Economic Optimizing Control for a U-loop Reactor Modelling, Estimation, and Economic Nonlinear Model Predictive Control

Nielsen, Marcus Krogh

Publication date:
2023

Document Version
Publisher's PDF, also known as Version of record

Ph.D. Thesis
Doctor of Philosophy

DTU Compute
Department of Applied Mathematics and Computer Science

Economic Optimizing Control for a U-loop Reactor

Modelling, Estimation, and Economic Nonlinear Model Predictive Control

Marcus Krogh Nielsen (mkrni)

Kongens Lyngby 2023

DTU Compute
Department of Applied Mathematics and Computer Science
Technical University of Denmark

Matematiktorvet
Building 303B
2800 Kongens Lyngby, Denmark
Phone +45 4525 3031
compute@compute.dtu.dk
www.compute.dtu.dk

Abstract

In this thesis, we present models and algorithms for monitoring and economic model predictive control applied to a fermentation process for single-cell protein (SCP) production in a U-loop bioreactor. We describe a modelling framework for compactly describing stoichiometry and kinetics in chemical and biochemical systems. In a technical report, we outline and apply the modelling framework for batch (BR), fedbatch (FBR), continuous stirred tank (CSTR), plug flow reactors (PFR), as well as their combinations, and present numerical examples of chemical and biochemical systems. We apply the modelling framework to existing and novel models describing growth of the bacteria *Methylococcus capsulatus* (Bath) for single-cell protein (SCP) production. Based on the modelling framework, we present a continuous U-loop fermentor model combining CSTR and PFR compartments. We describe model-based monitoring and control techniques for continuous-discrete nonlinear systems involving stochastic differential equations (SDEs). In a FOCAPO/CPC 2023 paper, we present the extended Kalman filter (CD-EKF), unscented Kalman filter (CD-UKF), ensemble Kalman filter (CD-EnKF), and a particle filter (CD-PF), test the estimations algorithms on a nonlinear test system, and discuss their performance. In papers presented as CCTA 2019 and CDC 2020, we apply the CD-EKF, present economic optimal control for optimal production economy, and perform a numerical experiment of economic nonlinear model predictive control (ENMPC) for SCP production in a U-loop bioreactor based on an SDE model. To include measurement information from commonly available online sensors, e.g. dissolved oxygen, temperature, and pH-value, we present monitoring and control techniques for continuous-discrete systems involving stochastic differential algebraic equations (SDAEs). In a ECC 2023 paper, we present an SDAE model describing cell growth and chemical equilibria, i.e. pH dynamics, and economic optimal control for SCP production in a laboratory-scale CSTR. The laboratory-scale CSTR model is based on a physical laboratory-scale bioreactor and is intended for systems identification experiments. In chapter 8, we present the CD-EKF for SDAE models and apply it to the laboratory-scale CSTR in chapter 11. As such, we outline the main components necessary for the implementation of ENMPC for SCP production in a U-loop bioreactor based on available online sensors. Finally, we present a high-performance Monte Carlo simulation toolbox for uncertainty and performance quantification in closed-loop systems. In a paper presented at CDC 2021, we describe and apply the Monte Carlo toolbox in a numerical experiment of SCP production in a fedbatch reactor with open-loop, PID, and model predictive control strategies.

Summary

Modern society is faced with a number of struggles with global reach, e.g. climate change and biodiversity. This has sparked an increasing focus among governments, institutions, and consumers on sustainable consumption of resources, i.e. alternative products with lower or net-zero environmental impact. Despite efforts from developed countries, the global population and meat consumption have a strong positive correlation with little to no change over the past decades. This suggests that we will have to look for alternative protein sources, not only for humans, but also for livestock, i.e. animal feed in meat production. Currently, fishmeal and soy are the most popular protein supplement products for animal feed. However, fishmeal, as the name suggests, comes from animals itself, and therefore present obvious mass balance issues - as one might ask, ”Where do the fish get the protein from?”. Soy be considered a sustainable source of protein. However, it also presents a range of issues regarding sustainability. Firstly, soy is mainly grown in South America, where the arable land requirements for soy production cause issues with deforestation. Deforestation presents issues with regard to both of the global struggles presented above; climate change and biodiversity. Secondly, the protein content of soy is significantly lower than that of fishmeal, so larger quantities of soy-based protein products are required compared to fishmeal. This only adds to the deforestation issue. A number of alternative sources of protein have sparked interest in recent decades. Among them are considered insect protein, duckweed protein, and single-cell protein. The former sources have strengths of their own, e.g. growing on industrial waste material such as pulp waste. However, in this work we focus on the ladder, single-cell protein.

In this thesis, we present advanced tools for monitoring and economic nonlinear model predictive control for a nonlinear system describing single-cell protein production in a novel bioreactor with good mixing and mass transfer properties. Unibio A/S is a biotechnology company operating a patented U-loop bioreactor for single-cell protein production. In the bioreactor, a bacterial culture is growing on cheap hydrocarbons as carbon and energy sources, e.g. methanol or methane, to produce a protein rich biomass ($\sim$ 72%) which is serves as an alternative protein sources for animal feed supplementation. Operation of the U-loop reactor displays stiff and unstable dynamics, calling for advanced monitoring and process control systems to maintain stable and high productivity and ensure financial viability of the alternative protein product produced by Unibio; Uniprotein.

The subject of this thesis is economic optimising control, or economic model predictive control, for single-cell protein production in a U-loop bioreactor. Economic model predictive control describe a model-based control strategy consisting two main components; state estimation and economic optimal control. The state estimator provides state feedback based on system measurements and the economic optimal control problem provides open-loop control strategies which are optimal with respect to economic measures of performance. The first control action from the solution to the optimal control problem is then implemented in the process. The sequence of state estimation and optimal control is then repeated when a new measurement becomes available. We present growth models for single-cell protein production using the bacteria *Methylococcus capsulatus* (Bath). We present models based on methanol and methane as carbon sources and several different nitrogen sources; nitric acid, ammonia, ammonium, nitrite, and nitrate, and molecular nitrogen. Furthermore, we describe chemical equilibrium reactions. These are included in growth models, resulting in differential algebraic systems, allowing for the inclusion of measurements from common online sensors, e.g. pH-value, in the state estimator.

In the context of state estimation for monitoring and control, we present methods of state estimation in nonlinear systems involving stochastic differential equations and stochastic differential algebraic equations. We focus on methods based on Bayesian inference and do not consider optimisation-based estimators in this work. For systems involving stochastic differential equations, we formulate and implement four estimators; the extended Kalman filter, the unscented Kalman filter, the ensemble Kalman filter, and a particle filter. For systems involving differential algebraic equations, we formulate and implement the extended Kalman filter. The extended Kalman filter directly applies the Kalman filter update on a local linearisation of a nonlinear systems and is thus a direct extension of the Kalman filter to nonlinear systems. The performance of the extended Kalman filter is largely dictated by the error associated with the linearisation, i.e. the nonlinearity of the system. The unscented Kalman filter propagates a set of deterministically sampled particles, so-called sigma-points, through the nonlinear state and measurements dynamics. This results in better convergence properties than the extended Kalman filter, but this estimator can also have trouble converging for high-nonlinear systems. The ensemble Kalman filter is a type of particle filter which approximates the nonlinear state and measurement distribution with a set of randomly sampled particles. The ensemble Kalman filter is a type of particle filter. The ensemble Kalman filter still has Gaussian assumptions, but avoids degeneracy issues by applying the Kalman filter update to each particles in its ensemble when a measurement is available. The particle filter approximates the nonlinear state and measurement distributions with a set of randomly sampled particles. In this work we consider Gaussian measurement noise, but the particle filter can be applied for arbitrary measurement distributions as long as likelihood can be computed for each particle in the measurement distribution. The particle filter can suffer from degeneracy, where the particle set converges to a single point, causing the estimator to fail. Both the ensemble and particle filters suffer from coverage issues for high-dimensional systems, i.e. the curse of dimensionality.

We present economic nonlinear model predictive control systems for single-cell protein production. We present a conceptual description and direct formulation of the control technology and describe the economic optimal control problem. In the economic optimal control problem, we directly optimise economic measures of performance in the nonlinear systems, e.g. revenue generated from production, cost associated with input or power consumption, or economic measured of sustainability, such as carbon footprint. The model describing single-cell protein production in a U-loop bioreactor presents a stiff and unstable system. Therefore, we consider a direct simultaneous approach to the solution of the nonlinear programme to address the instability of the system. Furthermore, we apply an implicit Euler method for temporal discretisation, i.e. right rectangular rule, of integrals and system dynamics to address stiffness.

We present numerical experiments of single-cell protein production in U-loop bioreactor with a methanol-based growth model. We present an optimal open-loop strategy for optimal profit based on an economic optimal control problem. We apply the extended Kalman filter for state estimation in an open-loop numerical experiment. Finally, we apply an economic nonlinear model predictive control system in a closed-loop numerical experiment on the U-loop bioreactor. The numerical experiment show that the control system can stabilise the start-up of the reactor and maintain stable high productivity during continuous operation and production. Additionally, we present a laboratory-scale continuous stirred tank bioreactor for single-cell protein production. In the single-cell protein production model, we include chemical equilibrium reaction dynamics, resulting in a differential algebraic systems. We present modelling, simulation, economic optimal control, and state estimation for this system. We solve an economic optimal control problem including the algebraic equations. We apply the extended Kalman filter in an open-loop numerical experiment using the optimal control strategy computed as the solution to the economic optimal control problem. The state estimation converge to the true values of the state and algebraic variables and accurately estimate a modelled parameter based on online measurements of dissolved oxygen concentration and pH-value.

Finally, we present and implement a high-performance Monte Carlo simulation toolbox. The toolbox is implemented in C with openmp for parallel Monte Carlo simulations of closed-loop systems. In Monte Carlo simulation for closed-loop systems, all simulations are independent of all other simulation, i.e. embarrassingly parallel. As such, the problem presents close to linear scaling with the number of cores. We apply the toolbox for uncertainty quantification of four different control systems in a numerical experiment for a fed-batch bioreactor. We test an open-loop controller, two PID controllers, one hand-tuned and one with optimal tuning determined from Monte Carlo simulations, and a nonlinear model predictive controller. The toolbox executes 30,000 open-loop and PID closed-loop simulations in less than a second, and

1,000 nonlinear model predictive control closed-loop simulation in approximately 30 minutes on the fed-batch bioreactor. This is a speed-up of more than 2,300 times compared to a benchmark Matlab serial implementation.

Summary (Danish)

Det moderne samfund står over for en række udfordringer med global rækkevidde, såsom klimaforandringer og biodiversitet. Dette har ført til et stigende fokus blandt regeringer, institutioner og forbrugere på bæredygtigt forbrug af ressourcer, dvs. alternative produkter med lavere eller *net-zero* miljøpåvirkning. Trods bestræbelser fra udviklede lande har den globale befolkning og det globale kødforbrug en stærk positiv korrelation med en lille eller ingen ændring over de seneste årtier. Dette antyder, at vi bliver nødt til at lede efter alternative proteinkilder, ikke kun for mennesker, men også for husdyr, dvs. dyrefoder i kødproduktionen.

I øjeblikket er fiskemel og sojabønneprotein de mest populære protein-tilskudsprodukter til dyrefoder. Imidlertid kommer fiskemel, som navnet antyder, fra dyr selv og giver derfor åbenlyse massebalanceproblemer - som man må spørge: ”Hvor får fisken proteinet fra?”. Sojabønner betragtes på visse måder som en god alternativ proteinkilde, men det giver også en række problemer med hensyn til bæredygtighed. For det første dyrkes sojabønner primært i det sydlige Amerika, hvor de har problemer med skovrydning på grund af kravet om dyrkbar jord til sojaproduktion. Skovrydning skaber problemer med hensyn til begge af de globale udfordringer, der er præsenteret ovenfor, klima og biodiversitet. For det andet er proteinindholdet i soja betydeligt lavere end fiskemel, så større mængder er påkrævet. Dette tilføjer kun til skovrydningsproblemet. En række alternative proteinkilder har vakt interesse i de seneste årtier. Blandt dem betragtes insektproteiner, andemadproteiner og *single-cell protein*. De tidligere kilder har styrker i sig selv, fx at vokse på industrielt affaldsmateriale såsom papirmasseaffald. Men i dette arbejde fokuserer vi på sidstnævnte, *single-cell protein*.

I denne afhandling præsenterer vi avancerede værktøjer til overvågning og økonomisk ikke-lineær modelprædiktiv regulering af et ikke-lineært system, der beskriver produktionen af *single-cell protein* i en ny bioreaktor med gode blande- og masseoverførselsegenskaber. Unibio A/S er et bioteknologifirma, der driver en patenteret U-loop bioreaktor til produktion af *single-cell protein*. I bioreaktoren vokser en bakteriekultur på billige substrater, f.eks. methanol eller metan, for at producere en proteinrig biomasse ($\sim$ 72%), der fungerer som en alternativ proteinkilde til supplementering af dyrefoder. Drift af U-loop reaktoren viser stive og ustabile dynamikker, hvilket kræver avancerede overvågnings- og processtyringssystemer for at opretholde stabil

og høj produktivitet og sikre økonomisk bæredygtighed af det alternative proteinprodukt, Uniprotein.

Emnet for denne afhandling er *economic optimising control*, eller *economic nonlinear model predictive control*, for en U-loop bioreaktor. *economic nonlinear model predictive control* beskriver en modelbaseret kontrolstrategi, der består af *state estimation*, der giver tilbagemelding om tilstanden baseret på systemmålinger og et *economic optimal control problem*, der giver åben sløjfe kontrolstrategier, der er optimale med hensyn til økonomiske mål. Vi præsenterer vækstmodeller for produktion af *single-cell protein* ved hjælp af bakterien *Methylococcus capsulatus* (Bath). Vi præsenterer modeller baseret på methanol og methan som kulstofkilder og flere forskellige kvælstofkilder; salpetersyre, ammoniak, ammonium, nitrit og nitrat og molekylært kvælstof. Derudover beskriver vi kemiske ligevægtsreaktioner. Disse kan inkluderes i vækstmodeller, hvilket resulterer i differential-algebraiske systemer, men tillader inkludering af målinger fra online sensorer, f.eks. pH-værdi.

I sammenhæng med *state estimation* til overvågning og kontrol, præsenterer vi metoder til ikke-lineære systemer involverende stokastiske differential ligninger og stokastiske differential-algebraiske ligninger. Vi fokuserer på metoder baseret på Bayesiansk inferens og overvejer ikke optimeringsbaserede metoder i dette arbejde. For systemer involverende differentialligninger, formulerer og implementerer vi fire metoder; det udvidede Kalman filter, unscentede Kalman filter, ensemble Kalman filteret og et partikelfilter. For systemer involverende differential algebraiske ligninger, formulerer og implementerer vi det udvidede Kalman filter. Det udvidede Kalman filter anvender direkte Kalman-filteropdateringen på en lokal linearisering af et ikke-lineært system og er derfor en direkte udvidelse af Kalman-filteret til ikke-lineære systemer. Det udvidede Kalman filter er i høj grad dikteret af fejlen forbundet med lineariseringen. Det uscentrede Kalman filter udbreder en sæt deterministisk samplede partikler, såkaldte sigma-punkter, og propagerer dem gennem de ikke-lineære tilstands- og måledynamikker. Dette resulterer i bedre konvergensegenskaber end den udvidede Kalman filter, men denne estimator kan fejle for meget ikke-lineære systemer. Ensemble Kalman filteret er en type partikelfilter, som approksimerer de ikke-lineærer tilstands- og målefordelinger med en sæt tilfældigt samplede partikler. Ensemble Kalman-filteret har stadig gaussiske antagelser, men undgår degeneringsproblemer ved at anvende Kalman-filteropdateringen på hver partikel i ensemblet når en måling er tilgængelig. Partikelfilteret approksimerer den ikke-lineære tilstand og målefordelinger med en sæt tilfældigt samplede partikler. I dette arbejde overvejer vi gaussisk målestøj, men filteret kan anvendes til vilkårlige målefordelinger, så længe likelihood kan beregnes for hver partikel i målefordelingen. Partikelfilteret kan have problemer med degenering, hvor partikelsættet konvergerer til et enkelt punkt, hvilket forårsager, at estimator fejler. Både ensemble Kalman filteret og partikelfiltre lider af dækningsproblemer for højdimmensionelle systemer, dvs. *curse of dimensionality.*

Vi præsenterer *economic nonlinear model predictive control* til produktion af *single-*

cell protein. Vi giver en konceptuel beskrivelse af kontrolteknologien og beskriver økonomisk optimale kontrolproblem. Modellerne for produktionen af *single-cell protein* er ofte stive og ustabile. Vi overvejer en direkte simultan tilgang til løsning af det ikke-lineære program for at håndtere systemets ustabilitet. Vi anvender en implicit Euler tidsdiskretisering for at håndtere stivheden.

Vi præsenterer numeriske eksperimenter af produktion af *single-cell protein* i en U-loop bioreaktor med en vækstmodel baseret på methanol. Vi præsenterer en optimal åben-sløjfe strategi for optimal profit baseret på et økonomisk optimalt kontrolproblem. Vi anvender den udvidede Kalman filter til *state estimation* i et åben sløjfe numerisk eksperiment. Til sidst anvender vi et *economic nonlinear model predictive control system* i et lukket sløjfe numerisk eksperiment på U-loop bioreaktoren. Det numeriske eksperiment viser, at kontrolsystemet kan stabilisere opstarten af reaktoren og opretholde stabil høj produktivitet. Derudover præsenterer vi en laboratorie-skala kontinuert bioreaktor til produktion af *single-cell protein*. I modellen for produktionen af *single-cell protein* inkluderer vi kemisk ligevægtsreaktionsdynamik, hvilket resulterer i et differential-algebraisk system. Vi præsenterer modellering, simulering, økonomisk optimal kontrol og *state estimation* for dette system. Vi løser et økonomisk optimalt kontrolproblem, der inkluderer de algebraiske ligninger. Vi anvender den udvidede Kalman filter i et åben sløjfe numerisk eksperiment ved at bruge den optimale kontrolstrategi, der er beregnet som løsningen på det økonomiske optimale kontrolproblem. Estimatoren konvergerer til de sande værdier af tilstandene og de algebraiske variable og estimerer en modelleret parameter nøjagtigt.

Endelig præsenterer og implementerer vi en *high-performance Monte Carlo simulation toolbox* implementeret i C ved hjælp af openmp til parallel Monte Carlo-simuleringer for lukkede-sløjfe systemer. I Monte Carlo simuleringer af lukketsløjfesystemer er alle simuleringer uafhængige af alle andre simuleringer, dvs. pinligt paralleliserbart. Problemet præsenterer derfor tæt på lineær skalering med antallet af kerner. Vi anvender softwaren til usikkerheds-kvantificering af fire forskellige kontrolsystemer i et numerisk lukket-sløjfe eksperiment på et fed-batch bioreaktoreksempel. Vi tester en åbensløjfe controller, to PID controllerer, en hånd-justeret og en med optimal tuning bestemt fra Monte Carlo-simuleringer, og en *nonlinear model predictive controller*.

Preface

This thesis was prepared at the Department of Applied Mathematics and Computer Science (DTU Compute) and at the Center for Energy Resources Engineering (CERE), technical university of Denmark, and at Unibio A/S in partial fulfillment of the requirements for acquiring a Ph.D. degree in applied mathematics.

The work presented in this thesis was carried out under supervision from Professor John Bagterp Jørgensen, Professor Krist V. Gernaey, Associate Professor Jakob Kjøbsted Huusom, Technical University of Denmark, and former CSO Ib Christensen and CTO Jess Dragheim, Unibio A/S.

This project was funded by Innovation Fund Denmark (9065-00269B).

Kongens Lyngby, April 17, 2023

[illegible]

Marcus Krogh Nielsen (mkrni)

Acknowledgements

I would like to thank my main university supervisor Professor John Bagterp Jørgensen for his advice, teaching, and general support through the duration of this project. His input has been crucial in my education and to the project as a whole. Furthermore, I would like to thank my university co-supervisors Professor Krist V. Gernaey and Associate Professor Jakob Kjøbsted Huusom for their input and support throughout this project. I would like to thank company supervisors Ib Christensen and Jess Dragheim for their advice and support and for making their brilliant technology available for research in a project such as this. Additionally, I would like to thank CSO Jens Dynesen for supervision during the later phases of my project, Michael Elleskov for his valuable input and expert opinions on the difficult production process, and Sten Bay Jørgensen for granting access to his vast pool of knowledge when it was needed. I would like to thank my university co-workers Asbjørn Thode Reenberg and Morten Wahlgreen Kaysfeld for their support and help both during the project and when writing the thesis. I would also like to thank my other co-workers Steen Hørsholt, Zhanhao Zhang, Sarah Ellinor Engell, and Anders Hilmar Damm Andersen for being such great colleagues during my time as a Ph.D. student. Finally, I would like to thank my girlfriend Louise Funch for the support she has given me throughout the project and especially during the final stages, as well as the rest of my family for their support and encouragement.

Contents

CHAPTER 1

Introduction

In this thesis, we present research on advanced monitoring and process control for single-cell protein (SCP) production by cultivation of the methanotrophic bacteria *Methylococcus capsulatus* (Bath) in a novel bioreactor. The work is conducted in collaboration with Unibio A/S (Unibio). Unibio is a biotechnology company producing SCP in a U-loop fermenter. The U-loop fermentation technology has been developed by Unibio to achieve good mixing and gas-liquid mass transfer properties for SCP production with cheap hydrocarbons, e.g. methanol and methane, as carbon sources.

1.1 Motivation

The modern global society faces a number of struggles with world-wide reach, e.g. climate change and biodiversity [1]. In response to this, a trend can be observed in developed countries toward more sustainable consumption [2], i.e. an increasing focus among consumers on the environmental impact of the products they consume. However, consumers display a general unwillingness to replace traditional protein sources, e.g. beef, pork, and chicken, with sustainable alternative protein sources [3]. As a result of this, and as the global population and wealth continues to increase, there seems to be a clear positive correlation with global population and global meat production and consumption [4]. Figure 1.1 illustrates the trends in global population and meat production. We must look for sustainable alternatives to the currently used sources of protein widely applied in the meat industry to address the need and demand for protein in a growing global population. Fishmeal and soy are examples of widely used animal feed protein supplements in the meat production industry, however these sources are not inherently sustainable [5, 6]. Fishmeal presents an obvious mass balance problem, as we utilise animal protein in the supplementation for other animals, including fish. This leads to an increasing need for fishmeal which leads to overfishing issues and thus also sustainability problems [7, 8]. Soy-based protein sources seem a superior alternative, as this is a crop and thus often more sustainable. However, the protein content in soybean meal is significantly lower than that in fishmeal, $\sim 45\%$ for soybean meal versus $\sim 65\%$ for fishmeal. As such, greater amounts of soy-based proteins are required for supplementation [9]. Additionally, soy requires vast areas of arable land, and as the protein demand increases, the demand for yet greater areas of arable land does as well. Soy is largely produced in South

America and this leads to deforestation in that area of the world which also has negative effects on both the climate and biodiversity [10].

Over the past decades, a number of alternatives to fishmeal and soy-based protein supplements have been investigated for animal feed and direct human consumption [11]. Examples of alternative sources are; insect protein [12, 13], duckweed protein [14, 15], and SCP [16–18]. These alternative sources utilise readily available feed stocks and industrial waste to produce proteins and do not present the same negative environmental impacts that fishmeal and soy do. In this work, we focus on SCP as an alternative protein source.

