparse regression technique using a linear basis function (LBF) model, which is compared to the proposed approach. LBF also leads to closed-form and interpretable expressions, yet it performs worse in our case studies, as discussed below. This might occur because the BMS algorithm used in our framework searches in a much wider space of expressions since it does not require any pre-assumed structure of the final model. This is a clear advantage over the sparse regression methods, which assume a given model structure. However, the performance of both approaches will depend on the problem at hand. As shown below, in our case, the BMS provides better solutions in terms of both the accuracy of the surrogate and the quality of the final solution reported compared with a standard sparse regression technique. In principle, a larger search space in terms of plausible models should lead to better results, yet this might not always be the case, as the BMS does not guarantee the global optimality of the identified solution.

Specifically for CSIV, we implemented an LBF model using Python 3.8.11 and compared it with the BMS approach. We used the same input data x (i.e., $[b_1, b_2, T_1, T_2, T_3, P_1, P_2, P_3]$) and $f(x)$ (conversion) used for training the BMS and the GP shown in Chapter 4.

$$f(x) = w \cdot \phi(x)$$

Where $f(x)$ represents the target outputs of the process, w the weights/parameters of the model, and $\phi(x)$ describes the basis functions. The following nonlinear transformations (basis functions) were used:

$$\phi(x) = \left[x, \sqrt{x}, x^2, x^3, x_i x_j\right]$$

Since we considered eight input variables, the dimensions of x,$\sqrt{x}$, x^2, x^3 are $[n \times 8]$ each (for the n training samples). $x_i x_j$ represents the binary interactions (i.e., $b_1 b_2, b_1 T_1, \ldots P_2 P_3$), given by an array with dimensions $[n \times 28]$. The dimensions of $\phi(x)$, therefore, are $[n \times 60]$. Subsequently, we scaled the data using a standard

scaler provided by sklearn 1.0. We then applied a LASSO regression to perform feature selection. By varying the regularization parameter α and computing the R^2 value, we obtained the results in Figure C.1. Since at $\alpha = 10^{-3}$ the R^2 value is not significantly smaller than at $\alpha = 10^{-9}$, we chose this value for the regression. Subsequently, a prediction with the identified model results in the observed vs. predicted graph shown in Figure C.1. The model performs worse than the one identified with the BMS considering the same performance metric as in the manuscript (for the training: $MAE = 2.948\%$, $RMSE = 3.761\%$, $R^2 = 0.828$, and for the testing: $MAE = 3.291\%$, $RMSE = 4.207\%$, $R^2 = 0.757$). The parameter values are shown in Table C.8.

Table C.8. Regression parameters obtained by applying a LASSO approach with $\alpha = 10^{-3}$ to the above-described input data. Nonzero values are colored green, while zero cells are colored orange.

ϕ	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
w	-0.002	0	0	0	0	0	0	0	0	0
ϕ	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
w	0	0	0	0	0	0	-0.057	0	-0.685	-1.175
ϕ	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**
w	-2.193	-0.037	-0.014	0.037	-0.056	0	0.39	1.65	2.905	0
ϕ	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**
w	0.25	0	0	0.412	0.269	0	0.366	0	0	0
ϕ	**41**	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**
w	0	0.31	-0.049	0	0.18	0.284	0.069	-0.501	0.339	0.122
ϕ	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**	**60**
w	-0.001	0.061	-0.754	0	0	0	-0.476	0	0.385	0.111

The obtained algebraic function was then implemented in GAMS to perform the same optimization as with the BMS expression, finding a solution much better than the one identified with the LBF model, as shown in Table C.9. We know that the LBF model performance depends on the basis functions chosen and that a thorough cross-validation study would improve the results. However, a primary advantage of using the BMS, a model with no pre-defined structure, is that such as study is not needed.

We cannot claim that our approach will always be superior since this will be case-dependent. However, it can undoubtedly behave better than other existing approaches and is likely to perform even better in the future as more efficient

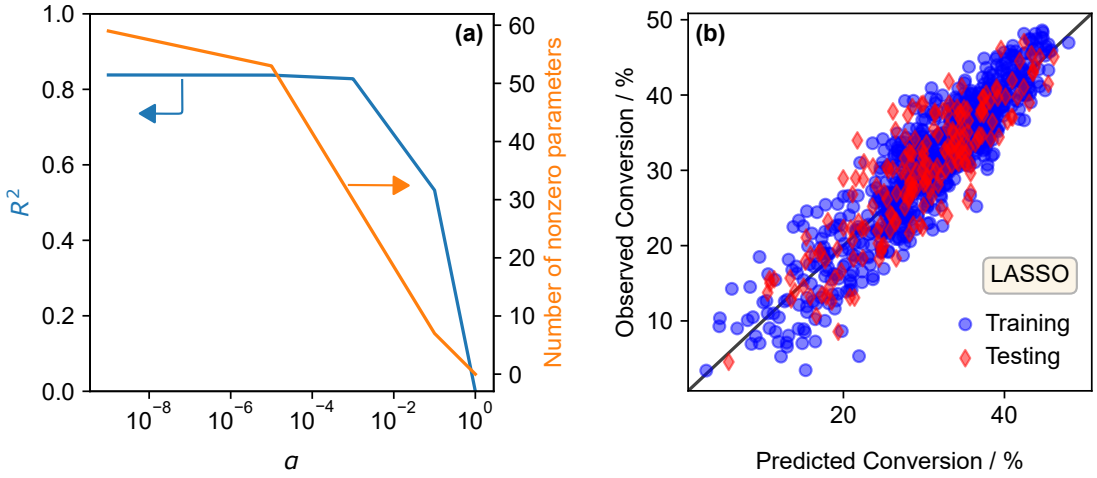

Figure C.1. (a) Obtained R^2 results as a function of the regularization parameter α (blue line). Additionally, the number of nonzero parameters w are shown as a function of α (orange line). (b) Observed versus predicted values for the training (blue circles) and test (red diamonds) set. The regression parameters w for the corresponding model are given in Table C.8.

symbolic regression algorithms become available.

C.4 Sensitivity analysis for the optimality gap of MAiNGO

We varied the optimality gap (ϵ_R) of MAiNGO for the values $\epsilon_R = [10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}]$. The maximum allowed CPU times were set to the corresponding training time the BMS required in the CSs: 16 500 s (CSI), 5600 s (CSII), 41 000 s (CSIII), and 123 500 s (CSIV). Multi-start was not applied. The results are summarized in Table C.10.

For CSI, the optimization led to the solution $x^*_{GP} = 0.61$, with an objective function value of $F(x^*_{GP})$ =413 kW for all values of ϵ_R. The maximum CPU time needed was 7 s, which was reached for $\epsilon_R = 10^{-4}$. Our proposed approach led to the same solution, as shown in the manuscript. Therefore, no difference was observed by varying the tolerance for the optimality gap.

For CSII, in all cases of ϵ_R, the corresponding optimality gap could be closed up to ϵ_R, leading to MAiNGO finding a global optimum. The solutions were identified after 3053 s/4315 s/4364 s/4476 s for the different ϵ_R. The same optimum solution $x^*_{GP} = [276\,°\text{C}, 198\,\text{bar}]$ was identified for all ϵ_R. The corresponding objective function value (conversion of nitrogen) was $F(x^*) = 37.8\%$. It is worth mentioning

Table C.9. Results for the linear basis function model (LBFM) for CSIV. In the first row, the optimization direction is given. The CPU time (in seconds) needed for model-based optimization is shown in the second row, followed by the model status obtained from BARON. The solution x^* found during the optimization is evaluated in Aspen HYSYS to obtain $f(x^*)$, which is then compared to the value of the surrogate $F(x^*)$ to determine the relative absolute error (which measures the mismatch between the surrogate and the original process model in the optimal solution found). The results for the proposed procedure shown in the manuscript (labeled with CSIV - BMS) are additionally shown for convenience.