1.1.1 Single-cell protein

SCP describes the production of protein rich products by cultivation of microbial or algal cells [19]. The concept has been known both in industry and academia for decades and appear in the literature as early as the 1960s [20–22]. SCP can be produced from a number of different micro-organisms, e.g. fungi, yeasts, algae, and bacteria. The protein content varies depending on the method of production and choice of microorganism. The protein contents ranges from $\sim 15 - 45\%$ for fungi, $\sim 50 - 55\%$ for yeasts, $\sim 20 - 80\%$ for algae, and $\sim 50 - 80\%$ for bacteria [23]. As can be seen here, fungi provide inferior protein content compared to most other choices of microbe and algae display the greatest variability in protein content. SCP has been studied in some detail for both human and animal consumption [24], but in later decades the use has shifted toward protein supplements for animal feed in the meat industry [25]. SCP production using methanotrophic bacteria and with

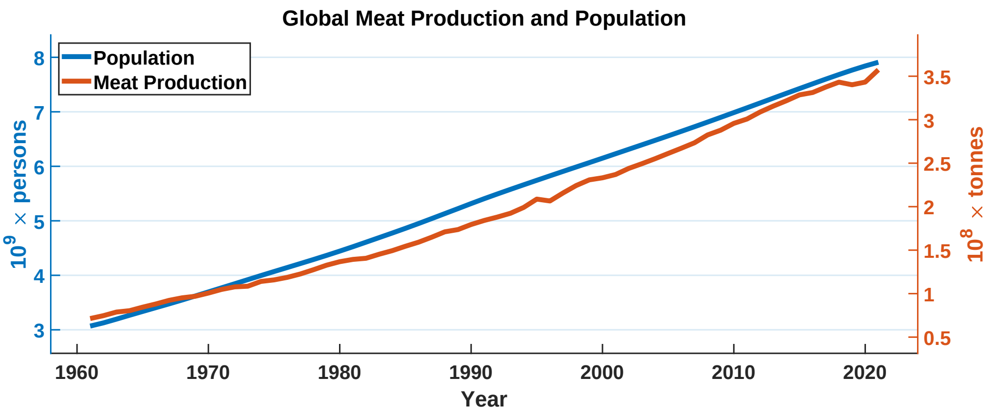


Figure 1.1: Trends in global population and global meat production in the years 1961-2021. Data is taken from the united nations' food and agriculture organization [4].

standard biomass is described by the stoichiometries [19]; for methane

$$\mathrm{CH_4} + 1.454\,\mathrm{O_2} \longrightarrow 0.520\,\mathrm{X} + 0.480\,\mathrm{CO_2}, \qquad r, \tag{1.1}$$

and for methanol

$$\mathrm{CH_3OH} + 0.954\,\mathrm{O_2} \longrightarrow 0.520\,\mathrm{X} + 0.480\,\mathrm{CO_2}, \qquad r, \tag{1.2}$$

where $X = \mathrm{CH_{1.8}O_{0.5}N_{0.2}}$ is the biomass. It is important that SCP is produced efficiently and using cheap substrates for it to be a financially viable alternative to existing protein supplements. For that reason, this work focusses on fermentation using methanotrophic bacteria. Methanotrophic bacteria, also known as methanotrophs, are methylotrophs. Methanotrophs are bacteria capable of utilising cheap hydrocarbons, e.g. methane and methanol, as carbon and energy sources. Therefore, these bacteria can be used for efficient SCP production under sufficiently favourable growth conditions [26]. *Methylococcus capsulatus* (Bath) is a type X (earlier type I) methanotroph and is currently used for SCP production Unibio A/S due to its good growth properties, high protein content, and favourable amino-acid profile [27]. The metabolism of *M. capsulatus* is studied in detail and described in the literature [28]. SCP can be produced by fermentation of *M. capsulatus* with either methanol or methane as carbon source, but the availability, purity, and price of methane in the form of natural gas or biogas makes it the most economical candidate for SCP production [29]. However, methane is a gas and therefore involves mass transfer from the gas phase to the liquid phase. As indicated by (1.1) and (1.2), growth on methane is more oxygen intensive than growth on methanol. This means that the mass transfer of methane to the liquid phase is limited by both the reactor properties and the increased competition with oxygen mass transfer. These factors, along with the explosive risks related to methane/oxygen rich gas mixture, have traditionally made it unfavourable to produce SCP with methane as carbon source. However, if high mass transfer between the gas and liquid phases can be efficiently achieved, fermentation of *M. capsulatus* with methane as carbon source could be a potential disruptor in the market of protein supplements for animal feed.

1.1.2 U-loop bioreactor

To address the problem of energy efficiency and poor mass transfer properties of traditional bioreactors, Unibio A/S has designed a novel reactor; the U-loop fermenter [30]. The U-loop reactor is a continuous bioreactor. In the reactor, the fermentation broth is recirculated through a U-shaped pipe in through a series of static mixers to achieve good mixing and high mass transfer [31, 32]. Unibio operates two such U-loop reactors at their facility in Kalundborg, Denmark; a $\sim 150L$ pilot-scale reactor and a $\sim 2200L$ demonstration-scale reactor. Figure 1.2 illustrates the Unibio's SCP production process. Several models describing growth of *M. capsulatus* for SCP production in a U-loop reactor have been presented [33–35]. The growth and reactor dynamics show that the optimal start-up trajectory of the reactor is unstable, i.e. an unstable retractor requiring active control to not diverge from the optimal trajectory [36]. Furthermore, the reactor is unstable at high concentrations because optimal steady state productivities are placed close to regions of low productivity [37]. This calls for model-based monitoring tools to guide operator decisions and advanced process control solutions to stabilise production and reach high productivity.

1.2 Economic model predictive control

Model predictive control (MPC) is a model-based feedback control strategy [39, 40]. MPC can be described in terms of two main components;

- state estimation,
- and economic optimal control problem (EOCP).

State estimation describe methods which reconstruct unmeasured model states from system measurements. In the context of MPC, optimisation involves solving and OCP. An OCP is a dynamical optimisation problem which describes optimal system behaviour, i.e. model behaviour, with respect to a pre-defined set of performance measures over a finite horizon. MPC may be further separated into applications in control of linear systems, so-called linear MPC (LMPC), and applications in nonlinear systems, so-called nonlinear MPC (NMPC). For LMPC, the state estimation algorithm of choice is the Kalman filter which provides optimal state estimates in linear systems with Gaussian process and measurement noise [41, 42]. Furthermore, the OCP in LMPC can in most cases be formulated as a convex quadratic program (QP). Convex optimisation problems are characterised by having only one global minimum, i.e. convergence guarantees a global optimum. Additionally, powerful open-source and commercial solvers exist for the solution of convex QPs [43–48]. For nonlinear systems, several state estimation algorithms exist and the best choice of algorithm depends on the application. State estimation in nonlinear systems can be separated into two main categories; optimisation-based method, e.g. moving-horizon estimation, and methods based on Bayesian inference, e.g. the extended Kalman filter and particle filters. In this work, we focus on Bayesian methods. The OCP in NMPC is a dynamical optimisation problem that, when discretised, is a general nonlinear program (NLP). For such optimisation problems, we usually consider locally optimal solutions as computed by numerical local optimisation algorithms, e.g. sequential quadratic programming (SQP) methods or interior-point (IP) methods [49, 50]. The inclusion of numerical optimisation in MPC implementations make them more computationally expensive than classical controllers, e.g. proportional-integral-derivative controllers (PIDs) or linear-quadratic-regulators (LQRs), for which explicit closed-form solutions can be formulated. MPC have advantages over classical controllers, e.g. anticipatory action and the inclusion of process constraints. Anticipatory action describes the ability of the MPC to anticipate future behaviour of a system and

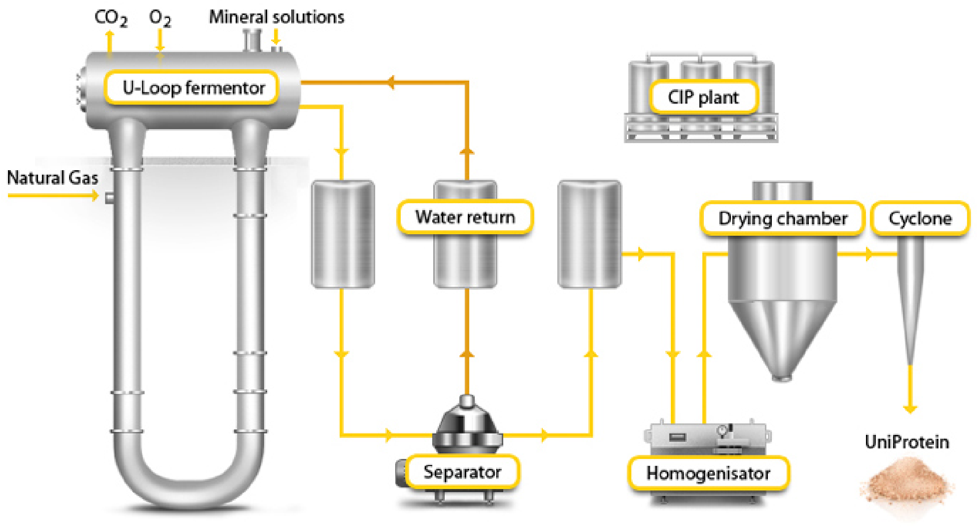


Figure 1.2: Conceptual illustration of single-cell protein production using a U-loop bioreactor. The illustrations is from Unibio A/S [38].

act in response to that future behaviour, due to the inclusion of model predictions. For process constraints, we may include operational and physical limitations in the system directly as constraints in the OCP. An additional advantage of MPC is that the objective of the OCP can be any nonlinear function of the states and inputs, e.g. target tracking and input regularisation, but also economic measures of performance, e.g. production revenue, input or power cost, or carbon footprint related to production. LMPCs and NMPCs with such economic objectives are called economic LMPC (ELMPC) and economic NMPC (ENMPC). In this work, we consider ENMPC for control of the SCP production in a U-loop bioreactor. Figure 1.3 illustrates an MPC system integrated with a process.

1.2.1 State estimation

Process operation, manual or automatic, is naturally limited by the information directly available from process measurements. However, effective decision-making may often be best done in response to internal unmeasurable states of a process, such as concentrations of substrates or inhibiting components. State estimation addresses this issue of feedback given a model of the process. In state estimation, the dynamical correlation between measured and unmeasured process states are used to infer unmea-

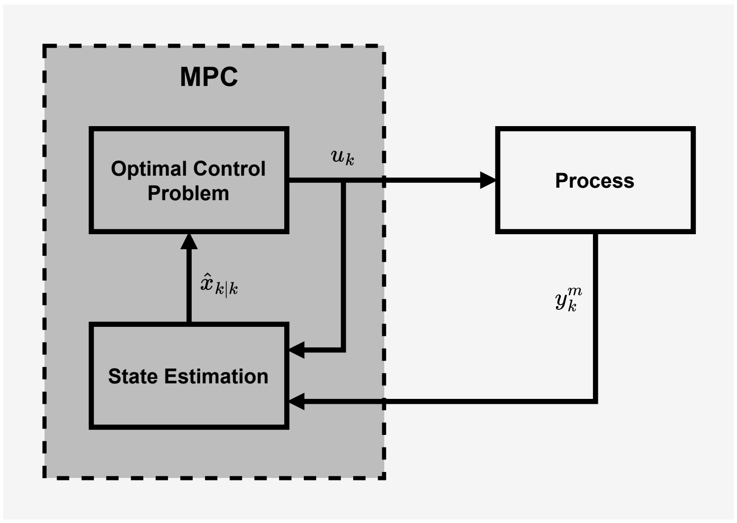


Figure 1.3: Illustration of a model predictive control system integrated with a process.

sured quantities. This means that even though only a few quantities can be measured directly, it is possible to gain information about unmeasured quantities through those measurements. State estimation algorithms may be separated into at least two categories; optimisation-based methods and methods based on Bayesian inference. In this work, we focus on methods based on Bayesian inference. State estimation methods based on Bayesian inference provides both estimates of state variables as well as the uncertainties related to each state estimate. This provides operators, or automatic control systems, information about the unmeasured states and the reliability of the estimates. This allows for more effective decision-making when operating a process. In the context of model-based control, the state estimator closes the control-loop by updating model states with feedback from system measurements. The Kalman filter provides optimal state estimates and has since its introduction in the 1960s [41, 42] become ubiquitous in model-based monitoring and control systems. However, the Kalman filter is limited to systems with linear dynamics. For nonlinear systems, the state evolution is governed by the Fokker-Planck equation (also Kolmogorov's forward equation). However, this is a partial differential equation with the same dimension as the state, making it's application infeasible for systems with more than a few states [51]. Figure 1.4 illustrates state estimation for a system with a measured and unmeasured variable.

We consider nonlinear state estimation algorithms as the U-loop bioreactor is a nonlinear system. State estimation in nonlinear systems are described in [51–53]. We consider four Bayesian methods of state estimation; the extended Kalman filter [51, 54], the unscented Kalman filter [55–61], the ensemble Kalman filter [62–65], and a particle filter [66–69]. The extended Kalman filter is a direct extension of the linear Kalman filter and applies the Kalman filter update to a local linearisation of the nonlinear system. The extended Kalman filter is a computationally efficient state estimation algorithm, but its performance is largely dictated by the linearisation error [66, 70]. The unscented Kalman filter propagates a set of deterministically sampled particles, so-called sigma-points, through the nonlinear state and measurement dynamics. As such, the unscented Kalman filter has better convergence properties for nonlinear systems, but can still be insufficient for highly nonlinear systems [71]. The ensemble Kalman filter is a type of particle filter which utilises the Kalman update in the filtering step. The state and measurement distributions are then represented by propagated and filtered particles. This approach has proven to be effective for highly nonlinear systems, but suffer from the the curse of dimensionality [72]. The particle filter, similarly to the ensemble Kalman filter, performs well for highly nonlinear systems and can be applied regardless of noise distributions even if the measurement noise is non-Gaussian. However, it also suffers from coverage issues for high-dimensional systems and can suffer from particle degeneracy, where the particle set collapses to a single point in the state-space [73]. We describe all four filters for nonlinear systems involving stochastic differential equations (SDEs). Additionally, we consider the extended Kalman filter for nonlinear systems involving stochastic differential algebraic equations (SDAEs). We include differential algebraic systems to include chemical equilibrium reactions describing readily available online measurements, e.g. pH-value and conductivity. State estimators for related systems in the petro-chemical industry have been described in the literature [74].

1.2.2 Economic optimal control

The OCP describes the optimal system behaviour over a finite horizon with respect to one or more chosen measures of performance. Consider the OCP formulated for a nonlienar system involving SDEs

$$\min_{[x(t);u(t)]_{t_0}^{t_f}} \quad \phi, \tag{1.3}$$

subject to

$$\begin{aligned} x(t_0) &= x_0, && (1.4a)\\ \frac{dx}{dt}(t) &= f(x(t),u(t),\theta), && (1.4b)\\ z^m(t) &= g^m(x(t),\theta), && (1.4c)\\ c_{\min}(t) \le c(&x(t),u(t),\theta) \le c_{\max}(t). && (1.4d) \end{aligned}$$

The objective function, ϕ, describes the performance measures we wish to optimise the process with respect to. Examples of performance measures may be traditional

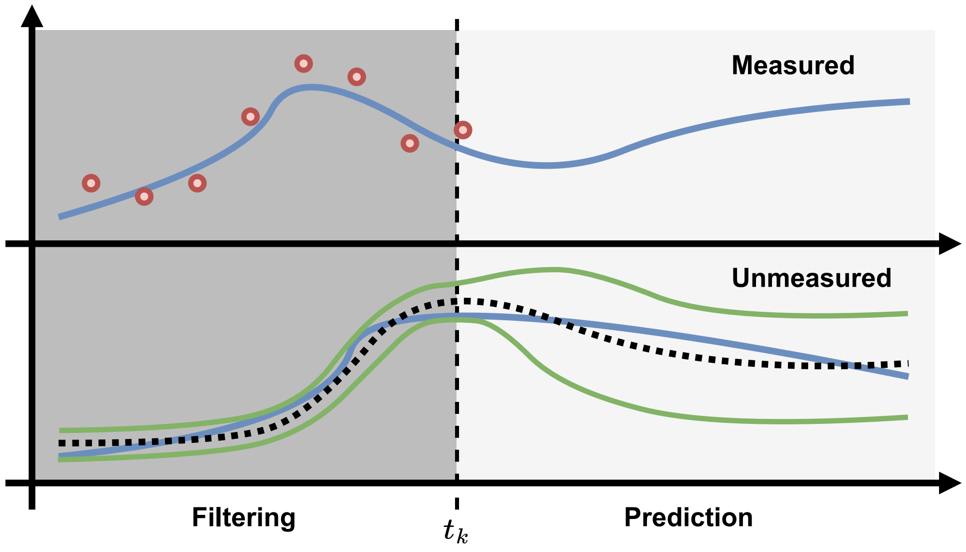


Figure 1.4: Illustration of state estimation in a system with a measured and unmeasured variable described by a model. The red dots are measurements, the blue lines are true values of the variables, the black dotted line is the estimate of the unmeasured variable, and the green lines illustrate the uncertainty in the estimate. Notably, the uncertainty is larger when we predict the future behaviour of a system and lower when we have measurement information.

measures, e.g. target tracking, input rate-of-movement penalty, or input regularisation, but it may also be economic measures, e.g. production revenue, input cost, or even measures of sustainability, such as carbon footprint. In essence, ϕ may be any nonlinear function of the state and input variables. In this work, we are controlling a production process and as a result of this, economic performance measures are of particular interest. The constraints in (1.4a)-(1.4c) defines the system dynamics and (1.4d) describes operations and physical constraints on the system, e.g. nonnegativity of variables, maximum reactor volume, and flow-rate limits. The OCP presented in (1.3)-(1.4) is in continuous-time. We apply a temporal discretisation to apply numerical optimisation algorithms. Three direct approaches to the formulation of the OCP are [75];

- direct single-shooting,
- direct multiple-shooting,
- and direct simultaneous approaches.

In direct single-shooting, all state variables are eliminated by the introduction of a simulator. The states are determined only by the choice of manipulated inputs and the initial condition for the state. This approach results in a lower-dimensional dense NLP, which can be useful for computational efficiency in real-time applications. However, the direct single-shooting approach can have trouble converging for systems with unstable dynamics. The multiple-shooting approach similarly introduces a simulator, but only simulates between discrete sampling intervals. This approach results in a higher dimensional semi-sparse NLP, but with better convergence properties for systems with unstable dynamics. The direct simultaneous approach includes the temporal discretisation in the constraints of the OCP directly. This results in a high-dimensional sparse NLP, but with good convergence properties for systems with unstable dynamics [75]. In this work, we apply the direct simultaneous approach to a stiff and unstable system. We note that the higher dimensional NLPs arising from multiple-shooting and simultaneous approaches do not necessarily result in less computationally efficient algorithms, as the use of sparse solvers can be applied for these formulations [50]. Figure 1.5 illustrates a trajectory optimisation for an OCP with a target tracking objective.

1.3 Objectives and contributions

The main objectives of this work are to;

1. describe growth models for cultivation of *Methylococcus capsulatus* (Bath) for the purpose of SCP production,
2. describe a dynamical model for the U-loop bioreactor,
3. formulate and test state estimation methods in nonlinear systems involving stochastics,
4. formulate an economic nonlinear model predictive control system for direct optimisation of economic performance measures,
5. and formulate methods of investigating performance of nonlinear control solutions.

In this work, we describe several different models describing growth of *Methylococcus capsulatus* (Bath). We present models based on both methanol and methane as carbon and energy sources, as well as several different nitrogen sources; nitric acid, ammonia, ammonium, nitrite, nitrate, and molecular nitrogen. We apply a compact framework for describing the stoichiometry in reactive systems and include chemical equilibrium reactions modelling acid/based equilibrium dynamics. We describe compact reactor models for batch reactors (BRs), fed-batch reactors (FBRs), continuous stirred tank reactors (CSTRs), and plug flow reactors (PFRs) and combine some of

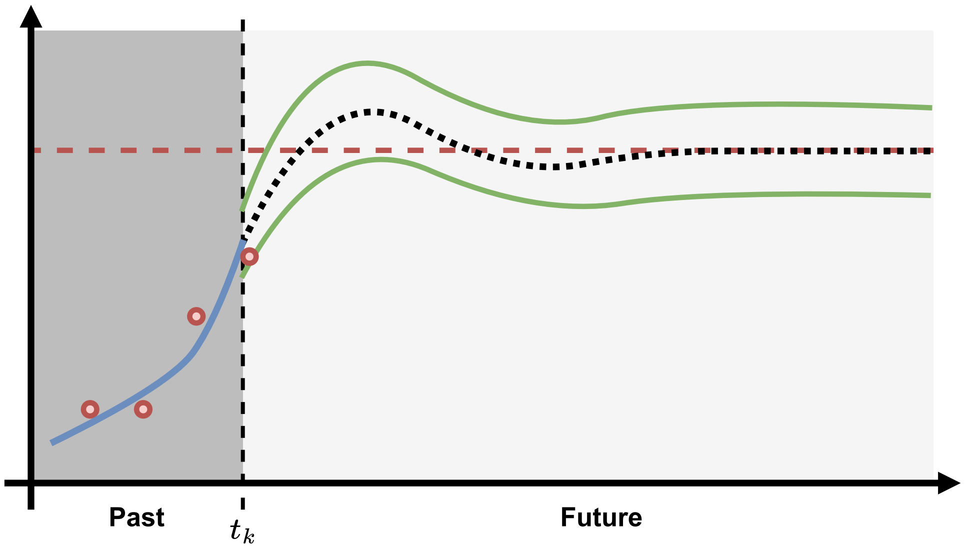


Figure 1.5: Illustration of an optimal trajectory computed as the solution to an OCP with a setpoint tracking objective. The red line is the target, the red dots are observations, the blue line is the filtered state estimation, the black dotted line is the optimal trajectory, and the green lines are uncertainties of the estimate computed by the state estimator.

these compartments, i.e. CSTR and PFR, to describe the U-loop bioreactor dynamics. The growth and reactor dynamics results in SCP production models involving nonlinear SDEs and SDAEs. We formulate four different state estimation methods for monitoring and control for continuous-discrete nonlinear systems involving SDEs; the extended Kalman filter, the unscented Kalman filter, the ensemble Kalman filter, and a particle filter. The choice of algorithm is depends on factors such as; nonlinearity and requirements for computational efficiency in real-time application. These advantages and disadvantages are discussed. Furthermore, we formulate the extended Kalman filter for continuous-discrete nonlinear systems involving SDAEs. This state estimation method can be applied in the differential algebraic systems arising from models including chemical equilibrium reactions. This is especially relevant when including common online measurements, e.g. pH-value and conductivity. We describe EOCPs and apply a direct simultaneous approach and an implicit numerical integration scheme to address the stiffness and instability of the U-loop reactor for SCP production. We formulate an ENMPC system which directly optimises economic performance measures in the SCP production process. We perform numerical experiments testing the ENMPC systems. Finally, we describe and implement a high-performance Monte Carlo simulation toolbox for tuning and performance quantification, i.e. uncertainty quantification (UQ), in closed-loop control systems.

1.4 Outline of the thesis

This work is structured as seven parts; modelling, estimation algorithms, economic model predictive control, numerical examples, controller performance quantification and tuning, conclusions and suggestions for future work, and appendix.

Part I - Modelling (Chapters 2-6)

In this part, we formulate models relevant to SCP production and describe model formulations and numerical solution details. In Chapter 2 we present a compartment model describing the reactor dynamics of the U-loop bioreactor. In Chapter 3, we describe and discuss models describing growth of the methanotrophic bacteria *Methylococcus capsulatus* (Bath) for SCP production. In Chapter 4, we describe models describing chemical equilibrium reactions in the context of SCP production. Chapter 5 describes modelling and numerical simulation of nonlinear systems involving SDEs. Finally, Chapter 6 describes modelling and numerical simulation of nonlinear systems involving SDAEs.

Part II - Estimation Algorithms (Chapters 7-8)

In this part, we formulate state estimation methods for nonlinear systems involving stochastic differential and differential algebraic equations (DAEs) model in the form described in Part I. In Chapter 7, we describe four state estimation methods for continuous-discrete nonlinear systems involving SDEs; the extended Kalman filter, unscented Kalman filter, ensemble Kalman filter, and a particle filter. Chapter 8 describes the extended Kalman filter for continuous-discrete nonlinear systems involving SDAEs.

Part III - Economic Model Predictive Control (Chapter 9)

In this part, we describe economic nonlinear model predictive control for SCP production. In Chapter 9, we describe economic model predictive control and the integration of the two main components; the state estimator and the EOCP. We present objective formulations for the EOCP, including; target tracking, input rate-of-movement penalty, as well as economic performance measures, such as production revenue and input cost.

Part IV - Numerical Examples (Chapters 10-11)

In this part, we present numerical examples of simulation, state estimation, economic optimal control, and economic nonlinear model predictive control for SCP production. In Chapter 10, we describe modelling, state estimation, economic optimal control, and economic nonlinear model predictive control for SCP production in a U-loop bioreactor. The model presented in this chapter is a continuous-discrete nonlinear system involving SDEs based on a growth model with methanol and nitric acid and carbon and nitrogen sources, respectively. In Chapter 11, we describe modelling, simulation, state estimation, and economic optimal control for a laboratory-scale CSTR for SCP production. The model described in this chapter includes chemical equilibrium reactions, resulting in a continuous-discrete nonlinear system involving SDAEs based on a growth model with methane and ammonium as carbon and nitrogen sources, respectively.

Part V - Controller Performance Quantification and Tuning (Chapter 12)

In this part, we present a method of quantifying the performance of a control systems for tuning and uncertainty quantification, i.e. testing controller variability and robustness. In Chapter 12, we describe a high-performance Monte Carlo simulation toolbox and present an implementation in C using openmp for parallelisation of the Monte Carlo simulations. The toolbox is applied for tuning and uncertainty quantification of controller performance in closed-loop systems, testing open-loop, PID, and NMPC.

Part VI - Conclusions and Suggestions for Future Work (Chapter 13)

In this part, we present conclusions for the work presented in this thesis, as well as suggestions for future work based on and related to the work presented in this thesis on advanced monitoring and control solutions for SCP production.

Part VII - Appendix (Appendices A-G)

In this part, we present papers and a technical report on which this work is partially based. In Appendix A, we present a paper describing modelling and economic optimal control for SCP production in a laboratory-scale CSTR. In Appendix B, we present a publication describing state estimation in continuous-discrete nonlinear systems involving SDEs. In Appendix C, we present a publication describing the formulation and implementation of a high-performance Monte Carlo simulation toolbox for uncertainty quantification in closed-loop systems. In Appendix D, we present a publication describing economic optimal control for SCP production in a U-loop bioreactor applying direct simultaneous approaches. In Appendix E, we present a publication describing the application of the extended Kalman filter for state estimation in the U-loop bioreactor for SCP production. In Appendix F, we present a publication describing economic nonlinear model predictive control for SCP production in a U-loop bioreactor. Finally, Appendix G presents a technical report describing a compact modelling framework for reactive systems, as well as models and implementation of BRs, FBRs, CSTRs, and PFRs applying the modelling framework.

1.5 Publications and presentations

In this section, we present the papers and presentations prepared in relation to the PhD project.

1.5.1 Paper I

Modelling and economic optimal control for a laboratory-scale continuous stirred tank reactor for single-cell protein production.

This paper introduces a novel model for the growth and pH dynamics of *Methylococcus capsulatus* in a laboratory-scale fermenter for SCP production. The model includes DAEs, non-negativity constraints, and linear scaling for stability in simulations. The study concludes with a numerical example for economic optimal control of biomass growth and pH tracking in the reactor.

1.5.2 Paper II

State estimation in continuous-discrete-time nonlinear stochastic systems

This paper presents and compares four state estimation methods for continuous-discrete nonlinear systems involving SDEs; the extended Kalman filter, unscented Kalman filter, ensemble Kalman filter, and a particle filter. The methods are implemented in Matlab and evaluated using simulations of a modified four-tank system, with accuracy determined by mean absolute percentage error. The study provides an overview and comparison of state estimation methods for continuous-discrete time nonlinear stochastic systems, aiding efficient implementation.

1.5.3 Paper III

A high-performance Monte Carlo simulation toolbox for uncertainty quantification of closed-loop systems

This paper presents a parallel Monte Carlo simulation toolbox for uncertainty quantification and tuning of controllers in nonlinear closed-loop systems. The toolbox demonstrates efficient scaling on multiple cores. The toolbox is applied in a numerical experiment for closed-loop control of a bioreactor in fed-batch operation. We test an open-loop controller, two PID controllers, and an NMPC and apply the Monte Carlo toolbox to quantify the performances.

1.5.4 Paper IV
Economic optimal control of a U-loop bioreactor using simultaneous collocation-based approaches

This paper studies economic optimal control of SCP production in a U-loop reactor. The model of the reactor contains both ordinary and partial differential equations, resulting in a large-scale and challenging OCP. Two simultaneous collocation-based approaches are applied to solve the problems and are implemented in C using IPOPT for solution of the NLP. A performance study shows that it is feasible to solve the EOCPs in real-time.

1.5.5 Paper V
The extended Kalman filter for nonlinear state estimation in a U-loop bioreactor

This paper studies nonlinear state estimation in a U-loop reactor for SCP production. The model of the reactor is a combination of stiff stochastic partial differential equations and SDEs, resulting in a high-dimensional system of 83 states. The paper investigates and discusses the use of the continuous-discrete extended Kalman filter for state estimation in this challenging system.

1.5.6 Paper VI
Economic nonlinear model predictive control of a U-loop bioreactor

A novel algorithm for economic nonlinear model predictive control of SCP production in a U-loop bioreactor is presented. The model of the reactor consists of 87 state variables and combines stochastic ordinary and partial differential equations. The algorithm combines a continuous-discrete extended Kalman filter and a direct simultaneous collocation based approach to the discretisation of the OCP and is computationally feasible for real-time implementation in the U-loop reactor.

1.5.7 Technical Report I

Modelling of reactive systems

This technical report presents a compact approach to the description of stoichiometry and kinetics for reactive systems, e.g. chemical and biochemical systems. The report describes the modelling framework and presents example kinetics for chemical and biochemical reactive systems. In the report, we present mass balances and differential equation models for four reactor types applying the modelling approach. The four types are; BRs, FBRs, CSTRs, and PFRs. Finally, we present examples of a chemical and biochemical reactive systems in the presented reactor types. The appendix of the report describes gas-liquid mass transfer, as well as Jacobians and implementation in Matlab and Python of the presented reactor types.

1.5.8 Presentations

1. Nielsen, M. K., Ritschel, T. K. S., Christensen, T., Dragheim, J., Huusom, J. K., Gernaey, K. V., Jørgensen, J. B. (2023). State estimation for continuous-discrete-time nonlinear stochastic systems. Foundations of Computer Aided Process Operations/Chemical Process Control (FOCAPO/CPC) 2023. (Poster)

2. Nielsen, M. K., Dynesen, J., Dragheim, J., Jørgensen, S. B., Huusom, J. K., Gernaey, K. V., Jørgensen, J. B. (2022). Economic MPC for single cell protein production in a U-loop reactor. CERE Discussion Meeting 2022/KT Konsortium Meeting 2022.

3. Nielsen, M. K., Dynesen, J., Dragheim, J., Jørgensen, S. B., Huusom, J. K., Gernaey, K. V., Jørgensen, J. B. (2022). Economic MPC for single cell protein production in a U-loop reactor. CERE Discussion Meeting 2022/KT Konsortium Meeting 2022. (Poster)

4. Nielsen, M. K., Ritschel, T. K. S., Christensen, I., Huusom, J. K., Gernaey, K. V., Jørgensen, J. B. (2022). 2022 Nordic Process Control Workshop (NMPW). (Poster)

1.5.9 Papers not included in the thesis

1. Kaysfeld, M. W., Kumar, D., Nielsen, M. K., Jørgensen, J. B. (2023). Dynamic optimization for monoclonal antibody production. In submission for the 22nd IFAC world congress.

2. Diaa-Eldeen, T., Nielsen, M. K., Berg, C. F., Hovd, M. Jørgensen, J. B. (2023). Data assimilation for combined parameter and state estimation in stochastic continuous-discrete nonlinear systems. In submission for the 21st IEEE European Control Conference (ECC).

3. Ryde, T. E., Wahlgreen, M. R., Nielsen, M. K., Hørsholt, S., Jørgensen, S. B., Jørgensen, J. B. (2021). Optimal feed trajectories for fedbatch fermentation with substrate inhibition kinetics. IFAC PapersOnLine, 54(3), (318–323).

4. Zhang, Z., Nielsen, M. K., Hørsholt, S., Muralidharan, G., Jørgensen, J. B. (2021). Digitalization, Control and Optimization for Cement Plants. In proceedings of the 31st European Symposium on Computed Aided Process Engineering (ESCAPE), (1319–1324).

5. Zhang, Z., Nielsen, M. K., Muralidharan, G., Hørsholt, S., Jørgensen, J. B. (2022). Model predictive control for blending processes in cement plants. IFAC PapersOnLine, 55(7), (483–488).

Part I

Modelling

CHAPTER 2

U-loop Bioreactor

The work presented in this chapter is partially based on the publications listed in Appendices D, E, and F on control for a U-loop bioreactor, and the technical report listed in Appendix G on modelling of reactive systems. In this chapter, we consider the U-loop bioreactor for SCP production. The U-loop fermenter is designed and operated by Unibio A/S at their production and development facility in Kalundborg, Denmark. At the facility, reactors are operated at pilot- and demonstration-scale. The pilot-scale reactor is operated exclusively for research and development and the demonstration-scale reactor is operated for both research and development and production. In this chapter, we present a compartment model describing the dynamics of the U-loop bioreactor. We also describe the layout, actuator, and sensor capabilities for the pilot- and demonstration-scale U-loop fermenters currently operated by Unibio A/S in Kalundborg, Denmark.

The chapter is structured as follows: in section 2.1, we motivate the model for the U-loop reactor, we consider in this work and describe the U-loop fermenters currently operated by Unibio A/S at their facility in Kalundborg, Denmark. In this section, we also provide schematics describing the actuator and sensor layout of the pilot- and demonstration-scale reactors located and operated at the Kalundborg site. In section 2.2, we describe models for the major compartments in the U-loop reactor model and finally present a description of the complete U-loop reactor model considered in this work. Section 2.3 summarises the work presented in this chapter.

2.1 U-loop fermenter

In this section, we consider the U-loop reactor compartment model presented by [31, 33]. The compartment model of the U-loop reactor is defined by four compartments; the top tank, U-loop leg, inlet mixer, and gas-liquid separator. We model the top tank as a CSTR. The U-loop leg is modelled as a PFR and the resulting system of partial differential equations (PDEs) is discretised using a finite-volume discretisation to arrive at a system of ordinary differential equations (ODEs). We inlet mixer models the mixing of inlet flows of gasses and liquids, and the liquid recycle flow from the top tank. The gas-liquid separator describes the separation of gasses and liquid as they flow back into the off-gas outlet and top tank, respectively. The resulting model is a system of ODEs. Figure 2.1 illustrates the flow in the U-loop fermenter.

The U-loop fermenter is a continuously operated bioreactor. The reactor layout is designed with high flow rate recirculation and static mixers along the U-loop leg for good mixing and high gas-liquid mass transfer properties [32]. In the bioreactor, a culture of *Methylococcus captulsatus* (Bath) grows to a protein rich fermentation broth. Gaseous and liquid substrates are continuously added and mixed with the fermentation broth as it flows through the U-loop leg. As the content reaches the top tank, it de-gasses as the pressure drops. The gas exits through the off-gas outlet stream and the liquid content is either recirculated through the U-loop leg or harvested through the liquid outlet if the biomass concentration is high enough for down-stream processing. Unibio A/S currently operates two scales of U-loop fermenters at the facility in Kalundborg, Denmark; a pilot-scale reactor for research and development, and a demonstration-scale reactor used for research and development and SCP production. Figure 2.2 describes the actuator and sensor layouts for the pilot-scale fermenter. Figure 2.3 describes the actuator and sensor layouts for the demonstration-scale fermenter.

2.2 Modelling the U-loop reactor

In this section, we present the model of the U-loop bioreactor described by [31, 33, 36]. We describe models of the top tank, U-loop leg, inlet mixer, and gas-liquid separator. We model the top tank as a CSTR with variable volume. We model the U-loop leg as a PFR. We apply a finite-volume discretisation to discretise the PDE describing the PFR. The discretised PDE gives rise to an ODE for the U-loop leg. Figure 2.4 is a flow diagram illustrating the U-loop reactor model.

2.2.1 CSTR model

A CSTR is a single closed-vessel reactor with both in- and outflows of liquid (and potentially gasses). In the CSTR, we assume that the context is completely and perfectly mixed at all points in the reactor. This means that concentrations at all points in the reactor are assumed identical [76–78]. The CSTR is described by the system of ODEs

$$\frac{dV}{dt} = e^T F - e_{\text{Out}}^T F_{\text{Out}}, \qquad V(t_0) = V_0, \tag{2.1a}$$

$$\frac{dn}{dt} = C_{\text{In}} F - c e_{\text{Out}}^T F_{\text{Out}} + R(c)V, \qquad n(t_0) = n_0, \tag{2.1b}$$

where $V \in \mathbb{R}$ is the liquid volume, $n \in \mathbb{R}^{n_c}$ are the mole numbers, and $c = n/V$ are the concentrations. $F \in \mathbb{R}^{n_{u,\text{In}}}$ is the inlet flow rate vector and $F_{\text{Out}} \in \mathbb{R}^{n_{u,\text{Out}}}$ is the outlet flow rate vector. $C_{\text{In}} \in \mathbb{R}^{n_c \times n_{u,\text{In}}}$ are the inlet concentrations. $e \in \{1\}^{n_{u,\text{In}}}$ and $e_{\text{Out}} \in \{1\}^{n_{u,\text{Out}}}$ are vectors of ones, i.e. for computing the sum of the in- and outlet flows. We compute the production rates using the reactive systems modelling

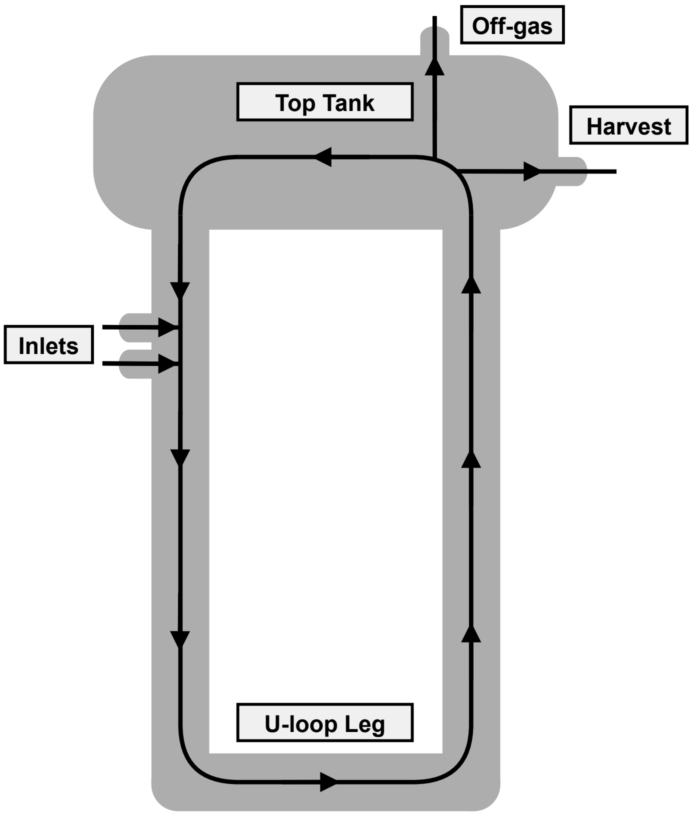


Figure 2.1: Illustration of the U-loop reactor. Substrates are added to the reactor fluid as it enters and flows through the U-loop leg to and from the top tank. Biomass rich reactor fluid is harvested from the top tank to produce single-cell protein.

framework presented in Appendix G, as

$$R(c) = S^T r(c), \tag{2.2}$$

where $S \in \mathbb{R}^{n_c \times n_r}$ is the stoichiometric matrix, $r(c) : \mathbb{R}^{n_c} \longrightarrow \mathbb{R}^{n_r}$ is a vector of the reaction rates for each reaction, and $R(c) : \mathbb{R}^{n_c} \longrightarrow \mathbb{R}^{n_c}$ are the production rates for each component. We derive the system of ODEs governing the CSTR from mass balances. The technical report listed in Appendix G describes the derivation.

2.2.2 PFR model

A PFR is an idealised pipe reactor, where the mixture completely fills the reactor and moves as a plug through the system, hence the name. In the PFR, we assume that the reactor content is perfectly mixed through any pipe cross-section normal to the linear flow direction, such that we can model the one-dimensional flow through the pipe. Additionally, we assume that the linear velocity of the mixture is uniform along the length of the reactor [76–78]. We describe the PFR by the system of PDEs

$$\frac{\partial c}{\partial t}(t,z) = -\frac{\partial N}{\partial z}(t,c(t,z)) + Q(t,c(t,z)), \qquad c(t_0,z) = c_0(z), \tag{2.3}$$

where c are concentrations, $N(\cdot)$ are fluxes, and $Q(\cdot)$ are generation rates, i.e. production and mass transfer rates. We consider Danckwerts' boundary conditions [79], as

$$c(t,0) = c_{\mathrm{In}}(t), \tag{2.4}$$

and

$$\frac{\partial c}{\partial z}(t,0) = 0, \qquad \frac{\partial c}{\partial z}(t,L) = 0, \tag{2.5}$$

where L is the total length of the reactor. We define the concentrations as

$$c(t,z) = \begin{bmatrix} c_l(t,z) \\ c_d(t,z) \\ c_g(t,z) \end{bmatrix}. \tag{2.6}$$

The concentrations are comprised of the concentration of the components only dissolved in the liquid phase, c_l, the gaseous components dissolved in the liquid phase, c_d, and the gaseous components in the gas phase, c_g. The fluxes are

$$N(t,c(t,z)) = v(t)c(t,z) - D\frac{\partial c}{\partial z}(t,z), \tag{2.7}$$

where $v(t)$ is the linear velocity and D is a diagonal matrix of dispersion coefficients for each components. The linear velocity is

$$v(t) = \frac{F_t(t)}{A}, \qquad F_t(t) = F_{t,l}(t) + F_{t,g}(t), \tag{2.8}$$

where A is the (assumed) constant cross-sectional area of the reactor pipe, F_t is the total flow rate, $F_{t,l} = e_l^T F_l$ is the total liquid flow rate, and $F_{t,g} = e_g^T F_g$ is the total gas flow rate. The generation term is

$$Q(t,c(t,z)) = \begin{bmatrix} Q_l(t,c(t,z)) \\ Q_d(t,c(t,z)) \\ Q_g(t,c(t,z)) \end{bmatrix}, \tag{2.9}$$

where the generation is defined for the components only present in the liquid phase, Q_l, the gaseous components dissolved in the liquid phase, Q_d, and the gaseous components in the gas phase, Q_g. The generation terms are

$$Q_l(t, c(t, z)) = R_l(c(t, z)), \tag{2.10a}$$

$$Q_d(t, c(t, z)) = R_d(c(t, z)) + \frac{1}{1 - \epsilon(t)} J_{gl}(c(t, z)), \tag{2.10b}$$

$$Q_g(t, c(t, z)) = R_g(c(t, z)) - \frac{1}{\epsilon(t)} J_{gl}(c(t, z)), \tag{2.10c}$$

where $R_i(\cdot)$ for $i \in \{l, d, g\}$ are the productions for the components in the liquid phase, gaseous components dissolved in the liquid phase, and gaseous components in the gas phase, respectively, ϵ is the gas phase volume fraction, and $J_{gl}(\cdot)$ are the gas-liquid mass transfer rates. The gas phase volume fraction is

$$\epsilon(t) = \frac{F_{t,g}(t)}{F_t(t)}. \tag{2.11}$$

The gas-liquid mass transfer rates are

$$J_{gl}(c(t, z)) = k_L a \left(c_{\text{Sat}}(t, z) - c_d(t, z) \right), \tag{2.12}$$

where $c_{\text{Sat}}(\cdot)$ are the saturation concentrations for the gaseous components dissolved in the liquid phase and $k_L a$ is a diagonal matrix of mass transfer coefficients. The saturation concentrations are

$$c_{\text{Sat}}(t, z) = \gamma c_g(t, z), \tag{2.13}$$

where γ is a diagonal matrix of saturation factors. The saturation factors are computed with Henry's law and the ideal gas law, as

$$\gamma = RT \left(H^{cp} \right)^{-1}, \tag{2.14}$$

where R is the ideal gas constant, T is the temperature, and H^{cp} is a diagonal matrix of Henry's constants for the gaseous components. Values of Henry's constants are presented in [80, 81].

Finite-volume discretisation

We formulate the PFR as a system of ODEs by applying a finite-volume discretisation of the PDE. The finite-volume discretisation is described in the technical report listed in Appendix G. Consider the system of ODEs arising from a finite-volume discretisation with N_z discrete volumes

$$\frac{d\bar{c}}{dt}(t) = -D^{(z)}_{N_z, n_c} \bar{N}(t, \bar{c}(t)) + \bar{Q}(t, \bar{c}(t)), \qquad \bar{c}(t_0) = \bar{c}_0, \tag{2.15}$$

where $\bar{c}$ are the concentrations in each discrete volume, $\bar{N}$ are the fluxes through each cross-section separating the discrete volumes, $\bar{Q}$ are the generations for each discrete volume, and $D^{(z)}_{N_z,n_c}$ is a forward difference matrix, i.e. approximates the spatial derivative of the fluxes with a forward-difference approximation. The concentrations, fluxes, and generations are defined for all volumes in the discretisation, as

$$\bar{c} = \begin{bmatrix} \bar{c}_1 \\ \bar{c}_2 \\ \vdots \\ \bar{c}_{N_z} \end{bmatrix}, \qquad \bar{N} = \begin{bmatrix} \bar{N}_{1/2} \\ \bar{N}_{1+1/2} \\ \vdots \\ \bar{N}_{N_z+1/2} \end{bmatrix}, \qquad \bar{Q} = \begin{bmatrix} \bar{Q}_1 \\ \bar{Q}_2 \\ \vdots \\ \bar{Q}_{N_z} \end{bmatrix}. \tag{2.16}$$

The fluxes are

$$\bar{N}(t, \bar{c}(t)) = \begin{bmatrix} \bar{N}_{1/2} \\ \bar{N}_{1:N_z-1} \\ \bar{N}_{N_z+1/2} \end{bmatrix} = \begin{bmatrix} v(t)c_{\text{In}}(t) \\ v(t)\bar{c}_{1:N_z-1}(t) - \bar{D}D^{(z)}_{N_z-1\times N_z,n_c}\bar{c}(t) \\ v(t)\bar{c}_{N_z}(t) \end{bmatrix}, \tag{2.17}$$

where $\bar{N}_{1:N_z-1}$ are the fluxes, $\bar{N}_{i+1/2}$, for the volumes $i \in \{1, 2, \ldots, N_z - 1\}$, $\bar{c}_{1:N_z-1}$ are the concentrations, $\bar{c}_i$, for the volumes $i \in \{1, 2, \ldots, N_z - 1\}$, and $\bar{D}$ is a diagonal matrix of dispersion coefficients. The forward-difference matrix is defined as

$$D^{(z)}_{M,n} = D^{(z)}_{M} \otimes I_n, \qquad D^{(z)}_{M} = \begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{bmatrix}, \tag{2.18}$$

where $I_n \in \mathbb{R}^{n\times n}$ is an identity matrix and $\otimes$ is the Kronecker product. Note that the size of the forward-difference matrix $D^{(z)}_{M} \in \mathbb{R}^{M\times M+1}$ is not quadratic. The dispersion matrix is

$$\bar{D} = D \otimes I_{N_z-1}, \tag{2.19}$$

where D is the diagonal matrix of dispersion coefficients for each component. We define the generation rates in each volume as

$$\bar{Q}_i(t, \bar{c}(t)) = \begin{bmatrix} \bar{Q}_{l,i}(t, \bar{c}(t)) \\ \bar{Q}_{d,i}(t, \bar{c}(t)) \\ \bar{Q}_{g,i}(t, \bar{c}(t)) \end{bmatrix}, \qquad i \in \{1, 2, \ldots, N_z\}, \tag{2.20}$$

where $\bar{Q}_{l,i}$ are the generations for the components only present in the liquid phase, $\bar{Q}_{d,i}$ are the generations for the gaseous components dissolved in the liquid phase, and $\bar{Q}_{g,i}$ are the generations for the gaseous components in the gas phase. We define each of those generations as

$$\bar{Q}_{l,i}(t, \bar{c}(t)) = \bar{R}_{l,i}(\bar{c}(t)), \tag{2.21a}$$

$$\bar{Q}_{d,i}(t, \bar{c}(t)) = \bar{R}_{d,i}(\bar{c}(t)) + \frac{1}{1-\epsilon(t)} \bar{J}_{gl,i}(\bar{c}(t)), \tag{2.21b}$$

$$\bar{Q}_{g,i}(t, \bar{c}(t)) = \bar{R}_{g,i}(\bar{c}(t)) - \frac{1}{\epsilon(t)} \bar{J}_{gl,i}(\bar{c}(t)), \tag{2.21c}$$

where $\bar{R}_i$ are the production rates in volume i and $\bar{J}_{gl,i}$ are the gas-liquid mass transfer rates in volume i. The productions in volume $i \in \{1, 2, \dots, N_z\}$ are defined by the modelling framework for reactive systems in Appendix G, as

$$\bar{R}_i(t, \bar{c}(t)) = \begin{bmatrix} \bar{R}_{l,i}(t, \bar{c}(t)) \\ \bar{R}_{d,i}(t, \bar{c}(t)) \\ \bar{R}_{g,i}(t, \bar{c}(t)) \end{bmatrix} = S^T r(\bar{c}_i(t)), \tag{2.22}$$

where $S \in \mathbb{R}^{n_r \times n_c}$ is the stoichiometric matrix and $r(c) : \mathbb{R}^{n_c} \longrightarrow \mathbb{R}^{n_r}$ is a vector of reaction rates. The gas-liquid mass transfer rates in volume $i \in \{1, 2, \dots, N_z\}$ are define by (2.12), as

$$\bar{J}_{gl,i}(\bar{c}(t)) = J_{gl}(\bar{c}_i(t)). \tag{2.23}$$

2.2.3 U-loop model

The U-loop reactor is a compartment model consisting of a CSTR, as presented in section 2.2.1, representing the top tank, a PFR, as presented in section 2.2.2, representing the U-loop leg, an inlet mixer, and a gas-liquid separator. Figure 2.4 illustrates the U-loop compartment model. In this work, we consider the U-loop reactor model as a system of ODEs in the form

$$\frac{dx}{dt}(t) = f(t, x(t), u(t), \theta), \qquad x(t_0) = x_0. \tag{2.24}$$

The following is an overview of the variables and compartment dynamics.