	CSIV - BMS	**CSIV - LBFM**
Optimization direction	Maximization	Maximization
CPU Optimization	2 s	1 s
Model status	Globally optimal	Globally optimal
$F(x^*_{BMS})$	46.21 %	48.83 %
x^*_{BMS}	1 [−]	0.6 [−]
	0 [−]	0.3 [−]
	400 °C	351 °C
	382 °C	334 °C
	311 °C	365 °C
	195 bar	196 bar
	230 bar	199 bar
	230 bar	226 bar
$x^* \rightarrow$ HYSYS $\rightarrow f(x^*)$	47.03 %	39.79 %
RAE	2 %	28 %

that in the manuscript Table 4.8 we reported the solution for a pure multi-start, which is why MAiNGO returned a feasible solution, not a global optimum. With our proposed approach, the solution was obtained to be $x^* = [265\,°\text{C},\ 230\,\text{bar}]$, leading to $F(x^*_{BMS}) = 31.5\%$. However, inserting both solutions (x^*_{BMS} and x^*_{GP}) into HYSYS resulted in $f(x^*_{BMS}) = 29.3\%$ for BARON and $f(x^*_{GP}) = 27.5\%$ for MAiNGO. Therefore, considering this reported global optimum by MAiNGO, our approach led to a better solution.

For CSIII, MAiNGO reached the maximum allowed CPU time of 41 000 s already for $\epsilon_R = 10^{-1}$. Again, this was the same time the BMS needed for training. MAiNGO returned a feasible point (not a global solution) with an objective function value of $F(x^*_{GP}) = 0.714\,\$\,\text{kg}^{-1}$, where the corresponding optimum arguments were $x^*_{GP} = [233\,°\text{C},\ 5750\,\text{kmol}\,\text{h}^{-1},\ 0.001\,[-],\ 1.250\,[-],\ 53\,\text{m}^3,\ 5375\,\text{kPa}]$. If this solution was inserted into HYSYS $f(x^*_{GP})$ =0.729 $\$\,\text{kg}^{-1}$ could be obtained. As a comparison, the BARON approach led to the solution $F(x^*_{BMS})$ =0.716 $\$\,\text{kg}^{-1}$

with the optimum arguments $x^*_{BMS} = [209\,°\text{C},\ 5848\,\text{kmol}\,\text{h}^{-1},\ 0.001\,[-],1.526\,[-],\ 55\,\text{m}^3,\ 5497\,\text{kPa}]$. If these values were inserted into HYSYS, the solution $f(x^*_{BMS})$ =0.727 $\$\,\text{kg}^{-1}$ was obtained. Again, considering this reported global optimum by MAiNGO, our approach led to a better solution.

For CSIV, MAiNGO reached the maximum allowed CPU time of 123 500 s for $\epsilon_R = 10^{-2}$, where it returned a feasible point of $x^*_{GP} = [1\,[-],\ 0\,[-],\ 292\,°\text{C},\ 334\,°\text{C},\ 337\,°\text{C},\ 230\,\text{bar},\ 200\,\text{bar},\ 200\,\text{bar}]$. The optimal solution was found to be $F(x^*_{GP}) = 54.49\%$ conversion. If this solution was inserted into HYSYS $f(x^*_{GP}) = 50.04\%$ conversion could be obtained.

As a comparison, the BARON approach led to the solution $F(x^*_{BMS}) = 46.21\%$ with the optimum arguments $x^*_{BMS} = [1\,[-],\ 0\,[-],\ 400\,°\text{C},\ 382\,°\text{C},\ 311\,°\text{C},\ 195\,\text{bar},\ 230\,\text{bar},\ 230\,\text{bar}]$. If these values were inserted into HYSYS, the solution $f(x^*_{BMS}) = 47.03\%$ was obtained.

Table C.10. MAiNGO results for different values of the relative optimality gap ϵ_R. The maximum allowed CPU times were chosen according to the times the BMS needed for the training: 16 500 s (CSI), 5600 s (CSII), 41 000 s (CSIII), and 123 500 s (CSIV). Since these CPU times were reached for CSIII and CSIV already for $\epsilon_R = 10^{-1}$, results for the other values of ϵ_R were not reported for simplicity. The solver status indicates if MAiNGO reported a globally optimal solution (GO) or only a feasible point (FP). The relative absolute difference (RAE) indicates the deviation of the optimum objective function ($F(x^*_{GP})$) from the value found when the solution x^*_{GP} is inserted into HYSYS, described by $f(x^*_{GP})$.