Variables

The states of the U-loop reactor are comprised of the volume and mole numbers in the top tank CSTR and the concentrations of each volume in the finite-volume discretisation of the U-loop leg PFR

$$x(t) = \begin{bmatrix} x_{CSTR}(t) \\ x_{PFR}(t) \end{bmatrix}, \tag{2.25}$$

where

$$x_{CSTR}(t) = n(t), \qquad x_{PFR}(t) = \bar{c}(t). \tag{2.26}$$

We note that the volume state of the CSTR is eliminated by the in- and outlet flows being equal in the model considered in this work. The technical report listed in Appendix G describes the derivation of the constant volume CSTR. The inputs for the U-loop reactor are the liquid as gas inlet flow rates

$$u(t) = \begin{bmatrix} F_l(t) \\ F_g(t) \end{bmatrix}, \qquad C_{\text{In}} = \begin{bmatrix} C_{l,\text{In}} & C_{g,\text{In}} \end{bmatrix}. \tag{2.27}$$

The input to the CSTR model are

$$u_{CSTR}(t) = \begin{bmatrix} F_{CSTR,\text{In}}(t) \\ F_{CSTR,\text{Out}}(t) \end{bmatrix}, \qquad c_{CSTR,\text{In}}(t) = \begin{bmatrix} \bar{c}_{l,N_z}(t) \\ \bar{c}_{d,N_z}(t) \end{bmatrix}, \tag{2.28}$$

where the inlet concentrations are the liquid outlet from the PFR exiting the gas-liquid separator and the in- and outlet flows are

$$F_{CSTR,\text{In}}(t) = F_{t,l}(t) = e_l^T F_l(t) + F_r, \tag{2.29a}$$
$$F_{CSTR,\text{Out}}(t) = F_h(t) + F_r, \tag{2.29b}$$

where the recycle flow rate, F_r, is constant and the harvest flow rate, $F_h = e_l^T F_l$, is the sum of the liquid inlet flow rates, defining a constant volume CSTR as the top tank. We note that the top tank is not required to have constant volume, but that in this particular work we treat it as such. The elimination of the volume state for the constant volume CSTR can be found in the technical report listed in Appendix G. The inputs to the PFR are the inflows from the inlet mixer

$$u_{PFR}(t) = F_t(t), \qquad c_{\text{In}}(t), \tag{2.30}$$

where F_t is the total flow rate of liquids and gasses, and c_{In} are the total concentrations of liquids and gasses exiting the mixer.

Reactor dynamics

The right-hand side function is

$$f(t, x(t), u(t), \theta) = \begin{bmatrix} f_{CSTR}(t, x_{CSTR}(t), u_{CSTR}(t), \theta) \\ f_{PFR}(t, x_{PFR}(t), u_{PFR}(t), \theta) \end{bmatrix}, \tag{2.31}$$

where f_{CSTR} are the CSTR dynamics defined in section 2.2.1 and f_{PFR} are the dynamics of the finite-volume discretisation of the PFR described in section 2.2.2. The top tank dynamics are

$$\begin{aligned} &f_{CSTR}(t, x_{CSTR}(t), u_{CSTR}(t), \theta) = \\ &\quad (c_{CSTR,\text{In}}(t) - c(t))\, F_{t,l}(t) + R(c(t))V, \end{aligned} \qquad n(t_0) = n_0. \tag{2.32}$$

The U-loop leg dynamics are

$$\begin{aligned} f_{PFR}(t, x_{PFR}(t), u_{PFR}, \theta) = \\ - D^{(z)}_{N_z, n_c} \bar{N}(t, \bar{c}(t)) + \bar{Q}(t, \bar{c}(t)), \end{aligned} \qquad \bar{c}(t_0) = \bar{c}_0. \tag{2.33}$$

In the mixer, the liquid inlet flows are mixed with the liquid recycle flow from the top tank. As such, the liquid inlet mix is

$$c_{\text{In},l}(t) = C_{\text{In},l} \frac{F_l(t)}{F_{t,l}(t)} + c \frac{F_r}{F_{t,l}(t)}, \qquad F_{t,l}(t) = e_l^T F_l(t) + F_r. \tag{2.34}$$

As there is no modelled gas phase in the top tank, the gas mix is defined entirely by the gaseous inlet flows. As such, the gaseous inlet mix is

$$c_{\text{In},g}(t) = C_{\text{In},g} \frac{F_g(t)}{F_{t,g}(t)}, \qquad F_{t,g}(t) = e_g^T F_g(t). \tag{2.35}$$

Given the liquid and gas inlet flows, the full gas-liquid mix may be defined as

$$c_{\text{In}}(t) = c_{\text{In},l}(t) \frac{F_{t,l}(t)}{F_t(t)} + c_{\text{In},g}(t) \frac{F_{t,g}(t)}{F_t(t)}, \qquad F_t(t) = F_{t,l}(t) + F_{t,g}(t). \tag{2.36}$$

In the separator, we assume that the gas-liquid mixture is perfectly and instantaneously separated into the gas and liquid phases. The liquid phase flows into the top tank, defining the inlet concentrations for the CSTR, and the gas phase exits through the off-gas stream.

2.3 Summary

In this chapter, we presented descriptions and a model for the U-loop bioreactor. We described the U-loop fermenters operated by Unibio A/S at their facility in Kalundborg, Denmark. We described the concept and design of the U-loop reactor and presented schematics describing the actuator and sensor layouts for the pilot- and demonstration-scale reactors operated by Unibio A/S. We presented a CSTR and PFR, describing the major compartments of the U-loop reactor. We applied a finite-volume discretisation to the system of PDEs arising from the PFR dynamics, and formulated a system of ODEs describing the discretised reactor. Finally, we presented the model of the U-loop bioreactor based on the four major compartments; the top tank (CSTR), the U-loop leg (PFR), the inlet mixer, and the gas-liquid separator.

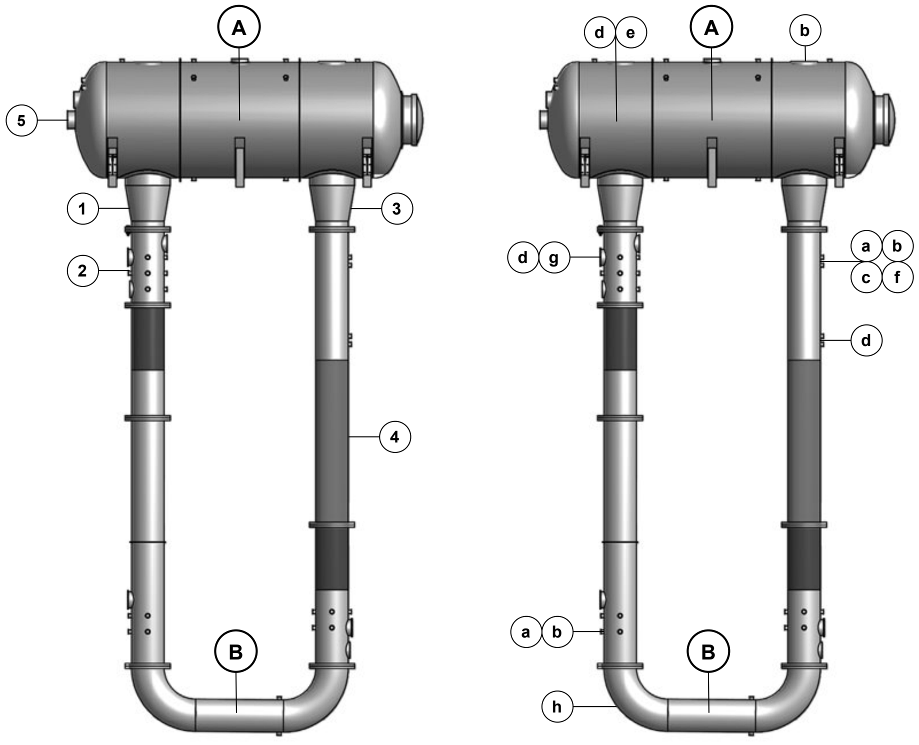


Figure 2.2: Illustration of the pilot-scale reactor layout, A) the degassing unit (top tank) and B) the U-loop leg. **Left:** Actuator layout of the plant (MVs), 1) recirculation pump, 2) inlet stream of gas and liquid substrates, acid/base, water (fresh or recirculated), trace-metals, minerals, etc., 3) pressure valve controlling leg pressure, 4) cooling jacket, and 5) harvest outlet stream. **Right:** Sensor layout of the plant measurements, a) pH-value, b) temperature (top tank temperature sensor is in the gas outlet), c) dissolved oxygen (DO), d) pressure, e) top tank liquid level, f) U-loop leg flow rate, g) K, NO_3, and NH_4^+ probe, and h) sampling valve for offline measurements. For the offline measurements, samples are taken every four hours for measurements of; pH, kH, conductivity, and DCW (dry cell weight). Additionally, samples are taken every hour for measurements of; NO_2^-, NO_3^-, and NH_4^+.

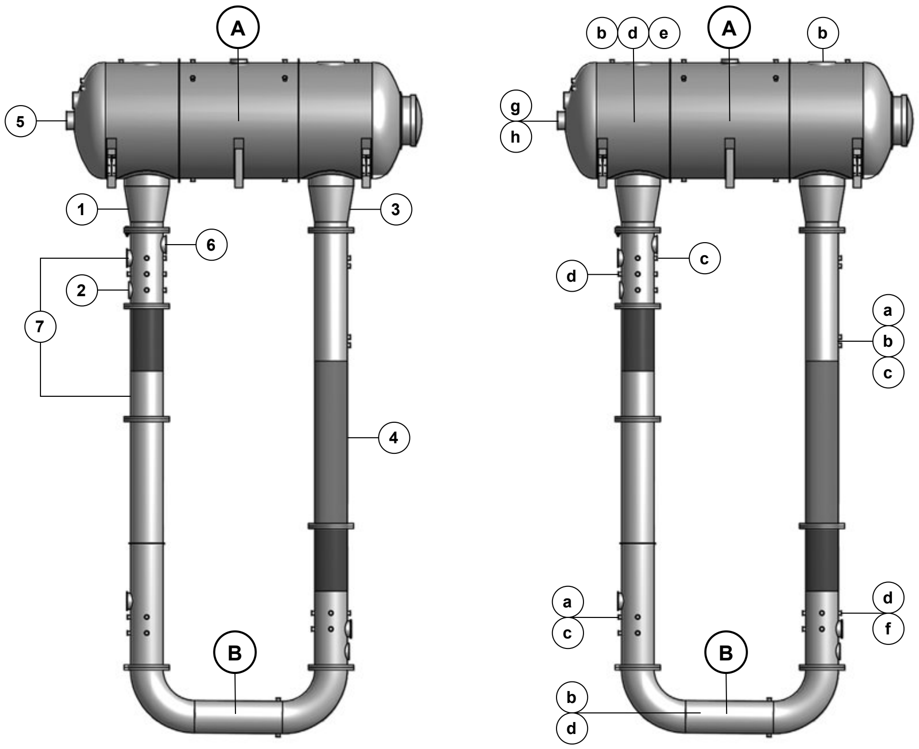


Figure 2.3: Illustration of the demonstration-scale reactor layout, A) the degassing unit (top tank) and B) the U-loop leg. **Left:** Actuator layout of the plant (MVs), 1) recirculation pump, 2) inlet stream of gas and liquid substrates, acid/base, trace-metals, minerals, etc., 3) pressure valve controlling leg pressure, 4) cooling jacket (currently not used in the demonstration-scale reactor), 5) harvest outlet stream, 6) water inlet stream (fresh or recirculated), 7) and active external cooling loop with heat exchanger. **Right:** Sensor layout of the plant measurements, a) pH-value, b) temperature (top tank temperature sensor is in the gas outlet), c) dissolved oxygen (DO), d) pressure, e) top tank liquid level, f) U-loop leg flow rate, g) K, NO_3, and NH_4^+ probe, h) sampling valve for offline measurements. For the offline measurements, samples are taken every 4 hours for measurements of; pH, kH, conductivity, and DCW (dry cell weight). Additionally, samples are taken every hour for measurements of; NO_2^-, NO_3^-, and NH_4^+.

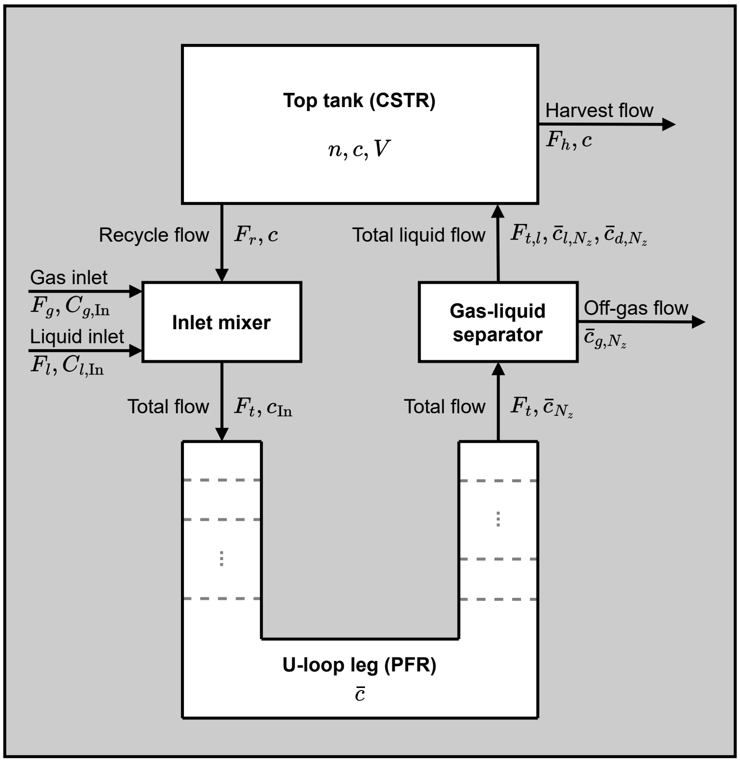


Figure 2.4: Illustration of the compartment model of the U-loop bioreactor. **States:** The top tank states are the mole numbers n (or concentrations c). The U-loop leg states are the concentrations in each volume of the finite-volume discretisation $\bar{c}$. **Inputs:** The inlet are the liquid inlet flows F_l and concentrations $C_{l,\text{In}}$ and the gas inlet flows F_g and concentrations $C_{g,\text{In}}$. F_r is the recycle flow, F_h is the harvest flow, $F_{t,l}$ and $F_{t,g}$ are the total liquid and gas flows, respectively, and F_t is the total flow.

CHAPTER 3

Growth Models

The work presented in this chapter is partially based on the technical report listed in Appendix G on modelling of reactive systems. Additionally, the work presented in this chapter is partially based on the publications listed in Appendices D, E, and F on control for a U-loop bioreactor and paper listed in Appendix A on modelling, simulation, and economic optimal control for a laboratory-scale bioreactor. In this chapter, we present kinetic models describing the growth of *Methylococcus capsulatus* for SCP production. We present models based on methanol and methane as carbon sources, and several different nitrogen sources; nitric acid, ammonia, ammonium, nitrite, nitrate, and molecular nitrogen. For each of the models, we describe stoichiometry, define the stoichiometric matrix, and describe growth kinetics to formulate the rates of reaction. We compactly describe the production rates for each component as

$$R(c) = S^T r(c), \tag{3.1}$$

where $R(c) : \mathbb{R}^{n_c} \longrightarrow \mathbb{R}^{n_c}$ are the production rate, $S \in \mathbb{R}^{n_r \times n_c}$ is the stoichiometric matrix, and $r(c) : \mathbb{R}^{n_c} \longrightarrow \mathbb{R}^{n_r}$ is a vector of the reaction rates.

The chapter is structured as follows: in section 3.1, we present a model based on methanol and nitric acid as substrates. section 3.2 presents a model based on methane and ammonia as substrates. In section 3.3, we present a model based on a metabolic study of the micro-organism. This model is based on methane as substrate and includes the anabolism of several sources of nitrogen; ammonium, nitrite, nitrate, and molecular nitrogen. Section 3.4 describes considerations and developments to the presented in section 3.3. In section 3.5, we present a reduced model based on the model presented in section 3.3. We apply the model in the paper listed in Appendix A. Finally, we present a summary of the chapter in section 3.6.

Notes on notation

We apply the modelling framework for reactive systems listed in Appendix G for the description of growth models for cultivation of *M. captulatus.* As such, we define each reactive system in terms of their respective stoichiometric matrix and rate of reaction vector. We describe the modelled components of each model as elements in the set of modelled components

$$\mathcal{C} = \mathcal{C}_l \cup \mathcal{C}_g, \tag{3.2}$$

where $\mathcal{C}_l$ is the set of components only present in the liquid phase and $\mathcal{C}_g$ is the set of components present in the gas phase and as gasses dissolved in the liquid phase. The vector of concentrations is

$$c = \begin{bmatrix} c_l \\ c_d \\ c_g \end{bmatrix}, \tag{3.3}$$

where c_l is a vector of components, $c_i = [i]$ for $i \in \mathcal{C}_l$, only present in the liquid, c_d is a vector of gaseous components dissolved in the liquid phase, $c_i = [i]$ for $i \in \mathcal{C}_g$, and c_g is a vector of gaseous components in the gas phase, $c_{i,g} = [i]$ for $i \in \mathcal{C}_g$.

3.1 Methanol model

In this section, we present a growth model with methanol and nitric acid as sources of carbon and nitrogen, respectively. The stoichiometry and kinetics of the model is presented by [31]. The growth model is described and analysed in [33] and optimal operating points are investigated in [37]. Economic optimal operation is investigated for the U-loop reactor in [36] and real-time economic optimal control for growth in a CSTR is investigated and presented in [82]. We describe the growth of *M. capsulatus* on methanol and nitric acid in the presence of oxygen, i.e. aerobic conditions, by the stoichiometry

$$\begin{aligned} &1.366\,\mathrm{CH_3OH} + 0.199\,\mathrm{HNO_3} + 0.600\,\mathrm{O_2} \longrightarrow \\ &\quad \mathrm{X} + 0.366\,\mathrm{CO_2} + 1.933\,\mathrm{H_2O}, \end{aligned} \qquad r_1. \tag{3.4}$$

Stoichiometry

In this model, we consider the liquid and gaseous components

$$\mathcal{C}_l = \{X, S\}, \qquad \mathcal{C}_g = \{O\}, \tag{3.5}$$

where

$$X = \mathrm{CH_{1.8}O_{0.5}N_{0.2}}, \qquad S = \mathrm{CH_3OH}, \qquad O = \mathrm{O_2}. \tag{3.6}$$

The resulting modelled reactive system is described by the stoichiometry

$$1.366S + 0.600O \longrightarrow X, \qquad r_1(c). \tag{3.7}$$

We describe the stoichiometry defined in (3.7) by the stoichiometric matrix

$$S = \begin{array}{cccc|c} \mathrm{X} & \mathrm{S} & \mathrm{O} & \mathrm{O}_g & \\ \hline 1 & -1.366 & -0.600 & 0 & \mathrm{r}_1 \end{array}. \tag{3.8}$$

Reaction rate

The vector of reaction rates for each reaction of the growth model is

$$r(c) = [r_1(c)]. \tag{3.9}$$

The reaction rate for reaction one, r_1, is

$$r_1(c) = \mu(c)c_X. \tag{3.10}$$

The specific growth rate is

$$\mu(c) = \mu_{\max}\mu_S(c)\mu_O(c), \tag{3.11}$$

where $\mu_{\max}$ is the maximum specific growth rate, μ_S is the specific growth rate on substrate, and μ_O is the specific growth rate on oxygen. The substrate, methanol, inhibits growth at high concentrations. Therefore, we apply Haldane kinetics to describe the specific growth rate on substrate, as

$$\mu_S(c) = \frac{c_S}{K_S + c_S + c_S^2/K_{S,I}}, \tag{3.12}$$

where K_S is the saturation constant and $K_{S,I}$ is the substrate inhibition constant. We apply Monod kinetics to describe the specific growth rate on oxygen, as

$$\mu_O(c) = \frac{c_O}{K_O + c_O}, \tag{3.13}$$

where K_O is the saturation constant. Table 3.1 presents the kinetic parameters for the model.

Optimal substrate concentration

The Haldane expression gives rise to a maximum growth rate and thus also an optimal concentration of substrate for growth, c_S^* [83]. To determine c_S^*, we describe the optimal growth as the constrained optimisation problem

$$\min_{c_S} \quad \phi = -\mu_S(c_S), \tag{3.14}$$

subject to

$$c_S \geq 0. \tag{3.15}$$

The solution to the problem arises as the solution to the 1st order necessary optimality conditions, i.e. Karush-Kuhn-Tucker (KKT) conditions, as

$$0 = \frac{d\phi}{dc_S}(c_S^*). \tag{3.16}$$

We solve (3.16) for the optimal concentration, c_S^*, and compute the optimal concentration satisfying the non-negativity constraint, as

$$c_S^* = \sqrt{K_{S,I} K_S}. \tag{3.17}$$

We apply this in control applications listed in Appendices D and F to stabilise the system. Figure 3.1 illustrates the Haldane growth kinetics and optimal growth conditions.

3.2 Methane model

In this section, we present a growth model with methane as carbon source and ammonia as nitrogen source. The stoichiometry and kinetics of the model is presented in [84]. The growth of *M. capsulatus* on methane and ammonia in the presence of oxygen, i.e. aerobic conditions, is described by the stoichiometry

$$\begin{gathered} 1.9186\,\mathrm{CH_4} + 0.1999\,\mathrm{NH_3} + 2.7874\,\mathrm{O_2} \longrightarrow \\ \mathrm{X} + 0.9186\,\mathrm{CO_2} + 1.6351\,\mathrm{H_2O}, \end{gathered} \qquad r_1. \tag{3.18}$$

Stoichiometry

In this model, we consider the liquid and gaseous components

$$\mathcal{C}_l = \{X, N\}, \qquad \mathcal{C}_g = \{S, O, C\}, \tag{3.19}$$

where

$$X = \mathrm{CH_{1.8}O_{0.5}N_{0.2}}, \quad N = \mathrm{NH_3}, \quad S = \mathrm{CH_4}, \quad O = \mathrm{O_2}, \quad C = \mathrm{CO_2}. \tag{3.20}$$

The resulting modelled reactive systems is described by the stoichiometry

$$S + 0.1042N + 1.4528O \longrightarrow 0.5212X + 0.4788C, \qquad r_1(c). \tag{3.21}$$

We describe the stoichiometry defined in (3.21) by the stoichiometric matrix

$$S = \begin{array}{c} \begin{array}{cccccccc} \mathrm{X} & \mathrm{N} & \mathrm{S} & \mathrm{O} & \mathrm{C} & \mathrm{S}_g & \mathrm{O}_g & \mathrm{C}_g \end{array} \\ \left[\begin{array}{cccccccc} 0.5212 & -0.1042 & -1 & -1.4528 & 0.4788 & 0 & 0 & 0 \end{array}\right] \mathrm{r}_1 \end{array}. \tag{3.22}$$

Table 3.1: Kinetic parameters for the methanol model [33].

Symbol	**Value**	**Unit**
$\mu_{\max}$	$3.7 \cdot 10^{-1}$	1/h
K_S	$6.6 \cdot 10^{-4}$	mol/L
$K_{S,I}$	$1.2 \cdot 10^{-2}$	mol/L
K_O	$2.0 \cdot 10^{-6}$	mol/L

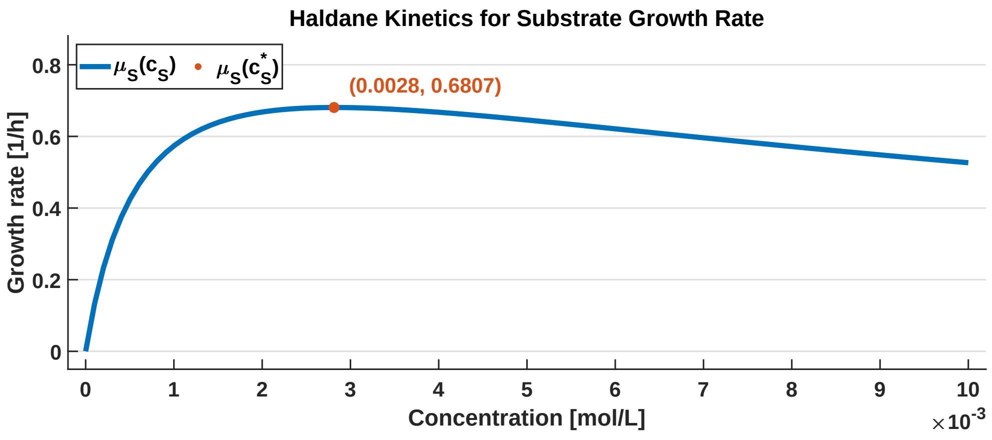


Figure 3.1: Haldane kinetics with optimal growth rate for growth on methanol for single-cell protein production.

Reaction rate

The vector of reaction rates for each reaction of the growth model is

$$r(c) = \left[r_1(c) \right]. \tag{3.23}$$

The reaction rate for reaction one, r_1, is

$$r_1(c) = \mu(c) c_X. \tag{3.24}$$

The specific growth rate is

$$\mu(c) = \mu_{\max} \mu_S(c) \mu_O(c), \tag{3.25}$$

where $\mu_{\max}$ is the maximum growth rate, μ_S is the specific growth rate on substrate, and μ_O is the specific growth rate on oxygen. We apply Monod kinetics to describe the specific growth rate on substrate, as

$$\mu_S(c) = \frac{c_S}{K_S + c_S}, \tag{3.26}$$

where K_S is the saturation constant. We apply Monod kinetics to describe the specific growth rate on oxygen, as

$$\mu_O(c) = \frac{c_O}{K_O + c_O}, \tag{3.27}$$

where K_O is the saturation constant. Table 3.2 presents the kinetic parameters for the model.

Table 3.2: Kinetic parameters for the methane model [84].

Symbol	**Value**	**Unit**
$\mu_{\max}$	$3.7 \cdot 10^{-1}$	1/h
K_S	$1.3 \cdot 10^{-6}$	mol/L
K_O	$3.0 \cdot 10^{-7}$	mol/L

3.3 Metabolic model

In this section, we present a growth model with methane as carbon source and ammonium, nitrite, nitrate, and molecular nitrogen as nitrogen sources. The growth model is based on the metabolic study of *M. capsulatus* presented in [28]. The model includes anabolic, catabolic, and co-metabolic reactions in the micro-organism. Co-metabolic processes are investigated and described in [34]. Modelling and system identification for the growth model applied in a U-loop reactor for SCP production is described in [35]. The model is described in detail in [29]. The growth of *M. capsulatus* is described by the catabolic reaction

$$\mathrm{CH_4} + \mathrm{O_2} \longrightarrow \mathrm{CO_2} + 4\,\mathrm{H^+} + 4\,\mathrm{e^-}, \qquad r_1, \tag{3.28}$$

the anabolism of nitrogen sources

$$\begin{aligned}&\mathrm{CH_4} + \mathrm{O_2} + \frac{2}{10}\,\mathrm{NH_4}^+ + \frac{2}{10}\,\mathrm{e^-} + \mathrm{ATP} \longrightarrow \\ &\quad \mathrm{CH_{1.8}O_{0.5}N_{0.2}} + \mathrm{ADP} + \frac{15}{10}\,\mathrm{H_2\,O},\end{aligned} \qquad r_2, \tag{3.29a}$$

$$\begin{aligned}&\mathrm{CH_4} + \mathrm{O_2} + \frac{2}{10}\,\mathrm{NO_2}^- + \frac{14}{10}\,\mathrm{e^-} + \mathrm{ATP} + \frac{16}{10}\,\mathrm{H^+} \longrightarrow \\ &\quad \mathrm{CH_{1.8}O_{0.5}N_{0.2}} + \mathrm{ADP} + \frac{19}{10}\,\mathrm{H_2O},\end{aligned} \qquad r_3, \tag{3.29b}$$

$$\begin{aligned}&\mathrm{CH_4} + \mathrm{O_2} + \frac{2}{10}\,\mathrm{NO_3}^- + \frac{18}{10}\,\mathrm{e^-} + \mathrm{ATP} + \frac{20}{10}\,\mathrm{H^+} \longrightarrow \\ &\quad \mathrm{CH_{1.8}O_{0.5}N_{0.2}} + \mathrm{ADP} + \frac{21}{10}\,\mathrm{H_2O},\end{aligned} \qquad r_4, \tag{3.29c}$$

$$\begin{aligned}&\mathrm{CH_4} + \mathrm{O_2} + \frac{1}{10}\,\mathrm{N_2} + \frac{8}{10}\,\mathrm{e^-} + \left(\alpha + \frac{16}{10}\right)\mathrm{ATP} + \frac{8}{10}\,\mathrm{H^+} \longrightarrow \\ &\quad \mathrm{CH_{1.8}O_{0.5}N_{0.2}} + \left(\alpha + \frac{16}{10}\right)\mathrm{ADP} + \frac{15}{10}\,\mathrm{H_2O},\end{aligned} \qquad r_5, \tag{3.29d}$$

and the co-metabolic reactions

$$\mathrm{NH_4}^+ + \mathrm{O_2} \longrightarrow \mathrm{NO_2}^- + 4\,\mathrm{H^+} + 2\,\mathrm{e^-}, \qquad r_6, \tag{3.30a}$$

$$\mathrm{NO_2}^- + \frac{1}{2}\,\mathrm{O_2} \longrightarrow \mathrm{NO_3}^-, \qquad r_7. \tag{3.30b}$$

Stoichiometry

In this model, we consider the liquid and gaseous components

$$\mathcal{C}_l = \{X, NH, NO2, NO3\}, \qquad \mathcal{C}_g = \{N, S, O, C\}, \tag{3.31}$$

where

$$X = \mathrm{CH_{1.8}O_{0.5}N_{0.2}}, \qquad NH = \mathrm{NH_4}^+, \tag{3.32a}$$
$$NO2 = \mathrm{NO_2}^-, \qquad NO3 = \mathrm{NO_3}^-, \tag{3.32b}$$
$$N = \mathrm{N_2}, \qquad S = \mathrm{CH_4}, \tag{3.32c}$$
$$O = \mathrm{O_2}, \qquad C = \mathrm{CO_2}. \tag{3.32d}$$

The resulting modelled reactive system is described by the stoichiometry

$$S + O \longrightarrow C, \qquad r_1(c), \tag{3.33a}$$
$$S + O + \frac{2}{10}NH \longrightarrow X, \qquad r_2(c), \tag{3.33b}$$
$$S + O + \frac{2}{10}NO2 \longrightarrow X, \qquad r_3(c), \tag{3.33c}$$
$$S + O + \frac{2}{10}NO3 \longrightarrow X, \qquad r_4(c), \tag{3.33d}$$
$$S + O + \frac{1}{10}N \longrightarrow X, \qquad r_5(c), \tag{3.33e}$$
$$NH + O \longrightarrow NO2, \qquad r_6(c), \tag{3.33f}$$
$$NO2 + \frac{1}{2}O \longrightarrow NO3, \qquad r_7(c). \tag{3.33g}$$

We describe the stoichiometry in (3.33) by the stoichiometric matrix

$$S = \begin{array}{c} \begin{array}{cccccccccccc} \mathrm{X} & \mathrm{NH} & \mathrm{NO2} & \mathrm{NO3} & \mathrm{N} & \mathrm{S} & \mathrm{O} & \mathrm{C} & \mathrm{N}_g & \mathrm{S}_g & \mathrm{O}_g & \mathrm{C}_g \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1 & 0 & 0 & 0 & 0 \\ 1 & -\frac{2}{10} & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -\frac{2}{10} & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & -\frac{2}{10} & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & -\frac{1}{10} & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{array}{c} \mathrm{r}_1 \\ \mathrm{r}_2 \\ \mathrm{r}_3 \\ \mathrm{r}_4 \\ \mathrm{r}_5 \\ \mathrm{r}_6 \\ \mathrm{r}_7 \end{array} \end{array} . \tag{3.34}$$

Reaction rate

The vector of reaction rates of the growth model is

$$r(c) = \begin{bmatrix} r_1(c) \\ r_2(c) \\ r_3(c) \\ r_4(c) \\ r_5(c) \\ r_6(c) \\ r_7(c) \end{bmatrix}. \tag{3.35}$$

The reaction rates for $i \in \{1, 2, \ldots, 7\}$ are

$$r_i(c) = \mu_i(c) c_X. \tag{3.36}$$

The specific growth rate for reaction one is

$$\begin{aligned} \mu_1(c) = & \left(\frac{\alpha}{2\delta} + \frac{8}{20}\right) \mu_2(c) + \left(\frac{\alpha}{2\delta} - \frac{1}{2}\right) \mu_3(c) \\ & + \left(\frac{\alpha}{2\delta} + \frac{9}{20\delta}\right) \mu_4(c) + \left(\frac{8\alpha + 5}{16\delta} + \frac{1}{5}\right) \mu_5(c) + \frac{m}{2\delta}, \end{aligned} \tag{3.37}$$

where α is a stoichiometry coefficient describing the ATP demand in biomass forming reactions, δ is a stoichiometric coefficient describing ATP yield from the electron transport chain, and m is the maintenance rate. There exists a strong correlation between α and δ [29]. The specific growth rate for reaction two is

$$\mu_2(c) = \mu_{\max} \mathrm{amin}_{10} \left(\mu_S(c), \mu_O(c), \mu_{NH}(c)\right), \tag{3.38}$$

where $\mu_{\max}$ is the maximum growth rate, μ_S is the specific growth rate on substrate, μ_O is the specific growth rate on oxygen, and μ_{NH} is the specific growth rate on ammonium. This is a smooth approximation to the expression for growth on multiple substrates described in [19], as the minimum of the growth rates

$$\mu(c) = \mu_{\max} \min\{\mu_1(c), \mu_2(c), \ldots\}. \tag{3.39}$$

The smooth minimum approximation is

$$\mathrm{amin}_N \left(\mu_a(c), \mu_a(c), \ldots\right) = \frac{w_a(c)\mu_a(c) + w_b(c)\mu_b(c) + \cdots}{w_a(c) + w_b(c) + \cdots}, \tag{3.40}$$

where the weights are described by the Nth order polynomial $w_i(c) = (1 - \mu_i(c))^N$ and N is a positive integer, $N \in \mathbb{N}$. The specific growth rates on substrate competes with the co-metabolic reaction r_6 and is therefore inhibited by the concentration of ammonium. As such, the specific growth rate on substrate is

$$\mu_S(c) = \frac{c_S}{K_S \left(1 + \frac{c_{NH}}{K_{NH,ox}}\right) + c_S}, \tag{3.41}$$

where K_S is the saturation constant for growth on substrate and $K_{NH,ox}$ is the ammonium inhibition constant. The kinetic parameter $K_{NH,ox}$ is related to the oxidation of ammonium to nitrite described by the co-metabolic reaction, r_6. We apply Monod kinetics to describe the specific growth rate on oxygen, as

$$\mu_O(c) = \frac{c_O}{K_O + c_O}, \tag{3.42}$$

where K_O is the saturation constant. We apply Monod kinetics to describe the specific growth rate on ammonia, as

$$\mu_{NH}(c) = \frac{c_{NH}}{K_{NH} + c_{NH}}, \tag{3.43}$$

where K_{NH} is the saturation constant. The specific growth rate for reaction three is

$$\mu_3(c) = \mu_{\max}\mathrm{amin}_{10}\left(\mu_S(c), \mu_O(c), \mu_{NO2}(c)\right)\xi_{NH}(c), \tag{3.44}$$

where μ_{NO2} is the specific growth rate on nitrite and ξ_{NH} describe catabolic repression related to ammonium. We apply Monod kinetics to describe the specific growth rate on nitrite, as

$$\mu_{NO2}(c) = \frac{c_{NO2}}{K_{NO2} + c_{NO2}}, \tag{3.45}$$

where K_{NO2} is the saturation constant. The catabolic repression function for a specific component, ξ_i for $i \in \mathcal{C}$, is modelled by a sigmoid activation function

$$\xi_i(c) = \frac{1}{1 + e^{K_{cr,i} - c_i}}, \tag{3.46}$$

where $K_{cr,i}$ is the activation constant, i.e. the system will be 50% at $c_i = K_{cr,i}$. The specific growth rate for reaction four is

$$\mu_4(c) = \mu_{\max}\mathrm{amin}_{10}\left(\mu_S(c), \mu_O(c), \mu_{NO3}(c)\right)\mathrm{amin}_{100}\left(\xi_{NH}(c), \xi_{NO2}(c)\right), \tag{3.47}$$

where μ_{NO3} is the specific growth rate on nitrate and ξ_{NO2} is catabolic repression related to nitrite. We apply Monod kinetics to describe the specific growth rate on nitrate, as

$$\mu_{NO3}(c) = \frac{c_{NO3}}{K_{NO3} + c_{NO3}}, \tag{3.48}$$

where K_{NO3} is the saturation constant. The specific growth rate for reaction five is

$$\begin{aligned} \mu_5(c) = \mu_{\max,N}&\mathrm{amin}_{10}\left(\mu_S(c), \mu_O(c), \mu_N(c)\right) \\ &\mathrm{amin}_{100}\left(\xi_{NH}(c), \xi_{NO2}(c), \xi_{NO3}(c)\right), \end{aligned} \tag{3.49}$$

where $\mu_{\max,N}$ is the maximum growth rate on molecular nitrogen, μ_N is the specific growth rate on molecular nitrogen, and ξ_{NO3} is catabolic repression related to nitrate.

We apply Monod kinetics to describe the specific growth rate on molecular nitrogen, as

$$\mu_N(c) = \frac{c_N}{K_N + c_N}, \tag{3.50}$$

where K_N is the saturation constant. The specific reaction rate for reaction six is

$$\mu_6(c) = V_{NO2}\text{amin}_{10}\left(\mu_{NH,ox}(c), \mu_O(c)\right), \tag{3.51}$$

where V_{NO2} is the maximum specific rate of the non-growth related reaction, r_6. We apply the same inhibition kinetics as for the anabolic reaction r_2 and describe the specific rate of reaction for ammonium oxidation, as

$$\mu_{NH,ox}(c) = \frac{c_{NH}}{K_{NH,ox}\left(1 + \frac{c_S}{K_S}\right) + c_{NH}}, \tag{3.52}$$

where $K_{NH,ox}$ is the saturation constant and K_S is the substrate inhibition constant. The specific reaction rate for reaction seven is

$$\mu_7(c) = V_{NO3}\text{amin}_{10}\left(\mu_{NO2,ox}(c), \mu_{O2}(c)\right), \tag{3.53}$$

where V_{NO3} is the maximum specific rate of the non-growth related reaction, r_7. We apply Monod kinetics to describe the specific rate of reaction of the nitrite co-metabolic reaction, as

$$\mu_{NO2,ox}(c) = \frac{c_{NO2}}{K_{NO2,ox} + c_{NO2}}, \tag{3.54}$$

where $K_{NO2,ox}$ is the saturation constant. Table 3.3 presents the kinetic parameters for the model.

3.4 Development of metabolic model

In this section, we present developments made for the model described in section 3.3. Particularly, we investigate the formulation of the catabolic repression model and propose improvements to better capture the desired dynamics.

3.4.1 Catabolic repression

Consider the catabolic repression function presented in section 3.3 for a component of concentration, c,

$$\xi(c) = \frac{1}{1 + e^{K_c - c}}. \tag{3.55}$$

Table 3.3: Kinetic parameters for the metabolic model [29].

Symbol	Value	Unit
$\mu_{\max}$	$2.3 \cdot 10^{-1}$	1/h
$\mu_{\max,N}$	$4.0 \cdot 10^{-2}$	1/h
V_{NO2}	$3.2 \cdot 10^{-2}$	molc/(molc X · h)
V_{NO3}	$2.3 \cdot 10^{-2}$	molc/(molc X · h)
α	$2.0 \cdot 10^{-2}$	mol ATP/molc X
δ	$2.0 \cdot 10^{-2}$	mol ATP/mol(2e)
m	$9.8 \cdot 10^{-5}$	mol ATP/(molc X · h)
K_S	$7.5 \cdot 10^{-5}$	mol/L
K_O	$5.5 \cdot 10^{-7}$	mol/L
K_{NH}	$1.3 \cdot 10^{-3}$	mol/L
K_{NO2}	$1.3 \cdot 10^{-3}$	mol/L
K_{NO3}	$1.3 \cdot 10^{-3}$	mol/L
K_N	$1.8 \cdot 10^{-3}$	mol/L
$K_{NH,ox}$	$3.3 \cdot 10^{-3}$	mol/L
$K_{NO2,ox}$	$1.3 \cdot 10^{-3}$	mol/L
$K_{cr,NH}$	$2.9 \cdot 10^{5}$	mol/L
$K_{cr,NO2}$	0.0	mol/L
$K_{cr,NO3}$	$2.1 \cdot 10^{-5}$	mol/L

The function presented in (3.55) is a so-called sigmoid activation function. A sigmoid activation function, $\xi : \mathbb{R} \longrightarrow [0, 1]$, maps the input to a value between 0 and 1, where 0 indicates that the system is inactive and 1 indicates that the system is active. The parameter K_c is the activation constant, which describing the point where the system is half active, i.e. $\xi(c) = 0.5$ for $c = K_c$. Sigmoid activation functions are smooth approximations to the instantaneous activation represented by the piece-wise constant function

$$\xi^{(\infty)}(c) = \begin{cases} 0 & \text{for } c \leq K_c \\ 1 & \text{otherwise} \end{cases}. \tag{3.56}$$

The sigmoid activation function described in section 3.3 is illustrated in Figure 3.2. Notably, the activation for the sigmoid function is not instant, but are asymptotically achieved as the variable, c, tends to negative and positive infinity, respectively, i.e. $\xi(c) \longrightarrow 0$ for $c \longrightarrow -\infty$ and likewise $\xi(c) \longrightarrow 1$ for $c \longrightarrow \infty$. We may consider the area within which the sigmoid achieves 95% activation, i.e. the interval between

$\xi(c) = 0.025$ and $\xi(c) = 0.975$. We may compute this interval as

$$\alpha_{\%} = \frac{1}{1+e^{K_c - c_{\%}}} \qquad \Longleftrightarrow \tag{3.57a}$$

$$\alpha_{\%} + \alpha_{\%} e^{K_c - c_{\%}} = 1 \qquad \Longleftrightarrow \tag{3.57b}$$

$$\alpha_{\%} e^{K_c - c_{\%}} = 1 - \alpha_{\%} \qquad \Longleftrightarrow \tag{3.57c}$$

$$e^{K_c - c_{\%}} = \frac{1-\alpha_{\%}}{\alpha_{\%}} \qquad \Longleftrightarrow \tag{3.57d}$$

$$c_{\%} = K_c - \ln\left(\frac{1-\alpha_{\%}}{\alpha_{\%}}\right), \tag{3.57e}$$

where $\alpha_{\%}$ is the activation decimal and $c_{\%}$ is the concentration at which an activation of $\alpha_{\%}$ is achieved. We denote the 95% activation points of the lower 2.5% activation point, $\alpha_{\%,l} = 0.025$, and the upper 97.5% activation point, $\alpha_{\%,u} = 0.975$, the activation characteristics. Looking at a particular example of the activation characteristics for the catabolic repression of nitrite in the presence of ammonium, $\xi_{NH}(c)$, we find that the activation characteristics are

$$\alpha_{\%,l} = -3.6635, \qquad \alpha_{\%,u} = 3.6636, \tag{3.58}$$

for the activation constant $K_{c,NH} = 2.9310 \cdot 10^{-5}$. This indicates, that the catabolic repression of nitrite is almost entirely inactive only when the concentration of ammonium is less that -3.6635 and that it is almost entirely active only when the concentration of ammonium is more than 3.6636. Figure 3.2 illustrates a sigmoid activation function with these activation characteristics and compares it to the instantaneous activation function, $\xi^{(\infty)}(\cdot)$. However, the concentration of ammonium is unlikely to exceed molar concentrations of 1.0 M and will never decrease below 0 M. Evaluating the activation between these two points, 0 and 1 M, respectively, we get

$$\xi_{NH}(0) = 4.9999 \cdot 10^{-1}, \qquad \xi_{NH}(1) = 7.3105 \cdot 10^{-1}, \tag{3.59}$$

amounting to between 50% and 73% activation at all times. The same is true for the remaining catabolic repression functions. The reason for this dynamic, is that [29] applies an unscaled sigmoid activation function. Such an activation function always has the same activation interval, but can shift its activation point by changing the activation constant, K_c. To better represent the catabolic repression dynamic, we introduce the scaled sigmoid activation function

$$\xi^{(\beta)} = \frac{1}{1+e^{\beta(K_c - c)}}, \tag{3.60}$$

where β is a scaling constant used to increase of decrease the activation interval. For the scaled sigmoid activation function, we can choose the activation characteristics of the function by choosing the value of β. To determine the value of β for an activation of 95% around K_c of $\pm\alpha_{\%}$, we derive the expression for the lower activation limit, $\alpha_{\%,l}$, as

$$\begin{aligned}
\xi^{(\beta)}(K_C - \alpha_{\%}) &= 0.025 && && (3.61a)\\
&= \frac{1}{1+e^{\beta(K_c-K_c+\alpha_{\%})}} && \Longleftrightarrow && (3.61b)\\
0.025 + 0.025e^{\beta(\alpha_{\%})} &= 1 && \Longleftrightarrow && (3.61c)\\
e^{\beta(\alpha_{\%})} &= \frac{1-0.025}{0.025} && \Longleftrightarrow && (3.61d)\\
\beta\left(\alpha_{\%}\right) &= \ln\left(\frac{1-0.025}{0.025}\right) && \Longleftrightarrow && (3.61e)\\
\beta &= \frac{\ln\left(\frac{1-0.025}{0.025}\right)}{\alpha_{\%}}. && && (3.61f)
\end{aligned}$$

The symmetry of the function around the activation point means that this β gives rise to the desired activation characteristics. Alternatively, we can derive a similar expression for the upper activation limit, $\alpha_{\%,u}$, and arrive at

$$\beta = -\frac{\ln\left(\frac{1-0.975}{0.975}\right)}{\alpha_{\%}}, \tag{3.62}$$

for the same value of β. Figure 3.3 illustrates the modified catabolic repression function for different scaling parameters, β. We note that the scaled sigmoid function approximates the piece-wise constant activation function as the value of the scaling parameter increases, i.e. $\xi^{(\beta)}(\cdot) \longrightarrow \xi^{(\infty)}$ for $\beta \longrightarrow \infty$. A more reasonable implementation of catabolic repression can be implemented with the scaled sigmoid function, where the value of the scaling parameter, β, is chosen or estimated for each of the components; ammonium, nitrite, and nitrate.