CS	ϵ_R	**CPU [s]**	**Status**	**Final relative gap**	$F(x^*_{GP})$	x^*_{GP}	**Solver status**	$x^* \rightarrow$ **HYSYS** $\rightarrow f(x^*_{GP})$	***RAE***
I	10^{-1}	1	ϵ_R reached	$9.99 \cdot 10^{-2}$	413 kW	0.61 [–]	GO	413 kW	0 %
	10^{-2}	3	ϵ_R reached	$9.96 \cdot 10^{-3}$	413 kW	0.61 [–]	GO	413 kW	0 %
	10^{-3}	5	ϵ_R reached	$9.89 \cdot 10^{-4}$	413 kW	0.61 [–]	GO	413 kW	0 %
	10^{-4}	7	ϵ_R reached	$9.99 \cdot 10^{-5}$	413 kW	0.61 [–]	GO	413 kW	0 %
II	10^{-1}	3053	ϵ_R reached	$1.00 \cdot 10^{-1}$	37.81 %	276 °C, 198 bar	GO	27.45 %	38 %
	10^{-2}	4315	ϵ_R reached	$1.00 \cdot 10^{-2}$	37.81 %	276 °C, 198 bar	GO	27.45 %	38 %
	10^{-3}	4364	ϵ_R reached	$1.00 \cdot 10^{-3}$	37.81 %	276 °C, 198 bar	GO	27.45 %	38 %
	10^{-4}	4476	ϵ_R reached	$1.00 \cdot 10^{-4}$	37.81 %	276 °C, 198 bar	GO	27.45 %	38 %
III	10^{-1}	41000	CPU time reached	$2.93 \cdot 10^{2}$	0.714 $\$\,kg^{-1}$	233 °C, 5750 $kmol\,h^{-1}$, 0.01 [–], 1.250 [–], 53 m^3, 5375 kPa	FP	0.729 $\$\,kg^{-1}$	2 %
IV	10^{-2}	123500	CPU time reached	$1.56 \cdot 10^{2}$	54.49 %	1 [–], 0 [–], 292 °C, 334 °C, 337 °C, 230 bar, 200 bar, 200 bar	FP	50.04 %	8 %

Appendix D

Supplementary information of Chapter 5

This part of the appendix contains the supplementary material of the article given in Chapter 5. Section D.1 introduce the reader to the flexibility topic. Section D.2 shows the application of the proposed approach for two different designs in CSII.

D.1 Motivational examples

Example description Three motivational examples (ME) are introduced. The first linear example (ME-I) was inspired by the work of Pulsipher et al. (2019). There are no variables z present to solve the flexibility problem. The nonlinear example (ME-II) was adapted from Pulsipher et al. (2019), where again, no control variables z are present. The last example (ME-III) was used by Ochoa and Grossmann (2020), where control variables z are included to solve the flexibility problem. The set of system constraints f_j for the examples are shown in Table D.1, together with the settings of the optimizers. To solve these problems, Pyomo v6.4.4 was interfaced with the solvers CPLEX v41.3.0 (ME-I and ME-III) and BARON v22.7.23 (ME-II). On the one hand, the examples were used to cross-check the obtained results with a proven publication. On the other hand, they illustrate the usage of the flexibility index as a quantification metric to compare systems.

Table D.1. Constraints and properties used for the motivational examples.

	ME-I	**ME-II**	**ME-III**
Reference	Original from Pulsipher et al. (2019)	Adapted from Pulsipher et al. (2019)	Original from Ochoa and Grossmann (2020)
Type	Linear	Nonlinear	Linear
z	Not present	Not present	Variable
f_j	$f_1 : \theta_1 + \theta_2 - 14 \leq 0$ $f_2 : \theta_1 - 2\theta_2 - 2 \leq 0$ $f_3 : -\theta_1 \leq 0$ $f_4 : -\theta_2 \leq 0$	$f_1 : 10.5\theta_1 - 500 + \theta_2 \leq 0$ $f_2 : 0.8\theta_1 - 100 + \theta_2 \leq 0$ $f_3 : -2.4\theta_1 + 100 - \theta_2 \leq 0$ $f_4 : -(\theta_1 - 42)^2 + 37 - \theta_2 \leq 0$	$f_1 : z - \theta_1 + 2\theta_2 - 5 \leq 0$ $f_2 : -z - \frac{\theta_1}{3} - \frac{\theta_2}{2} - 3 \leq 0$ $f_3 : z + \theta_1 - \theta_2 - 6 \leq 0$ $f_{4,5} : \underline{z} \leq z \leq \overline{z}$ $f_{6,7} : \underline{\theta}_1 \leq \theta_1 \leq \underline{\theta}_1$ $f_{8,9} : \underline{\theta}_2 \leq \theta_2 \leq \underline{\theta}_2$
$\underline{\theta_k} \quad \forall k \in K$	[0; 0]	[0; 0]	[0; 0]
$\overline{\theta_k} \quad \forall k \in K$	[20; 20]	[50; 100]	[8; 5]
$\underline{z}$ and $\overline{z}$	-	-	[-8, 8]
Solver	CPLEX	BARON	CPLEX
Relative optimality gap	0	0	0