3.5 Reduced metabolic model

The model presented in this section is based on the paper listed in Appendix A. In this section, we present a reduced metabolic model based on the model presented in section 3.3. We propose to include only two of the seven metabolic reactions for control applications. The growth of *M. capsulatus* on methane and ammonium in the presence of oxygen, i.e. aerobic conditions, is described by the stoichiometry

$$\begin{aligned}
CH_4 + O_2 + H_2O &\longrightarrow CO_2 + 4\,H_3O^+, && r_1, && (3.63a)\\
CH_4 + O_2 + \frac{2}{10}\,NH_4^+ &\longrightarrow CH_{1.8}O_{0.5}N_{0.2} + \frac{15}{10}\,H_2O, && r_2. && (3.63b)
\end{aligned}$$

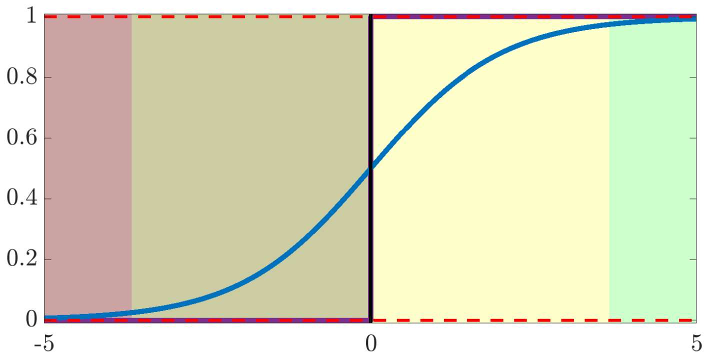


Figure 3.2: Illustration of activation functions $\xi(\cdot)$ (blue) and $\xi_\infty(\cdot)$ (purple). The red region illustrates where the activation function is almost entirely inactive, i.e. less than 2.5% activation. The yellow region indicates that the activation function is transitional, i.e. between 2.5% and 97.5% activation. The green region indicates that the activation function is almost entirely active, i.e. more than 97.5% activation. The grey shaded area indicates where $c < 0$, i.e. states which are not feasible in practice for chemical systems.

Stoichiometry

In this model, we consider the liquid and gaseous components

$$\mathcal{C}_l = \{X, N\}, \qquad \mathcal{C}_g = \{S, O, C\}, \tag{3.64}$$

where

$$X = \mathrm{CH_{1.8}O_{0.5}N_{0.2}}, \quad N = \mathrm{NH_4}^+, \quad S = \mathrm{CH_4}, \quad O = \mathrm{O_2}, \quad C = \mathrm{CO_2}. \tag{3.65}$$

The resulting modelled reactive system is described by the stoichiometry

$$S + O \longrightarrow C, \qquad r_1(c), \tag{3.66a}$$

$$S + O + \frac{2}{10}N \longrightarrow X, \qquad r_2(c). \tag{3.66b}$$

We describe the stoichiometry defined in (3.66) by the stoichiometric matrix

$$S = \begin{array}{c} \\ \\ \end{array}\begin{array}{cccccccc|c} \mathrm{X} & \mathrm{N} & \mathrm{S} & \mathrm{O} & \mathrm{C} & \mathrm{S}_g & \mathrm{O}_g & \mathrm{C}_g & \\ 0 & 0 & -1 & -1 & 1 & 0 & 0 & 0 & \mathrm{r}_1 \\ 1 & -\frac{2}{10} & -1 & -1 & 0 & 0 & 0 & 0 & \mathrm{r}_2 \end{array}. \tag{3.67}$$

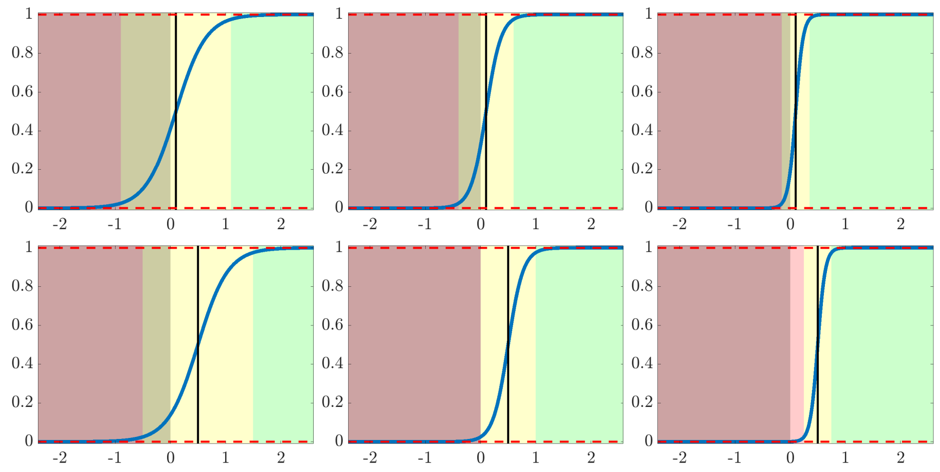


Figure 3.3: Illustration of scaled activation functions $\xi^{(\beta)}(\cdot)$ for different choices of scaling parameter, β. The illustrations are for: $\beta = 1.0$ and $K_c = 0.1$ (top left), $\beta = 0.5$ and $K_c = 0.1$ (top center), $\beta = 0.25$ and $K_c = 0.1$ (top right), $\beta = 1.0$ and $K_c = 0.5$ (bottom left), $\beta = 0.5$ and $K_c = 0.5$ (bottom center), and $\beta = 0.25$ and $K_c = 0.25$ (bottom right). The red region illustrates where the activation function is almost entirely inactive, i.e. less than 2.5% activation. The yellow region indicates that the activation function is transitional, i.e. between 2.5% and 97.5% activation. The green region indicates that the activation function is almost entirely active, i.e. more than 97.5% activation. The grey shaded area indicates where $c < 0$, i.e. states which are not feasible in practice for chemical systems.

Reaction rate

The vector of reaction rates for each reaction of the growth model is

$$r(c) = \begin{bmatrix} r_1(c) \\ r_2(c) \end{bmatrix}. \tag{3.68}$$

The reaction rates for $i \in \{1, 2\}$ are

$$r_i(c) = \mu_i(c) c_X. \tag{3.69}$$

The growth rate for the co-metabolic reaction, r_1, is

$$\mu_1(c) = \left(\frac{\alpha}{2\delta} + \frac{8}{20} \right) \mu_2(c) + \frac{m}{2\delta}, \tag{3.70}$$

where α is a stoichiometric coefficient describing the ATP demand in biomass forming reactions, δ is a stoichiometric coefficient describing ATP yield from the electron transport chain, and m is the maintenance rate. The growth rate for the anabolic reaction, r_2, is

$$\mu_2(c) = \mu_{\max}\mu_S(c)\mu_N(c)\mu_O(c), \tag{3.71}$$

where $\mu_{\max}$ is the maximum growth rate, μ_S is the specific growth rate on substrate, and μ_N is the specific growth rate on ammonium. The specific growth rate on substrate competes with a co-metabolic reaction and is therefore inhibited by the concentration of the nitrogen source, i.e. ammonium. As such, the specific growth rate on substrate is

$$\mu_S(c) = \frac{c_S}{K_S\left(1 + \frac{c_N}{K_{N,ox}}\right) + c_S}, \tag{3.72}$$

where K_S is the saturation constant and $K_{N,ox}$ is the nitrogen inhibition constant. We apply Monod kinetics to describe the specific growth rate on nitrogen, as

$$\mu_N(c) = \frac{c_N}{K_N + c_N}, \tag{3.73}$$

where K_N is the saturation constant. We apply Monod kinetics to describe the specific growth rate on oxygen, as

$$\mu_O(c) = \frac{c_O}{K_O + c_O}, \tag{3.74}$$

where K_O is the saturation constant. Table 3.4 describe the kinetic parameters for the model.

3.6 Summary

In this chapter, we presented growth models for cultivation of *Methylococcus capsulatus* (Bath) for SCP production. We applied the modelling framework for reactive systems and defined a series of growth models in terms of their stoichiometry matrix and vector of reaction rates. We described a growth model based on methanol and nitric acid as carbon and nitrogen sources, respectively. We applied Haldane kinetics for the growth on substrate, i.e. methanol, and described optimal growth conditions for the model. We presented a growth model based on methane and ammonia as carbon and nitrogen sources, respectively. We presented a model based on a metabolic study, including seven metabolic reactions describing the anabolism from ammonium, nitrite, nitrate, and molecular nitrogen with methane as carbon source. We presented developments made for this by introducing a scaled sigmoid activation function as the model for catabolic repression. Finally, we presented a reduced model based on the metabolic model including the anabolism of ammonium with methane as carbon source. Control applications will be presented using this model later in this work.

Table 3.4: Kinetic parameters for the reduced metabolic model. The parameters are from the paper listed in Appendix A.

Symbol	**Value**	**Unit**
$\mu_{\max}$	$2.3 \cdot 10^{-1}$	1/h
α	$2.0 \cdot 10^{-2}$	mol ATP/molc X
δ	$2.0 \cdot 10^{-2}$	mol ATP/mol(2e)
m	$9.8 \cdot 10^{-5}$	mol ATP/(molc X $\cdot$ h)
K_S	$7.5 \cdot 10^{-5}$	mol/L
K_O	$5.5 \cdot 10^{-7}$	mol/L
K_N	$1.3 \cdot 10^{-3}$	mol/L
$K_{N,ox}$	$3.3 \cdot 10^{-3}$	mol/L

CHAPTER 4

pH Equilibrium Dynamics

The work presented in this chapter is partially based on the paper listed in Appendix A. In this chapter, we investigate pH equilibrium dynamics in reactive systems. We describe general methods of modelling pH equilibrium by minimising Gibbs free energy and by applying thermodynamic equilibrium constants. Additionally, we present equilibrium reactions relevant to the modelling of SCP production. In the models for chemical equilibrium, we consider in this work, we assume that the reaction rates for the equilibrium reactions are negligible compared to the other time-dependent system dynamics, i.e. the equilibrium reactions are instantaneous.

The chapter is structured at follows: In section 4.1, we briefly describe chemical equilibrium dynamics in the context of minimisation of Gibbs free energy. In section 4.2, we introduce the thermodynamic equilibrium constant and describe how this can be used to compute chemical equilibrium. Section 4.3 presents chemical equilibrium reactions relevant to the production of SCP. Finally, we summarise the chapter in section 4.4.

4.1 Gibbs free energy minimisation

In this section, we present computations for chemical equilibria by Gibbs free energy minimisation. Mole numbers in equilibrium reactions can be determined by minimisation of Gibbs free energy. Gibbs free energy is a thermochemical property describing the free enthalpy of a chemical systems at constant pressure and temperature [19, 85]. Consider the general equilibrium reaction

$$x_1 X_1 + x_2 X_2 + \ldots \rightleftharpoons y_1 Y_1 + y_2 Y_2 + \ldots, \tag{4.1}$$

where x_i and y_i are stoichiometric coefficients and $X_i \in \mathcal{C}$ and $Y_i \in \mathcal{C}$ are chemical species in the set of chemical species involved in the chemical equilibrium reaction, $\mathcal{C}$. We note that we can also express the reaction in the form of the algebraic equation

$$0 = y_1 Y_1 + y_2 Y_2 + \cdots - x_1 X_1 - x_2 X_2 - \cdots = S^T C, \tag{4.2}$$

where S is the stoichiometric matrix and C is a vector of components, i.e. reactants and products. The reaction is at equilibrium when the Gibbs free energy is minimised, i.e. when the change in Gibbs free energy is zero. As such, we define the chemical equilibrium problem

$$\min_{n} \quad G(n), \tag{4.3}$$

subject to

$$\begin{aligned} An &= b, && \text{(4.4a)} \\ n &\geq 0, && \text{(4.4b)} \end{aligned}$$

where $n \in \mathbb{R}^{n_c}$ are the mole numbers for the chemical components, $G(\cdot) : \mathbb{R}^{n_c} \longrightarrow \mathbb{R}$ is the Gibbs free energy of the system, $A \in \mathbb{R}^{n_a \times n_c}$ is a matrix describing the atomic composition of the chemical components, and $b = An_0$, where n_0 is the initial mole numbers of the system. We describe the Gibbs free energy of the system of liquid components as

$$G(n) = n^T g(T, P, \lambda), \qquad \lambda = \frac{n}{e^T n}, \tag{4.5}$$

where λ are the mole fraction of the components, $e \in \{1\}^{n_c}$ is a vector of ones, $g(\cdot)$ is the Gibbs free energy for the components, T is the temperature, and P is the pressure. The component-wise Gibbs free energy is described by the vectorised expression

$$g(T, P, \lambda) = g^0(T, P) + RT \ln(\lambda). \tag{4.6}$$

We may describe the equilibrium state as the solution to (4.5), i.e. by the 1st order optimality, KKT, conditions, as

$$\begin{aligned} 0 &= \nabla \mathcal{L}(n, \mu), && \text{(4.7a)} \\ 0 &= An - b, && \text{(4.7b)} \\ 0 &\leq n, && \text{(4.7c)} \end{aligned}$$

and solve for the mole numbers n [49]. Standard thermodynamic quantities are described in [86, 87]. Notably, we may use the algebraic equations in (4.7) together with growth and reactor dynamics, to formulate a differential equation model describing growth and chemical equilibrium in a reactor. Growth models are described in Chapter 3 and chemical reactor models are described in Chapter 2 and in the technical report listed in Appendix G. Nonlinear systems involving SDAEs are described in Chapter 6.

4.2 Thermodynamic equilibrium constant

Closely related to the Gibbs free energy minimisation is the thermodynamic equilibrium constant, K_r, for a reaction, r. In this section, we describe models of chemical equilibrium applying thermodynamic equilibrium constants [19, 85]. Consider the general equilibrium reaction

$$x_1 X_1 + x_2 X_2 + \ldots \rightleftharpoons y_1 Y_1 + y_2 Y_2 + \ldots, \qquad K_r, \qquad \Delta G_r^0, \qquad r_+, \qquad r_-, \tag{4.8}$$

where r_+ and r_- are the forward and backward reaction rates for the equilibrium reaction. The reaction rates are

$$r_+ = k_+ \prod [X_i]^{x_i}, \tag{4.9a}$$
$$r_- = k_- \prod [Y_i]^{y_i}, \tag{4.9b}$$

where k_+ and k_- are kinetics parameters for the forward and backward directions of the equilibrium reaction, respectively. The reaction is at equilibrium when the change in Gibbs free energy is zero

$$0 = \Delta G_r \tag{4.10a}$$
$$= \Delta G_r^0(T) + RT \ln (K_r), \tag{4.10b}$$

where $\Delta G_r^0(T) = \Delta H_r^0 - T\Delta S_r^0$ is the standard Gibbs free energy of the reaction and ΔH_r^0 and ΔS_r^0 are the standard enthalpy and entropy of the reaction, respectively [19]. T is the temperature and R is the ideal gas constant. The equilibrium constant arises from (4.10), as

$$0 = \Delta G_r^0(T) + RT \ln (K_r) \qquad \Longleftrightarrow \tag{4.11a}$$
$$\ln (K_r) = -\frac{\Delta G_r^0(T)}{RT} \qquad \Longleftrightarrow \tag{4.11b}$$
$$K_r = \exp\left(-\frac{\Delta G_r^0(T)}{RT}\right). \tag{4.11c}$$

At equilibrium, the concentrations of reactants and products satisfy

$$r_+ = r_- \qquad \Longleftrightarrow \tag{4.12a}$$
$$k_+ \prod [X_i]^{x_i} = k_- \prod [Y_i]^{y_i} \qquad \Longleftrightarrow \tag{4.12b}$$
$$K_r = \frac{k_+}{k_-} = \frac{\prod [Y_i]^{y_i}}{\prod [X_i]^{x_i}}. \tag{4.12c}$$

Using (4.12c), we may compute the chemical equilibrium of a reactive system. Consider the general system of N coupled equilibrium reactions

$$x_1^{(1)} X_1 + x_2^{(1)} X_2 + \cdots \rightleftharpoons y_1^{(1)} Y_1 + y_2^{(1)} Y_2 + \cdots, \qquad K_{r_1}, \qquad \Delta G_{r_1}^0, \tag{4.13a}$$
$$x_1^{(2)} X_1 + x_2^{(2)} X_2 + \cdots \rightleftharpoons y_1^{(2)} Y_1 + y_2^{(2)} Y_2 + \cdots, \qquad K_{r_2}, \qquad \Delta G_{r_2}^0, \tag{4.13b}$$
$$\vdots \tag{4.13c}$$
$$x_1^{(N)} X_1 + x_2^{(N)} X_2 + \cdots \rightleftharpoons y_1^{(N)} Y_1 + y_2^{(N)} Y_2 + \cdots, \qquad K_{r_N}, \qquad \Delta G_{r_N}^0. \tag{4.13d}$$

Assuming that all reactions reach equilibrium instantaneously, we can compute the equilibrium concentrations as the solution to the system of nonlinear algebraic equations

$$0 = \prod [Y_i]^{y_i^{(1)}} - K_{r_1} \prod [X_i]^{x_i^{(1)}}, \tag{4.14a}$$

$$0 = \prod [Y_i]^{y_i^{(2)}} - K_{r_2} \prod [X_i]^{x_i^{(2)}}, \tag{4.14b}$$

$$\vdots \tag{4.14c}$$

$$0 = \prod [Y_i]^{y_i^{(N)}} - K_{r_N} \prod [X_i]^{x_i^{(N)}}, \tag{4.14d}$$

as well as mass and charge balance equations for the component concentrations, $[X_i]$ and $[Y_i]$ for $i \in \{1, 2, \ldots, N\}$. Similarly to the previous section, we may use (4.14), mass, and energy balance equations, together with growth and reactor models to formulate a differential algebraic system describing growth and chemical equilibrium in a reactor.

4.3 pH equilibrium reactions in single-cell protein production

In this section, we present acid-base equilibrium reactions in aqueous solution relevant to the production of SCP. In all equilibrium reactions, we assume that the temperature and pressure are constant. Therefore, the thermodynamic equilibrium constants will also be considered constant. Furthermore, we assume that all reaction are instantaneous, i.e. the system is at equilibrium at all times. We note that pH is computed as

$$pH = -\log_{10} \left[\mathrm{H_3O^+}(aq)\right] \tag{4.15}$$

4.3.1 Equilibrium reactions

Water

In aqueous solution, we describe the equilibrium of water, hydronium, and hydroxide by the reaction

$$2\,\mathrm{H_2O(aq)} \rightleftharpoons \mathrm{H_3O^+(aq)} + \mathrm{OH^-(aq)}, \qquad K_W, \tag{4.16}$$

where K_W is the equilibrium constant for water. As such, the concentrations of hydronium and hydroxide satisfies the algebraic expression

$$K_W = [\mathrm{H_3O^+}]\,[\mathrm{OH^-}]. \tag{4.17}$$

Carbon dioxide

In aqueous solution, we describe the equilibrium of carbon dioxide and carbonic acid by the reaction

$$\mathrm{CO_2(aq) + H_2O\,(aq) \rightleftharpoons H_2CO_3(aq)}, \qquad K_{C,1}, \tag{4.18}$$

where $K_{C,1}$ is the equilibrium constant for the reaction. As such, the concentrations of carbon dioxide and carbonic acid satisfies the algebraic expression

$$K_{C,1} = \frac{[\mathrm{H_2CO_3}]}{[\mathrm{CO_2}]}. \tag{4.19}$$

We describe the equilibrium of carbonic acid, bicarbonate, and hydronium by the reaction

$$\mathrm{H_2CO_3(aq) + H_2O\,(aq) \rightleftharpoons HCO_3{}^-(aq) + H_3O^+(aq)}, \qquad K_{C,2}, \tag{4.20}$$

where $K_{C,2}$ is the equilibrium constant for the reaction. As such, the concentrations of carbonic acid, bicarbonate, and hydronium satisfies the algebraic expression

$$K_{C,2} = \frac{[\mathrm{HCO_3{}^-}]\,[\mathrm{H_3O^+}]}{[\mathrm{H_2CO_3{}^-}]}. \tag{4.21}$$

We describe the equilibrium of bicarbonate, carbonate, and hydronium by the reaction

$$\mathrm{HCO_3{}^-(aq) + H_2O\,(aq) \rightleftharpoons CO_3{}^{2-}(aq) + H_3O^+(aq)}, \qquad K_{C,3}, \tag{4.22}$$

where $K_{C,3}$ is the equilibrium constant for the reaction. As such, the concentrations of bicarbonate, carbonate, and hydronium satisfies the algebraic expression

$$K_{C,3} = \frac{[\mathrm{CO_3{}^{2-}}]\,[\mathrm{H_3O^+}]}{[\mathrm{HCO_3{}^-}]}. \tag{4.23}$$

Phosphoric acid

In aqueous solution, we describe the equilibirium of phosphoric acid, dihydrogen phosphate, and hydronium by the reaction

$$\mathrm{H_3PO_4(aq) + H_2O\,(aq) \rightleftharpoons H_2PO_4{}^-(aq) + H_3O^+(aq)}, \qquad K_{P,1}, \tag{4.24}$$

where $K_{P,1}$ is the equilibrium constant for the reaction. As such, the concentrations of phosphoric acid, dihydrogen phosphate, and hydronium satisfies the algebraic expression

$$K_{P,1} = \frac{[\mathrm{H_2PO_4{}^-}]\,[\mathrm{H_3O^+}]}{[\mathrm{H_3PO_4}]}. \tag{4.25}$$

We describe the equilibrium of dihydrogen phosphate, hydrogen phosphate, and hydronium by the reaction

$$H_2PO_4{}^{-}(aq) + H_2O\,(aq) \rightleftharpoons HPO_4{}^{2-}(aq) + H_3O^{+}(aq), \qquad K_{P,2}, \tag{4.26}$$

where $K_{P,2}$ is the equilibrium constant for the reaction. As such, the concentrations of dihydrogen phosphate, hydrogen phosphate, and hydronium satisfies the algebraic expression

$$K_{P,2} = \frac{[HPO_4{}^{2-}]\,[H_3O^{+}]}{[H_2PO_4{}^{-}]}. \tag{4.27}$$

We describe the equilibrium of hydrogen phosphate, phosphate, and hydronium by the reaction

$$HPO_4{}^{2-}(aq) + H_2O\,(aq) \rightleftharpoons PO_4{}^{3-}(aq) + H_3O^{+}(aq), \qquad K_{P,3}, \tag{4.28}$$

where $K_{P,3}$ is the equilibrium constant for the reaction. As such, the concentrations of hydrogen phosphate, phosphate, and hydronium satisfies the algebraic expression

$$K_{P,3} = \frac{[PO_4{}^{3-}]\,[H_3O^{+}]}{[HPO_4{}^{2-}]}. \tag{4.29}$$

Ammonia

In aqueous solution, we describe the equilibrium of ammonia, ammoinium, and hydroxide by the reaction

$$NH_3(aq) + H_2O\,(aq) \rightleftharpoons NH_4{}^{+}(aq) + OH^{-}(aq), \qquad K_N, \tag{4.30}$$

where K_N is the equilibrium constant for the reaction. As such, the concentrations of ammonia, ammonium, and hydroxide satisfies the algebraic expression

$$K_N = \frac{[NH_4{}^{+}]\,[H_3O^{+}]}{[NH_3]}. \tag{4.31}$$

Nitric acid

Nitric acid is a strong acid, and we therefore assume that nitric acid in aqueous solution completely dissolves into nitrate and hydronium. The reaction describing the dissolution of nitric acid is

$$HNO_3(aq) + H_2O\,(aq) \longrightarrow NO_3{}^{-}(aq) + H_3O^{+}(aq). \tag{4.32}$$