Results for motivational examples The results of the linear and nonlinear motivational examples ME-I and ME-II are summarized in Table D.2, where a schematic representation is given in Figure D.1. Comparing the solutions of δ^* in Table D.2, ME-I reveals a lower flexibility than ME-II ($\delta^* = 0.161$ vs. $\delta^* = 0.232$). For the two examples, maximum possible upper and lower deviations $\Delta\theta_k^max$ and $\Delta\theta_k^{min}$ were chosen that depend on the choice of the nominal operating point θ_k^N and the bounds ($\underline{\theta}_k$ and $\overline{\theta}_k$) of the considered uncertain parameter θ_k. This was done due to two reasons. First, the numerical ranges of the uncertain parameters can be taken into consideration, which makes the comparison of the value of δ^* more intuitive. Second, in a real world example, one usually does not have information about the allowed deviation $\Delta\theta_k^max$ or $\Delta\theta_k^min$. However, on the other hand, the bounds of the uncertain parameters are more likely to be known, where the operating point θ^N can be chosen by the modeler or process owner.
Problem (5.9) searches for the shortest distance from the nominal operating point to the next possible constraint. Therefore, the set of parameters located on the constraint closest to the nominal operating point is described by the critical uncertain parameters θ^c. This concept can be schematically shown by considering the projection of the constraints f_j onto the uncertain parameter plane (Figure D.1). In the above shown examples, the constraints are fixed and not influenced by any

Table D.2. Results summary of the motivational examples ME-I and ME-II given in Table D.1. A graphical representation of the examples is given in Figure D.1.

	ME-I	**ME-II**
Type	Linear without control	Nonlinear without control
θ^N	[4.0, 5.0]	[37.0, 43.0]
$\Delta\theta_k^{min}, \quad k \in K$	$\theta_k^N - \underline{\theta}_k$	
$\Delta\theta_k^{max}, \quad k \in K$	$\overline{\theta}_k - \theta_k^N$	
δ^*	0.161	0.232
θ^*	[6.6, 7.4]	[40.0, 33.0]
Active constraint	f_1	f_4
CPU	0.4 s	0.9 s