Sodium hydroxide

Sodium hydroxide is a strong base, and we therefore assume that sodium hydroxide in aqueous solution completely dissolves into sodium and hydroxide. The reaction describing the dissolution of sodium hydroxide is

$$NaOH\,(aq) \longrightarrow Na^{+}(aq) + OH^{-}(aq). \tag{4.33}$$

4.3.2 Mass and energy balances

Mass balances

We consider the sets of molecules relating to carbon dioxide, phosphoric acid, and ammonia, as

$$\mathcal{C}_C = \{CO_2(aq), H_2CO_3(aq), {HCO_3}^-(aq), {CO_3}^{2-}(aq)\}, \tag{4.34a}$$
$$\mathcal{C}_P = \{H_3PO_4(aq), {H_2PO_4}^-(aq), {HPO_4}^{2-}(aq), {PO_4}^{3-}(aq)\}, \tag{4.34b}$$
$$\mathcal{C}_{NH} = \{NH_3(aq), {NH_4}^+(aq)\}, \tag{4.34c}$$

respectively. We note that, for strong acids and bases, the reactions are complete, and therefore the concentration of the ions are equal to the total concentration. Therefore, we do not describe mass balances for complete reactions. We define the mass balances in terms of the known total concentrations equal, as

$$c_i = \sum_{j \in \mathcal{C}_i} c_j, \qquad i \in \{C, P, NH\}, \tag{4.35}$$

where $c_j = [j]$ is the concentration of component j. We describe the mass balances for carbon dioxide, phosphoric acid, and ammonia, as

$$0 = c_C - \left([CO_2(aq)] + [H_2CO_3(aq)] + \left[{HCO_3}^-(aq)\right] + \left[{CO_3}^{2-}(aq)\right]\right), \tag{4.36a}$$
$$0 = c_P - \left([H_3PO_4(aq)] + \left[{H_2PO_4}^-(aq)\right] + \left[{HPO_4}^{2-}(aq)\right] + \left[{PO_4}^{3-}(aq)\right]\right), \tag{4.36b}$$
$$0 = c_{NH} - \left([NH_3(aq)] + \left[{NH_4}^+(aq)\right]\right). \tag{4.36c}$$

Charge balance

We define the general charge balance equation

$$0 = \sum_{i \in \mathcal{P}} \alpha_i c_i - \sum_{i \in \mathcal{N}} \alpha_i c_i, \tag{4.37}$$

where $\mathcal{P}$ is the set of positively charged ions with charge numbers α_i for $i \in \mathcal{P}$ and $\mathcal{N}$ is the set of negatively charged ions with charge numbers α_i for $i \in \mathcal{N}$. We describe the charge balances for the chemical equilibrium reactions, as

$$\begin{aligned} 0 = &\left[H_3O^+(aq)\right] + \left[{NH_4}^+(aq)\right] + \left[Na^+(aq)\right] \\ &- \left(\left[OH^-(aq)\right]\right] + \left[{HCO_3}^-(aq)\right] + 2\left[{CO_3}^{2-}(aq)\right] \\ &+ \left[{H_2PO_4}^-(aq)\right] + 2\left[{HPO_4}^{2-}(aq)\right] + 3\left[{PO_4}^{3-}(aq)\right] + \left[{NO_3}^-(aq)\right]\right). \end{aligned} \tag{4.38}$$

This results in a system of 8 equilibrium equations, 3 mass balance equations, and 1 charge balance equation, i.e. 12 algebraic equations with 12 algebraic variables.

4.4 Summary

In this chapter, we presented models for calculation of equilibrium in chemical equilibrium reactions. We briefly described chemical equilibrium by minimisation of Gibbs free energy of a system. Motivated by Gibbs free energy minimisation, we introduced the thermodynamic equilibrium constant and relate it to Gibbs free energy minimisation. We described chemical equilibrium calculations based on thermodynamic equilibrium constants, as well as mass and charge balances. Finally, we described equilibrium reaction related to single-cell protein production. We described water in equilibrium with hydronium and hydroxide, as well as equilibrium reactions relating to carbon dioxide, phosphoric acid, and ammonia. We also described strong acid and base reactions for sodium hydroxide and nitric acid. We presented mass and charge balances for these equilibrium reactions, resulting in an algebraic systems of 8 equilibrium equations, 3 mass balance equations, and 1 charge balance equation, for a total for 12 algebraic equations solved for concentrations of algebraic components at chemical equilibrium.

CHAPTER 5 Systems Involving Stochastic Differential Equations

In this chapter, we present SDEs and describe methods for computing numerical solutions to such systems, i.e. numerical integration schemes. We present the explicit-explicit Euler-Maruyama method for numerical integration of systems with non-stiff dynamics and an implicit-explicit method for numerical integration of systems with stiff dynamics. We present Newton's method for application in the implicit-explicit numerical SDE solver. In this chapter, we consider SDE models in the form

$$d\boldsymbol{x}(t) = f(\boldsymbol{x}(t), u(t), \theta)dt + \sigma(\boldsymbol{x}(t), u(t), \theta)d\boldsymbol{\omega}(t), \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \tag{5.1}$$

where $\boldsymbol{x}(t) \in \mathbb{R}^{n_x}$ are state variables, $u(t) \in \mathbb{R}^{n_u}$ are manipulated variables, $\boldsymbol{\omega}(t) \in \mathbb{R}^{n_\omega}$ is a standard Wiener process, i.e. $d\boldsymbol{\omega} \sim \mathcal{N}(0, Idt)$, and θ are parameters. $f(\cdot) : (\boldsymbol{x}(t), u(t), \theta) \longrightarrow \mathbb{R}^{n_x}$ are the drift state dynamics and $\sigma(\cdot) : (\boldsymbol{x}(t), u(t), \theta) \longrightarrow \mathbb{R}^{n_x \times n_\omega}$ are the state diffusion dynamics.

The chapter is structured as follows: in section 5.1, we describe the explicit-explicit Euler-Maruyama scheme for numerical integration of SDEs with non-stiff dynamics. In section 5.2, we describe an implicit-explicit scheme for numerical integration of SDEs with stiff dynamics. Section 5.3 presents Newton's method for iterative root-finding as it relates to solution of the residual equations arising from the implicit-explicit integration scheme presented in the previous section. We also discuss non-negativity constraints in the context of Newton's method. Finally, section 5.4 summarises the work presented in the chapter.

5.1 Explicit-explicit Euler-Maruyama method

In this section, we describe the explicit-explicit Euler-Maruyama method for numerical integration of SDEs. The Euler-Maruyama scheme is an extension of the explicit

Euler (forward Euler) method for numerical integration of ODEs to stochastic systems [88]. Consider the solution to (7.1) over the interval $t \in [t_k, t_{k+1}]$ on integral form

$$\boldsymbol{x}(t_{k+1}) = \boldsymbol{x}(t_k) + \int_{t_k}^{t_{k+1}} f(\boldsymbol{x}(s), u(s), \theta) ds + \int_{t_k}^{t_{k+1}} \sigma(\boldsymbol{x}(s), u(s), \theta) d\boldsymbol{\omega}(s), \tag{5.2}$$

where the stochastic integral is an Ito integral [51]. The explicit-explicit Euler-Maruyama method defines the solution at time $t = t_{k+1}$ as the series of numerical integrations applying the right rectangular rule at discrete times $t_{k,n} = t_k + n\Delta t_{k,n}$

$$x_{k,n+1} = x_{k,n} + f(x_{k,n}, u_{k,n}, \theta)\Delta t_{k,n} + \sigma(x_{k,n} u_{k,n}, \theta)\Delta\omega_{k,n}, \tag{5.3}$$

for $n \in \{0, 1, \dots, N-1\}$. We consider the definition the state variables $\boldsymbol{x}_{k,n} = \boldsymbol{x}(t_{k,n})$, the inputs variables $u_{k,n} = u(t_{k,n}) = u_k$ for zero-order hold within the interval $t \in [t_k, t_{k+1}]$, and the process noise $\Delta\omega_{k,n} = z_{k,n}\sqrt{\Delta t_{k,n}}$ for $z_{k,n} \sim \mathcal{N}(0, 1)$ [89]. The initial state, $x_{k,0} = x(t_{k,0})$, is assumed known. We note that for this numerical integration scheme, we may explicitly compute the solution $x_{k,N} = x(t_{k+1})$.

5.2 Implicit-explicit method

In this section, we describe an implicit-explicit method for numerical integration of SDEs. Consider the solution to (7.1) over the interval $t \in [t_k, t_{k+1}]$ on integral form

$$\boldsymbol{x}(t_{k+1}) = \boldsymbol{x}(t_k) + \int_{t_k}^{t_{k+1}} f(\boldsymbol{x}(s), u(s), \theta) ds + \int_{t_k}^{t_{k+1}} \sigma(\boldsymbol{x}(s), u(s), \theta) d\boldsymbol{\omega}(s). \tag{5.4}$$

For the implicit-explicit method, we define the solution at time $t = t_{k+1}$ as the series of solutions at discrete times $t_{k,n} = t_k + n\Delta t_{k,n}$

$$x_{k,n+1} = x_{k,n} + f(x_{k,n+1}, u_{k,n}, \theta)\Delta t_{k,n} + \sigma(x_{k,n}, u_{k,n}, \theta)\Delta\omega_{k,n}, \tag{5.5}$$

for $n \in \{0, 1, \dots, N-1\}$. We note, that as the name suggests, we cannot explicitly describe the solutions at each time-step for systems with nonlinear state dynamics. Instead, we define the solution at the next time-step implicitly as the roots of the residual arising from (5.5), i.e. solutions satisfying

$$\begin{aligned} R(x_{k,n+1}) &= 0 && \text{(5.6a)} \\ &= x_{k,n+1} - x_{k,n} - f(x_{k,n+1}, u_{k,n}, \theta)\Delta t_{k,n} - \sigma(x_{k,n}, u_{k,n}, \theta)\Delta\omega_{k,n}. && \text{(5.6b)} \end{aligned}$$

The numerical solutions to the SDE, $x_{k+1} = x_{k,N}$, arise as roots in the residual in (5.6). We may compute roots iteratively by application of a root-finding algorithm, e.g. Newton's method. Newton's method requires computation of the Jacobian of the residual function. We compute the Jacobian of the residual function in (5.6), $R(x)$, as

$$\frac{\partial R}{\partial x}(x_{k,n+1}) = I - \frac{\partial f}{\partial x}(x_{k,n+1}, u_{k,n}, \theta)\Delta t_{k,n}. \tag{5.7}$$

5.3 Newton's method

In this section, we describe Newton's method (also called Newton-Raphson's method). Newton's method is an algorithm for iteratively finding roots in a function. The method is ubiquitous in numerical techniques, e.g. optimisation and simulation [49, 90, 91]. Consider the nonlinear function $R(x) : \mathbb{R}^{n_x} \longrightarrow \mathbb{R}^{n_x}$. Consider the problem of finding roots for the function, $R(x)$, i.e. values of x satisfying

$$0 = R(x). \tag{5.8}$$

We apply Newton's method to find a value of x which satisfy (5.8), by iteratively solving

$$x_{n+1} = x_n - \left(\frac{\partial R}{\partial x}(x_n)\right)^{-1} R(x_n), \qquad n \in \{0, 1, \dots\}, \tag{5.9}$$

until the residual function defined in (5.6) converges to a desired tolerance, i.e. $R(x_n) < \epsilon$. In cases where the Jacobian of the residual is singular, i.e. $\frac{\partial R}{\partial x}^{-1}$ does not exist, we apply the pseudo-inverse, $\frac{\partial R}{\partial x}^{\dagger}$, [92].

5.3.1 Non-negative roots

For some states representing physical quantities, the state variables have non-negativity constraints, i.e. only non-negative states correspond to physically feasible solutions. This applies, for instance, to chemical or biochemical systems, where the states represent mole number, concentrations, number of cells, or volumes in the reactor. Therefore, enforcement of such non-negativity constraints is highly relevant in the context of SCP production, and therefore the work presented in this thesis. We propose two methods of enforcing non-negativity constraints in Newton's method; 1) bounding the solution to be non-negative in each iteration and 2) taking the absolute value of the solution in each iteration. The methods proposed here do not maintain the convergence properties of Newton's method, but show good properties in practice [93].

Bounded Newton

In the so-called bounded Newton, we compute the n'th iteration of Newton's method and bound the solution to the step by a small value, as

$$x_{n+1} = \max\left\{ x_n - \left(\frac{\partial R}{\partial x}(x_n)\right)^{-1} R(x_n), \epsilon \right\}, \tag{5.10}$$

where ϵ is a small value close to zero or zero.

Absolute Newton

In so-called absolute Newton, we compute the n'th iteration of Newton's method and take the absolute value of the solution, as

$$x_{n+1} = \text{abs}\left\{ x_n - \left(\frac{\partial R}{\partial x}(x_n) \right)^{-1} R(x_n) \right\}. \tag{5.11}$$

5.4 Summary

In this chapter, we presented methods for numerical integration of SDEs. We presented the explicit-explicit Euler-Maruyama method and an implicit-explicit method for numerical integration of SDEs. For the explicit-explicit Euler-Maruyama method, the numerical solution to the state at the next time-step is defined explicitly in terms of the states at the previous time-step. For the implicit-explicit method, we defineed the solution to the state at the next time-step implicitly, as roots in a residual function, in terms of the state at the previous time-step. To compute roots in the residual function, we presented Newton's method for iterative root-finding. Finally, we presented two methods of introducing non-negativity constraints in the step computation of Newton's method. This is relevant for chemical and biochemical systems, where the state variables represent physical states which cannot be negative, e.g. concentrations or volumes.

CHAPTER 6

Systems Involving Stochastic Differential Algebraic Equation

This chapter is partially based on the work presented in the paper listed in Appendix A. In this chapter, we present SDAEs and describe methods for numerical integration of such systems. We present an implicit-explicit numerical integration scheme for systems with stiff (and non-stiff) state and algebraic dynamics. Finally, we describe how differential algebraic systems may lead to ill-conditioning in the Jacobian which may cause Newton's method to diverge. We address this issue of ill-conditioning by introducing linear scaling of the algebraic function and variables. In this chapter, we consider SDAEs in the form

$$d\boldsymbol{x}(t) = f(\boldsymbol{x}(t), \boldsymbol{y}(t), u(t), \theta)dt + \sigma(\boldsymbol{x}(t), \boldsymbol{y}(t), u(t), \theta)d\boldsymbol{\omega}(t), \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \tag{6.1a}$$

$$0 = g(\boldsymbol{x}(t), \boldsymbol{y}(t), \theta), \tag{6.1b}$$

where $f(\cdot) : (\boldsymbol{x}(t), \boldsymbol{y}(t), u(t), \theta) \longrightarrow \mathbb{R}^{n_x}$ is the drift function, $\sigma(\cdot) : (\boldsymbol{x}(t), \boldsymbol{y}(t), u(t), \theta) \longrightarrow \mathbb{R}^{n_x \times n_\omega}$ is the diffusion function, and $g(\cdot) : (\boldsymbol{x}(t), \boldsymbol{y}(t), \theta) \longrightarrow \mathbb{R}^{n_y}$ is the algebraic function. The variable t is time, $\boldsymbol{x}(t) \in \mathbb{R}^{n_x}$ are states, $y(t) \in \mathbb{R}^{n_y}$ are algebraic variables, $u(t) \in \mathbb{R}^{n_u}$ are inputs, and θ are parameters. The process noise $\boldsymbol{\omega}(t) \in \mathbb{R}^{n_\omega}$ is a standard Weiner process, i.e. $d\boldsymbol{\omega}(t) \sim \mathcal{N}(0, Idt)$.

The chapter is structured as follows: in section 6.1, we present an implicit explicit numerical integration scheme for solution of SDAEs. We present an implicit definition of the solution at the next time-step and apply Newton's method to solve the resulting root-finding problem. Ill-conditioning in the Jacobian of the residual may lead to divergence in Newton's method. In section 6.2, we introduce a linear scaling of the algebraic function and variables to address the problem of ill-conditioning. Finally, we summarise the chapter in section 6.3

6.1 Implicit-explicit method

In this section, we present an extension of the implicit-explicit numerical integration scheme presented in Chapter 5 to SDAEs in the form presented in (6.1). Consider the solution to (6.1) for $t \in [t_k, t_{k+1}]$ on integral form

$$\begin{aligned} \boldsymbol{x}(t_{k+1}) = \boldsymbol{x}(t_k) &+ \int_{t_k}^{t_{k+1}} f(\boldsymbol{x}(s), \boldsymbol{y}(s), u(s), \theta) ds \\ &+ \int_{t_k}^{t_{k+1}} \sigma(\boldsymbol{x}(s), \boldsymbol{y}(s), u(s), \theta) d\boldsymbol{\omega}(s), \end{aligned} \tag{6.2}$$

where the state and algebraic variables satisfy the implicit relationship defined by

$$0 = g(\boldsymbol{x}(t), \boldsymbol{y}(t), \theta), \qquad t \in [t_k, t_{k+1}]. \tag{6.3}$$

The implicit-explicit method defines the solution at time $t = t_{k+1}$ as the series of solutions at discrete times $t_{k,n} = t_k + n\Delta t_{k,n}$

$$x_{k,n+1} = x_{k,n} + f(x_{k,n+1}, y_{k,n+1}, u_{k,n}, \theta)\Delta t_{k,n} + \sigma(x_{k,n}, y_{k,n}, u_{k,n}, \theta)\Delta\omega_{k,n}, \tag{6.4a}$$
$$0 = g(x_{k,n+1}, y_{k,n+1}, \theta), \tag{6.4b}$$

for $n \in \{0, 1, \ldots, N-1\}$. The states $x_{k,n} = x(t_{k,n})$, the algebraic variables $y_{k,n} = y(t_{k,n})$, the inputs $u_{k,n} = u(t_{k,n}) = u_k$ for zero-order hold within the interval $t \in [t_k, t_{k+1}]$, and $\Delta\omega_{k,n} = z_{k,n}\sqrt{\Delta t_{k,n}}$ for $z_{k,n} \sim \mathcal{N}(0,1)$ [89]. The initial state $x_{k,0} = x(t_{k,0})$ is assumed known. We compute the solution to each iteration defined by (6.4) by applying Newton's method, as presented in 5.3, to the residual

$$0 = R(z_{k,n+1}). \tag{6.5}$$

We define the residual function

$$R(z_{k,n+1}) = \begin{bmatrix} x_{k,n+1} - x_{k,n} - f_{k,n+1}\Delta t_{k,n} - \sigma_{k,n}\Delta\omega_{k,n} \\ g_{k,n+1} \end{bmatrix}. \tag{6.6}$$

The functions are

$$f_{k,i} = f(x_{k,i}, y_{k,i}, u_{k,i-1}, \theta), \tag{6.7a}$$
$$\sigma_{k,i} = \sigma(x_{k,i}, y_{k,i}, u_{k,i}, \theta), \tag{6.7b}$$
$$g_{k,i} = g(x_{k,i}, y_{k,i}, \theta). \tag{6.7c}$$

The variables are

$$z_{k,i} = \begin{bmatrix} x_{k,i} \\ y_{k,i} \end{bmatrix}. \tag{6.8}$$

The Jacobian of the residual in (6.6) is

$$\frac{\partial R}{\partial z}(z_{k,n+1}) = \begin{bmatrix} I - \frac{\partial f_{k,n+1}}{\partial x}\Delta t_{k,n} & -\frac{\partial f_{k,n+1}}{\partial y}\Delta t_{k,n} \\ \frac{\partial g_{k,n+1}}{\partial x} & \frac{\partial g_{k,n+1}}{\partial y} \end{bmatrix}, \tag{6.9}$$

where

$$\frac{\partial f_{k,n+1}}{\partial x} = \frac{\partial f}{\partial x}(x_{k,n+1}, y_{k,n+1}, u_{k,n}, \theta), \tag{6.10a}$$
$$\frac{\partial f_{k,n+1}}{\partial y} = \frac{\partial f}{\partial y}(x_{k,n+1}, y_{k,n+1}, u_{k,n}, \theta), \tag{6.10b}$$
$$\frac{\partial g_{k,n+1}}{\partial x} = \frac{\partial g}{\partial x}(x_{k,n+1}, y_{k,n+1}, \theta), \tag{6.10c}$$
$$\frac{\partial g_{k,n+1}}{\partial y} = \frac{\partial g}{\partial y}(x_{k,n+1}, y_{k,n+1}, \theta). \tag{6.10d}$$

6.2 On ill-conditioning of the Jacobian

In this section, we present a method for linear scaling of the algebraic function and linear parametrisation of the algebraic variables to arrive at a more well-conditioned system for numerical integration of system in the form (6.1). In the context of chemical equilibrium reactions, but also more widely, the algebraic equations may introduce large variations in the magnitude of the state and algebraic variables, and between individual algebraic variables and equations. Such differences in scale can lead to ill-conditioning in the Jacobian presented in (6.9). This ill-conditioning can in turn lead to divergence in the Newton iterations. To address this issue, we present a linear scaling of the algebraic function and linear parametrisation of the algebraic variables [93]. Consider the algebraic equations from (6.1)

$$0 = g(\boldsymbol{x}(t), \boldsymbol{y}(t), \theta). \tag{6.11}$$

We introduce the linear parametrisation of the algebraic variables,

$$\boldsymbol{y}(t) = S_y \bar{\boldsymbol{y}}(t), \tag{6.12}$$

and define the scaled algebraic function,

$$\bar{g}(\boldsymbol{x}(t), \boldsymbol{y}(t), \theta) = S_g g(\boldsymbol{x}(t), S_y \bar{\boldsymbol{y}}(t), \theta), \tag{6.13}$$

where $S_g \in \mathbb{R}^{n_g \times n_g}$ and $S_y \in \mathbb{R}^{n_y \times n_y}$ are diagonal scaling matrices, such that

$$S_g = \operatorname{diag}(s_g) = \begin{bmatrix} s_{g,1} & & & \\ & s_{g,2} & & \\ & & \ddots & \\ & & & s_{g,n_g} \end{bmatrix}, \tag{6.14a}$$

$$S_y = \operatorname{diag}(s_y) = \begin{bmatrix} s_{y,1} & & & \\ & s_{y,2} & & \\ & & \ddots & \\ & & & s_{y,n_y} \end{bmatrix}. \tag{6.14b}$$

We note that the scaled algebraic function in (6.13) and the algebraic function with no scaling in (6.1) share the same roots, as

$$0 = g(x^*, y^*, \theta) \qquad \Longrightarrow \qquad 0 = S_g g(x^*, y^*, \theta), \tag{6.15}$$

and given that $y^* = S_y \bar{y}^*$. The scaling introduced in (6.13) scales the algebraic equations, but it also scales the Jacobians of the algebraic function, as

$$\frac{\partial \bar{g}}{\partial x} = \frac{\partial \bar{g}}{\partial g}\frac{\partial g}{\partial x} \tag{6.16a}$$

$$= S_g \frac{\partial g}{\partial x}, \tag{6.16b}$$

$$\frac{\partial \bar{g}}{\partial \bar{y}} = \frac{\partial \bar{g}}{\partial g}\frac{\partial g}{\partial y}\frac{\partial y}{\partial \bar{y}} \tag{6.16c}$$

$$= S_g \frac{\partial g}{\partial y} S_y, \tag{6.16d}$$

and the Jacobian of the state drift function, as

$$\frac{\partial f}{\partial y} = \frac{\partial f}{\partial y}\frac{\partial y}{\partial \bar{y}} = \frac{\partial f}{\partial y} S_y. \tag{6.17}$$

We introduce the linear scaling and parametrisation defined in (6.16) and (6.17) and define the scaled residual

$$\bar{R}(z_{k,n+1}) = \begin{bmatrix} x_{k,n+1} - x_{k,n} - f_{k,n+1}\Delta t_{k,n} - \sigma_{k,n}\Delta\omega_{k,n} \\ \bar{g}_{k,n+1} \end{bmatrix} \tag{6.18a}$$

$$= \begin{bmatrix} x_{k,n+1} - x_{k,n} - f_{k,n+1}\Delta t_{k,n} - \sigma_{k,n}\Delta\omega_{k,n} \\ S_g g_{k,n+1} \end{bmatrix}. \tag{6.18b}$$

and residual Jacobian

$$\frac{\partial \bar{R}}{\partial z}(z_{k,n+1}) = \begin{bmatrix} I - \frac{\partial f_{k,n+1}}{\partial x}\Delta t_{k,n} & -\frac{\partial f_{k,n+1}}{\partial \bar{y}}\Delta t_{k,n} \\ \frac{\partial \bar{g}_{k,n+1}}{\partial x} & \frac{\partial \bar{g}_{k,n+1}}{\partial \bar{y}} \end{bmatrix} \tag{6.19a}$$

$$= \begin{bmatrix} I - \frac{\partial f_{k,n+1}}{\partial x}\Delta t_{k,n} & -\frac{\partial f_{k,n+1}}{\partial y} S_y \Delta t_{k,n} \\ S_g \frac{\partial g_{k,n+1}}{\partial x} & S_g \frac{\partial g_{k,n+1}}{\partial y} S_y \end{bmatrix}. \tag{6.19b}$$

Appropriate choice of scaling parameters, s_g and s_y, improves conditioning in the residual Jacobian.