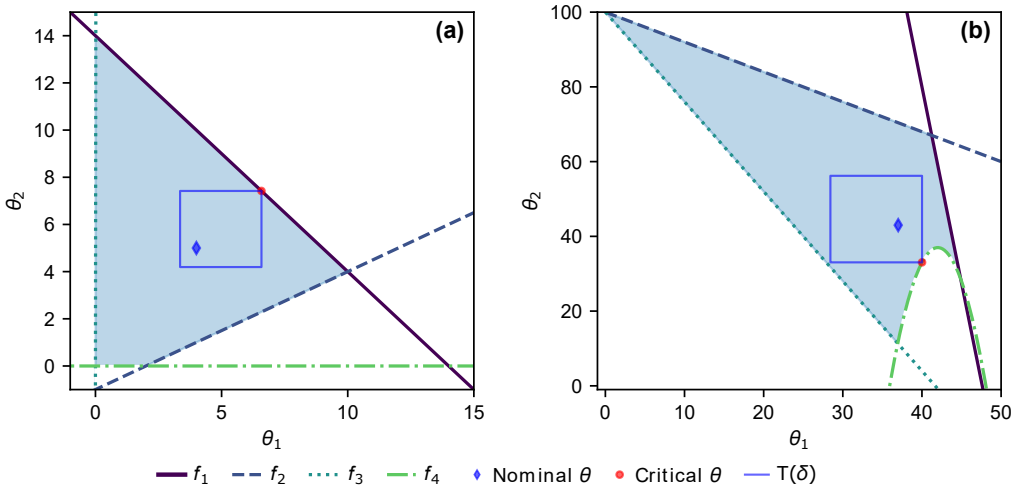

Figure D.1. Graphical representation of the motivational examples ME-I (a) and ME-II (b). The linear and nonlinear constraints f_j (dashed and solid lines) include the feasible region, which is shown by the bright blue shaded area. The chosen nominal operating point θ^N (blue diamond) is encountered by the set $T(\delta)$ (blue box). The box touches one constraint at the critical point δ^c (red circle).

control variable z. If we consider constraints that are depending on such a control variable z, as given in ME-III (Table D.1), the size of the feasible region changes upon varying the value for z. This is indicated schematically in Figure D.2. By solving the flexibility problem for such a system with control variables results in searching for the closest constraint or bound that is either not influenced by a control variable (and therefore can not be moved), or by reaching a constraint that is controlled but the control variable has reached a lower or upper limit. Figure D.3 shows a solution for ME-III. In this example, the closest constraint independent of the control variable is the upper bound of θ_2. Therefore, there are several solutions for the critical θ^c, since $T(\delta)$ touches a line (represented by the red circles in Figure D.3). It is worth noticing that constraint f_3 was moved

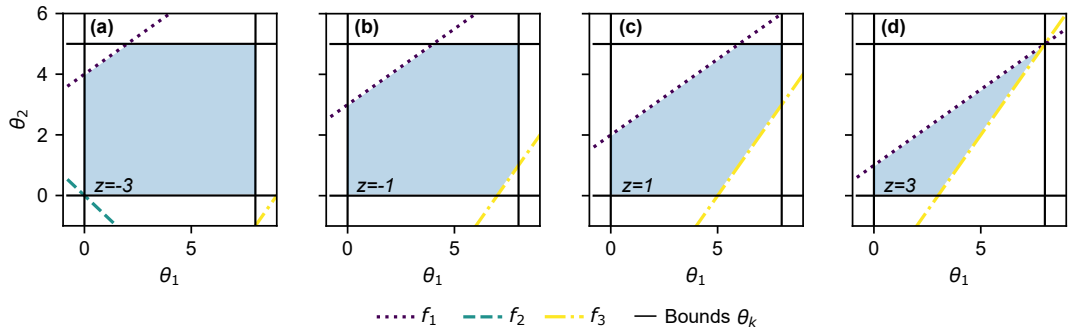

Figure D.2. Projection of the constraints onto the $\theta_1 - \theta_2$-plane. The constraints represent the system ME-III given in Table D.1. Increasing the value of $z = -3$ (a) up to $z = 3$ (d), the size of the feasible region is reduced. It is worth mentioning that constraint f_2 moved to negative θ_2-values in (b), (c), and (d), and is therefore not visible anymore due to the limits of the plots.

as far downwards (lower y-axis-intersections) as needed until a corner of $T(\delta)$ hit another bound (same in Figure D.2 (b), (c), and (d)).

D.2 Comparison of the flexibility in CSII

In addition to the design of the chromatographic column introduced in Section 5.4 (subsequently described by the design d_1), we adapted the column length and diameter to create another design d_2. The differences between the two designs are indicated in Table D.3. Everything else (i.e., upper, and lower bounds for the sampling, process parameters, etc.) is the same for both designs.

Table D.3. Design parameters for the two designs of case study II.

Property	**Design d_1**	**Design d_2**
Column length L_{col}	10 cm	15 cm
Cross-sectional area of the column A_{col}	0.2 cm^2	0.1 cm^2

We followed the same procedure for the sampling, model building, and incorporation of the surrogate model into the flexibility index formulation for the design d_2, as given in Chapter 5.4. The results for the model building for the two designs are compared in Table D.4. The identified closed-form expressions for the surrogate models and their estimated parameters are given in Table D.5 and Table D.6.

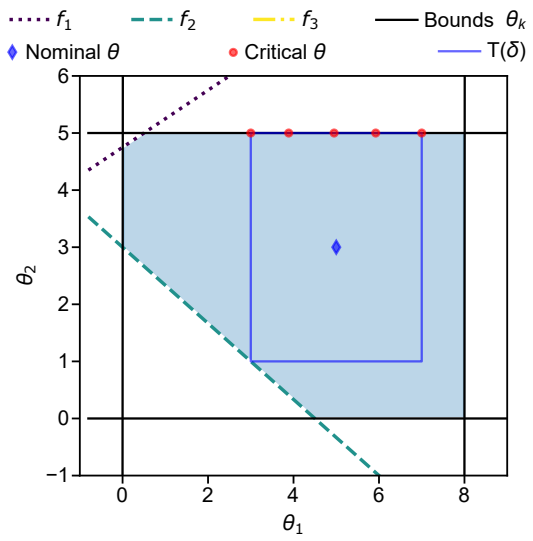

Figure D.3. Graphical representation of the solution for the flexibility index problem for ME-III. The constraints f_j (dashed and solid lines) include the feasible region, which is shown by the bright blue shaded area. The chosen nominal operating point θ^N (blue diamond) is encountered by the set $T(\delta)$ (blue box). The upper bound of θ_2 (independent of the control variable) is the constraint closest to the nominal operating point. $T(\delta)$ therefore touches a line (red circles).

Table D.4. The training performance criteria are summarized for the Bayesian machine scientist (BMS). Each row represents one design of the case study II. The CPU time (in hours) needed for the model training is shown in the left part of the table. The error metrics (root mean squared error, mean absolute error, coefficient of determination) are shown for the training and testing data (format: training/testing). The error units are given in squared brackets. The identified algebraic expressions are indicated in Table D.5, whereas the corresponding model parameters are reported in Table D.6.

CS	CPU training	RMSE	MAE	R^2
Design d_1	2.7 h	0.014 / 0.012 [–]	0.009 / 0.008 [–]	0.998 / 0.998 [–]
Design d_2	2.4 h	0.016 / 0.016 [–]	0.010 / 0.011 [–]	0.997 / 0.997 [–]

Table D.5. The most plausible closed-form expressions for each design in case study II identified by the Bayesian machine scientist (BMS) are shown. The corresponding estimated parameter values are reported in Table D.6.

CS	Prediction target	Identified expression
d_1	$LR = LR(c_{in}, Q, t_{load})$ $z = [Q]$ $\theta = [c_{in}, t_{load}]$	$a_0 \left(\frac{t_{load}}{\left(a_0^{\frac{t_{load}/c_{in}}{a_1}}\right) t_{load} + \frac{a_1}{\exp(c_{in}^{a_0})}} \right)^{-\frac{a_2 (a_0 Q)^{a_2}}{a_0} \left(\frac{Q}{c_{in}} + a_1\right)}$
d_2	$LR = LR(c_{in}, Q, t_{load})$ $z = [Q]$ $\theta = [c_{in}, t_{load}]$	$a_1^{\frac{\sqrt{\frac{Q a_2}{t_{load}}}}{c_{in} Q \left(t_{load}^2 + a_0\right)}} + a_1$

Table D.6. Parameter values of the most plausible surrogate model identified by the Bayesian machine scientist (BMS) for each case study (CS). The corresponding model equations are given in Table D.5.

Parameter	Design	
	d_1	d_2
a_0	0.894	524.029
a_1	24.458	0.02
a_2	-1.844	$7.325 \cdot 10^7$

The most significant difference between the two designs can be found in the obtained flexibility index. A visualization of the result is given in Figure D.4. As visible, the flexibility of the design d_1 given in (a) of Figure D.4 ($\delta^* = 0.811$) is significantly larger than the one obtained for the design d_2 ($\delta^* = 0.389$), shown in Figure D.4 (b). Therefore, considering the flexibility of the two columns, one can conclude that one should choose the design d_1 (shorter column with a larger cross-sectional area) to reach a more flexible process with respect to the studied uncertain parameters.

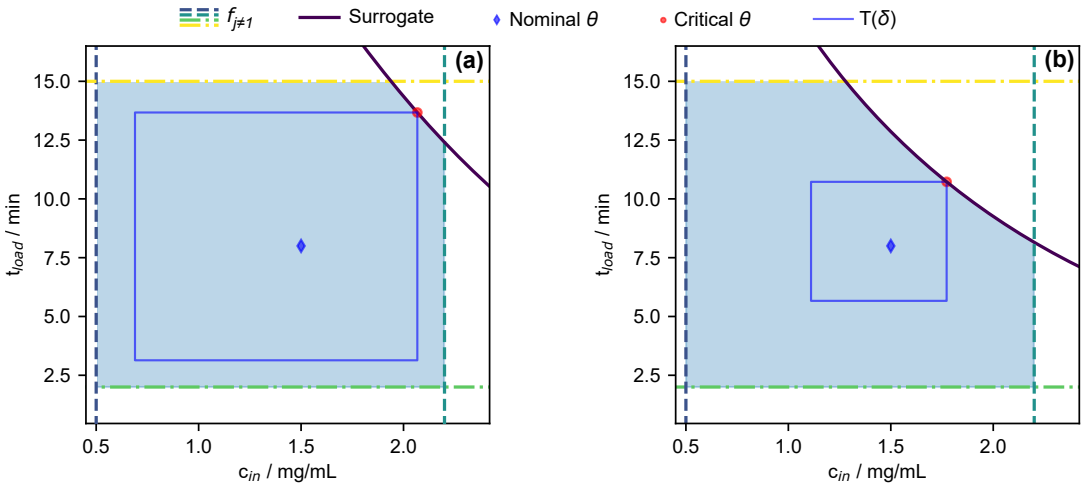

Figure D.4. Graphical representation of the solution for the flexibility index problem for the design d_1 (a) and the design d_2 (b) of case study II. The feasible region is shown in shaded light blue color. The constraints in dashed lines represent the bounds of the uncertain parameters. The solid lines represent the surrogate constraint which can be influenced by the control variable z. The chosen nominal operating point θ^N (blue diamond) is encountered by the set $T(\delta)$ (blue box). The surrogate constraints F_1 are active constraints (red circles).

List of contributions

List of publications and conference proceedings

T. Forster, D. Vázquez, M.N. Cruz-Bournazou, A. Butté, G. Guillén-Gosálbez. Modeling of bioprocesses via MINLP-based symbolic regression of S-system formalisms. *Computers & Chemical Engineering*, **2023**, 170, 108108.

T. Forster, D. Vázquez, G. Guillén-Gosálbez. Algebraic surrogate-based process optimization using Bayesian symbolic learning. *AIChE Journal*, **2023**, e18110.

T. Forster, D. Vázquez, G. Guillén-Gosálbez. Global optimization of symbolic surrogate process models based on Bayesian learning. *Computer Aided Chemical Engineering*, **2023**, 52, 1241-1246.

T. Forster, D. Vázquez, I. Fons Moreno-Palancas, G. Guillén-Gosálbez. Algebraic surrogate-based flexibility analysis of process units with complicating process constraints. *Computers & Chemical Engineering*, **2024**, 184, 108630.

T. Forster, D. Vázquez, I. Fons Moreno-Palancas, G. Guillén-Gosálbez. Flexibility analysis using surrogate models generated via symbolic regression. *Computer Aided Chemical Engineering*, **2024**, 53, 2791-2796.

T. Forster, D. Vazquez, C. Müller, G. Guillén-Gosálbez. Machine learning uncovers analytical kinetic models of bioprocesses. *Chemical Engineering Science*, **2024**, 300, 120606.

List of conference presentations

Hybrid Modeling of Bioprocesses Based on a Kinetic Canonical Formalism. 2022 American Institute of Chemical Engineers (AIChE) Annual Meeting, Phoenix (AZ), USA. November 2022, 2022.

Global optimization of symbolic surrogate process models based on Bayesian learning. 33rd European Symposium on Computer Aided Process Engineering (ESCAPE), Athens, Greece. June, 2023.

Flexibility Analysis Using Surrogate Models Generated via Symbolic Regression. 34th European Symposium on Computer Aided Process Engineering (ESCAPE), Florence, Italy. June, 2024.