6.3 Summary

In this chapter, we presented methods for numerical integration of SDAEs. We presented an implicit-explicit numerical integration scheme for systems of SDAEs. This integration scheme implicitly defines the solutions to the state and algebraic variables in the next time-step as roots in a residual function. We applied Newton's method to iteratively find roots in the residual function. The algebraic variables can result in ill-conditioning in the Jacobian of the residual, which may cause Newton's method to diverge. To address this, we introduced a linear scaling for the algebraic function and linear parametrisation of the algebraic variables to improve conditioning in the Jacobian.

Part II

Estimation Algorithms

CHAPTER 7 State Estimation for Nonlinear Systems involving Stochastic Differential Equations

The work presented in this chapter is based on the publication listed in Appendix B. In this chapter, we present methods for state estimation in continuous-discrete nonlinear systems involving SDEs. We consider SDE models in the form

$$d\boldsymbol{x}(t) = f(\boldsymbol{x}(t), u(t), \theta)dt + \sigma(\boldsymbol{x}(t), u(t), \theta)d\boldsymbol{\omega}(t), \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \tag{7.1a}$$

$$\boldsymbol{y}^m(t_k) = h^m(\boldsymbol{x}(t_k), \theta) + \boldsymbol{v}(t_k, \theta), \tag{7.1b}$$

where $\boldsymbol{x}(t)$ are state variables, $u(t)$ are manipulated variables, $\boldsymbol{y}^m(t_k)$ are measurement variables, and θ are model parameters. The process noise, $\boldsymbol{\omega}(t)$, is a standard Wiener process, i.e. $d\boldsymbol{\omega}(t) \sim \mathcal{N}(0, I dt)$ and $\boldsymbol{v}(t_k, \theta) \sim \mathcal{N}(0, R(\theta))$ is normally distributed measurement noise.

The objective of state estimation is to use noisy observations from a physical system to infer states in the system which are not, or cannot, be measured directly. State estimation is widely applied in industry for process monitoring [94], fault-detection [95–97], and advanced process control [98, 99]. Methods for state estimation can be separated into optimisation-based methods, e.g. least-squares and moving horizon estimation as described by [100], and methods based on statistical inference, i.e. Bayesian approaches such as Kalman filters as originally described in [41, 42]. In this work, we consider Bayesian approaches. A significant advantage of statistical methods is that they provide estimates of both the first and second (and sometimes more) moments of a system, thus describing both the expected state of the system, as well as a measure of the uncertainty of that estimate. We separate the methods into two steps; the time update and measurement update. In the time update, we apply the knowledge of the state dynamics to propagate the estimate of the state forward in time, including the uncertainty, i.e. prediction. In the measurement update, we take observations of the physical systems and apply statistical inference to update the state estimate, i.e. filtering. We consider four state estimation methods for systems in the form (7.1); the continuous-discrete extended Kalman filter (CD-EKF), unscented Kalman filter (CD-UKF), ensemble Kalman filter (CD-EnKF), and a particle filter (CD-PF). As mentioned above, the methods are separated into two steps; the time and measurement updates. In the time update, at time t_k, we compute predicted estimates of state expectation and covariance at time t_{k+1}, i.e. $\hat{x}_{k+1|k}$ and $P_{k+1|k}$, respectively. In the measurement update, we obtain a measurement from the system at time t_k, y_k, and apply it to update the estimates of state expectation and covariance, i.e. compute the filtered estimates of state mean and covariance $\hat{x}_{k|k}$ and $P_{k|k}$.

This chapter is structured as follows: in section 7.1, we present the CD-EKF which directly extends the update from the linear Kalman filter to a local linearisation of a nonlinear system. In section 7.2, we present the CD-UKF which approximates the nonlinear state and measurement distributions with a set of deterministically sampled particles. Section 7.3 presents the CD-EnKF, which approximates the nonlinear state and measurement distributions with a set of randomly sampled particles and apply the Kalman update to each particle. In section 7.4, we present a CD-PF which approximates the nonlinear state and measurement distributions with a set of randomly sampled particles and applies likelihood resampling to filter the set. Section 7.5 presents a discussion of the four different state estimation algorithms presented in the chapter. Finally, section 7.6 summarises the work presented in the chapter.

7.1 Continuous-discrete extended Kalman filter

In this section, we consider the CD-EKF for systems in the form (7.1). As mentioned previously, the CD-EKF is a state estimation algorithm which directly extends the update equations of the linear Kalman filter [41]. Specifically, the CD-EKF applies the prediction and filtering strategies for the linear Kalman filter on a local linearisation of the nonlinear state and measurement dynamics. In this section, we describe the initialisation of the CD-EKF, as well as time and measurement update. Figure 7.1 illustrates the update strategy for the CD-EKF.

7.1.1 Initialisation

We initialise the CD-EKF with the initial guesses for the state mean and covariance

$$\hat{x}_{0|0} \in \mathbb{R}^{n_x}, \qquad P_{0|0} \in \mathbb{R}^{n_x \times n_x}. \tag{7.2}$$

The initialisation of the CD-EKF is a measure of the prior knowledge of the system and it is of crucial importance to the performance of the state estimator. For instance, if the true state of the system is very unlikely in the initial distribution to the estimator, the filter may trust model predictions to a too large extend and not converge - even if the assumption of local linearity holds [51, 101].

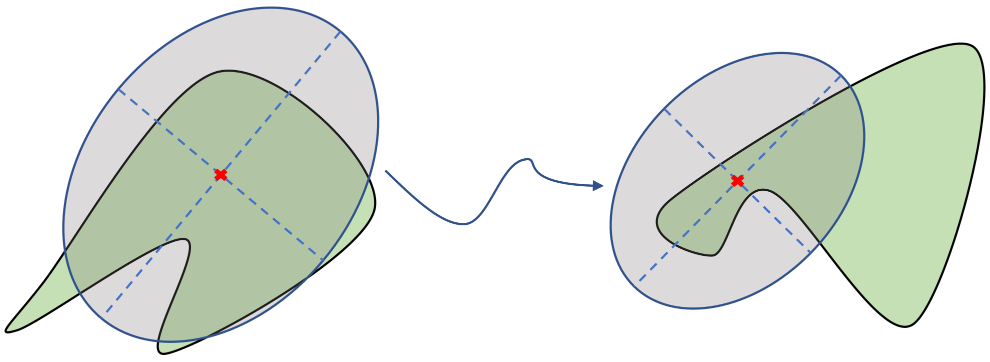


Figure 7.1: Illustration of the update for the extended Kalman filter. The true nonlinear distribution, green underlying to the left, is characterised by it's first two moments, i.e. mean and covariance. The state estimates of mean and covariance are computed by applying the Kalman filter update to a local linearisation of the nonlinear system. The true nonlinear distribution, green underlying to the right, is approximated by the first two moments propagated under the assumption of local linearity.

7.1.2 Time update

In the time update, we compute the one-step predictions of the state mean and covariance. We compute the predicted state estimate as the solution to the expectation of the stochastic state dynamics, i.e. an ODE because (7.1) is a martingale, for $t \in [t_k, t_{k+1}]$, as

$$\frac{d\hat{x}_k}{dt}(t) = f(\hat{x}_k(t), u(t), \theta), \qquad \hat{x}_k(t_k) = \hat{x}_{k|k}, \tag{7.3a}$$

$$\frac{dP_k}{dt}(t) = A_k(t)P_k(t) + P_k(t)A_k^T(t) + \sigma_k(t)\sigma_k^T(t), \qquad P_k(t_k) = P_{k|k}, \tag{7.3b}$$

where

$$A_k(t) = \frac{\partial f}{\partial x}(\hat{x}_k(t), u(t), \theta), \qquad \sigma_k(t) = \sigma(\hat{x}_k(t), u(t), \theta). \tag{7.4}$$

The predicted state mean and covariance estimates are the solutions to (7.3) at time $t = t_{k+1}$

$$\hat{x}_{k+1|k} = \hat{x}_k(t_{k+1}), \qquad P_{k+1|k} = P_k(t_{k+1}). \tag{7.5}$$

Alternatively, we may compute the predicted state covariance estimate by the integral expression [102]. We compute the predicted state covariance estimate as the solution at time $t = t_{k+1}$ of the integral expression

$$P_k(t) = \Phi_{xx}(t, t_k) P_k(t_k) \Phi_{xx}^T(t, t_k) + \int_{t_k}^{t} \Phi_{xx}(t, s) \sigma_k(s) \sigma_k^T(s) \Phi_{xx}^T(t, s) ds, \tag{7.6}$$

where the sensitivities are

$$\frac{d\Phi_{xx}(t, s)}{dt} = A_k(t) \Phi_{xx}(t, s), \qquad \Phi_{xx}(s, s) = I. \tag{7.7}$$

7.1.3 Measurement update

In the measurement update, we apply the filtering strategy in the linear Kalman filter on a local linearisation of the nonlinear system. We compute the innovation and its covariance, as

$$e_k = y_k^m - \hat{y}_{k|k-1}^m, \tag{7.8a}$$

$$R_{e,k} = C_k P_{k|k-1} C_k^T + R, \tag{7.8b}$$

where y_k^m is a measurement taken at time t_k and

$$\hat{y}_{k|k-1}^m = h^m(\hat{x}_{k|k-1}, \theta), \tag{7.9a}$$

$$C_k = \frac{\partial h^m}{\partial x}(\hat{x}_{k|k-1}, \theta). \tag{7.9b}$$

We compute the Kalman gain,

$$K_{f_x,k} = P_{k|k-1} C_k^T R_{e,k}^{-1}, \tag{7.10}$$

and we compute the filtered state mean and covariance estimates,

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_{f_x,k} e_k, \tag{7.11a}$$

$$P_{k|k} = P_{k|k-1} - K_{f_x,k} R_{e,k} K_{f_x,k}^T, \tag{7.11b}$$

$$= \left(I - K_{f_x,k} C_k\right) P_{k|k-1} \left(I - K_{f_x,k} C_k\right)^T + K_{f_x,k} R K_{f_x,k}^T, \tag{7.11c}$$

where (7.11c), so-called Joseph stabilising form, is for numerically stability.

7.2 Continuous-discrete unscented Kalman filter

In this section, we present CD-UKF for systems in the form (7.1). As mentioned previously, the CD-UKF is a particle filter in which a set of deterministically sampled particles, so-called sigma-points, represent the nonlinear state and measurement distributions. The filter applies a so-called unscented transformation [55]. In the time update, the sigma-points are propagated through the nonlinear system dynamics and the predicted state mean and covariance estimates are computed statistically from the sigma-points. In the measurement update, we apply the Kalman filter update to the sigma-points. We computed the filtered state mean and covariance estimates statistically from the filtered sigma-points. Figure 7.2 illustrates the CD-UKF.

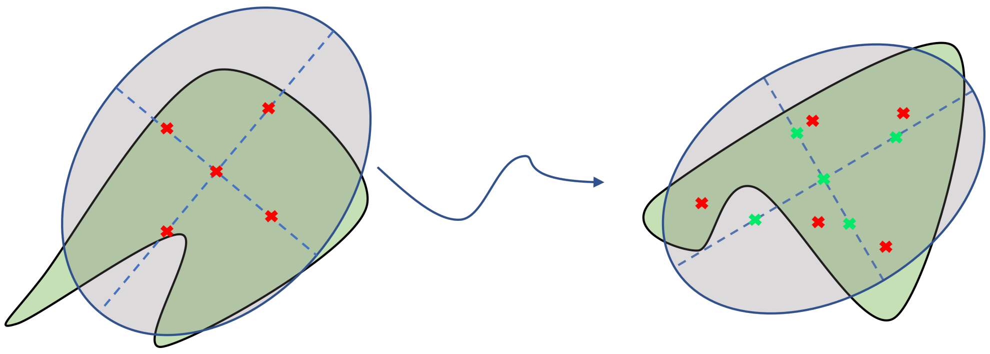


Figure 7.2: Illustration of the update for the unscented Kalman filter. The true nonlinear distribution, green underlying to the left, is characterised by it's first two moments, i.e. mean and covariance, and described by a set of particles (sigma-points) deterministically drawn from the distribution (unscented transformation). The particles are propagated through the nonlinear state dynamics. The true nonlinear distribution, green underlying to the right, is characterised by the approximations of the expectation and covariance determined statistically from the propagated particles.

7.2.1 Initialisation

We initialise the CD-UKF with the initial guesses for the state mean and covariance

$$\hat{x}_{0|0} \in \mathbb{R}^{n_x}, \qquad P_{0|0} \in \mathbb{R}^{n_x \times n_x}. \tag{7.12}$$

The initialisation of the CD-UKF is a measure of the prior knowledge of the system and it is therefore of crucial importance to the performance of the state estimator. The initial value of the covariance matrix in large part determines the spread of the sampled sigma-points. Consequently, the covariance also influences the quality of the state estimates by influencing both the bias in the initial approximation of the distribution, i.e. the sigma-points, as well as the emphasis given to observations in the measurement update.

7.2.2 Time update

In the time update, we compute the parameters

$$\bar{c} = \alpha^2 (\bar{n} + \kappa), \tag{7.13a}$$
$$\bar{\lambda} = \alpha^2 (\bar{n} + \kappa) - \bar{n}, \tag{7.13b}$$

where $\alpha \in]0,1]$ and $\kappa \in [0,\infty[$ are tuning parameters describing the spread and bias in the sampling of the sigma-points, and $\bar{n} = n_x + n_\omega$. We compute the weights associated with the sigma-point distribution

$$\bar{W}_m^{(0)} = \frac{\bar{\lambda}}{\bar{n}+\bar{\lambda}}, \tag{7.14a}$$

$$\bar{W}_c^{(0)} = \frac{\bar{\lambda}}{\bar{n}+\bar{\lambda}} + 1 - \alpha^2 + \beta, \tag{7.14b}$$

$$\bar{W}_m^{(i)} = \bar{W}_c^{(i)} = \frac{1}{2\left(\bar{n}+\bar{\lambda}\right)}, \tag{7.14c}$$

for $i \in \{1,2,\dots,2\bar{n}\}$, and where $\beta \in [0,\infty[$ is a tuning parameter associated with the state distribution ($\beta = 2$ is optimal for a Gaussian distribution). We deterministically sample a set of $2\bar{n}+1$ sigma-points from the state distribution. The sigma-points are separated into two sets. We define the set of $2n_x + 1$ sigma-points

$$\hat{x}_{k|k}^{(0)} = \hat{x}_{k|k}, \tag{7.15a}$$

$$\hat{x}_{k|k}^{(i)} = \hat{x}_{k|k} + \sqrt{c}\left(\sqrt{P_{k|k}}\right)_i, \tag{7.15b}$$

$$\hat{x}_{k|k}^{(n_x+i)} = \hat{x}_{k|k} - \sqrt{c}\left(\sqrt{P_{k|k}}\right)_i, \tag{7.15c}$$

for $i \in \{1,2,\dots,n_x\}$ and denote them the deterministic set. In (7.15), $\left(\sqrt{P_{k|k}}\right)_i$ denotes the i'th column in the Cholesky decomposition of the covariance matrix, $P_{k|k}$. Alternatively, we can compute the sigma-points in the directions of the eigenvectors [103]. Furthermore, we define the set of $2n_\omega$ sigma-points

$$\hat{x}_{k|k}^{(2n_x+i)} = \hat{x}_{k|k}, \tag{7.16}$$

for $i \in \{1,2,\dots,2n_\omega\}$ and denote them the stochastic set. Additionally, we compute the process noise

$$d\omega_k^{(2n_x+i)}(t) = \sqrt{c\ dt}\,(I)_i\,, \tag{7.17a}$$

$$d\omega_k^{(2n_x+n_\omega+i)}(t) = -\sqrt{c\ dt}\,(I)_i\,, \tag{7.17b}$$

where $i \in \{1,2,\dots,n_\omega\}$ and $(I)_i$ denotes the i'th column of an identity matrix of size $n_\omega \times n_\omega$. We propagate the deterministic set of sigma-points through the deterministic dynamics for $t \in [t_k, t_{k+1}]$ and compute the predictions as the solution to

$$d\hat{x}_k^{(i)}(t) = f(\hat{x}_k^{(i)}, u(t), \theta)dt, \qquad \hat{x}_k^{(i)}(t_k) = \hat{x}_{k|k}^{(i)}, \tag{7.18}$$

for $i \in \{0,1,\dots,2n_x\}$. We propagate the stochastic set of sigma-points through the stochastic dynamics for $t \in [t_k, t_{k+1}]$ and compute the predictions as the solution to

$$\hat{x}_k^{(2n_x+i)}(t_k) = \hat{x}_{k|k}^{(2n_x+i)}, \tag{7.19a}$$

$$d\hat{x}_k^{(2n_x+i)}(t) = f(\hat{x}_k^{(2n_x+i)}(t), u(t), \theta)dt + \sigma(\hat{x}_k^{(2n_x+i)}(t), u(t), \theta)d\omega_k^{(2n_x+i)}(t), \tag{7.19b}$$

for $i \in \{1, 2, \ldots, 2n_\omega\}$. Solving the initial value problems (7.18) and (7.19), we arrive at the predicted sigma-points

$$\hat{x}_{k+1|k}^{(i)} = \hat{x}_k^{(i)}(t_{k+1}), \qquad i \in \{1, 2, \ldots, 2\bar{n}\}. \tag{7.20}$$

The predicted state mean and covariance estimates are computed statistically from the predicted sigma-points as

$$\hat{x}_{k+1|k} = \sum_{i=0}^{2\bar{n}} \bar{W}_m^{(i)} \hat{x}_{k+1|k}^{(i)}, \tag{7.21a}$$

$$P_{k+1|k} = \sum_{i=0}^{2\bar{n}} \bar{W}_c^{(i)} \left(\hat{x}_{k+1|k}^{(i)} - \hat{x}_{k+1|k}\right) \left(\hat{x}_{k+1|k}^{(i)} - \hat{x}_{k+1|k}\right)^T. \tag{7.21b}$$

7.2.3 Measurement update

In the measurement update, we compute a set of sigma-points which we propagate through the measurement dynamics. We apply the Kalman filter update to each sigma-point and compute the state estimate statistically from the propagated particles. We compute the parameters

$$c = \alpha^2 (n_x + \kappa), \tag{7.22a}$$

$$\lambda = \alpha^2 (n_x + \kappa) - n_x, \tag{7.22b}$$

and we compute the weights associated with the sigma-points

$$W_m^{(0)} = \frac{\lambda}{n_x + \lambda}, \tag{7.23a}$$

$$W_c^{(0)} = \frac{\lambda}{n_x + \lambda} + 1 - \alpha^2 + \beta, \tag{7.23b}$$

$$W_m^{(i)} = W_c^{(i)} = \frac{1}{2(n_x + \lambda)}, \tag{7.23c}$$

for $i \in \{1, 2, \ldots, 2n_x\}$. We compute a set of $2n_x + 1$ sigma-points

$$\hat{x}_{k|k-1}^{(0)} = \hat{x}_{k|k-1}, \tag{7.24a}$$

$$\hat{x}_{k|k-1}^{(i)} = \hat{x}_{k|k-1} + \sqrt{c}\left(\sqrt{P_{k|k-1}}\right)_i, \tag{7.24b}$$

$$\hat{x}_{k|k-1}^{(n_x+i)} = \hat{x}_{k|k-1} - \sqrt{c}\left(\sqrt{P_{k|k-1}}\right)_i, \tag{7.24c}$$

for $i \in \{1, 2, \ldots, n_x\}$. Similar to the time update, we may alternatively computed the sigma-points in the directions of the eigenvectors [103]. We compute the innovation as

$$e_k = y_k^m - \hat{y}_{k|k-1}^m, \tag{7.25}$$

where y_k^m is a measurement taken at time t_k and

$$\hat{y}_{k|k-1}^m = \hat{z}_{k|k-1}^m = \sum_{i=0}^{2n_x} W_m^{(i)} \hat{z}_{k|k-1}^{m,(i)}, \qquad \hat{z}_{k|k-1}^{m,(i)} = h^m(\hat{x}_{k|k-1}^{(i)}, \theta). \tag{7.26}$$

We compute the covariance and cross-covariance information from the sigma-points

$$R_{zz,k|k-1} = \sum_{i=0}^{2n_x} W_c^{(i)} \left(\hat{z}_{k|k-1}^{m,(i)} - \hat{z}_{k|k-1}^m\right) \left(\hat{z}_{k|k-1}^{m,(i)} - \hat{z}_{k|k-1}^m\right)^T, \tag{7.27a}$$

$$R_{e,k} = R_{yy,k|k-1} = R_{zz,k|k-1} + R, \tag{7.27b}$$

$$R_{xy,k|k-1} = \sum_{i=0}^{2n_x} W_c^{(i)} \left(\hat{x}_{k|k-1}^{(i)} - \hat{x}_{k|k-1}\right) \left(\hat{z}_{k|k-1}^{m,(i)} - \hat{z}_{k|k-1}^m\right)^T, \tag{7.27c}$$

and we compute the Kalman gain as

$$K_{f_x,k} = R_{xy,k|k-1} R_{e,k}^{-1}. \tag{7.28}$$

The filtered mean and covariance estimates are computed as

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_{f_x,k} e_k, \tag{7.29a}$$

$$P_{k|k} = P_{k|k-1} - K_{f_x,k} R_{e,k} K_{f_x,k}^T. \tag{7.29b}$$

Similarly to the CD-EKF, square-root variations of the CD-UKF have been formulated to improve numerical stability [104].

7.3 Continuous-discrete ensemble Kalman filter

In this section, we present the CD-EnKF for systems in the form (7.1). As mentioned previously, the CD-EnKF is a particle filter which approximates the nonlinear state and measurement distributions with a set of randomly sampled particles, the so-called ensemble. In the time-update, we propagate each particle in the ensemble through the stochastic state dynamics. We compute the state estimates statistically from the predicted ensemble. In the measurement update, a measurement ensemble is computed by propagating the state ensemble through the nonlinear measurement dynamics. For each particle, we draw a measurement from a measurement distribution defined from a system measurement. We update each particle in the state ensemble by applying the Kalman filter equations to pairs predicted measurements from the ensemble and sampled measurement. We compute the filtered state estimate statistically from the filtered state ensemble.

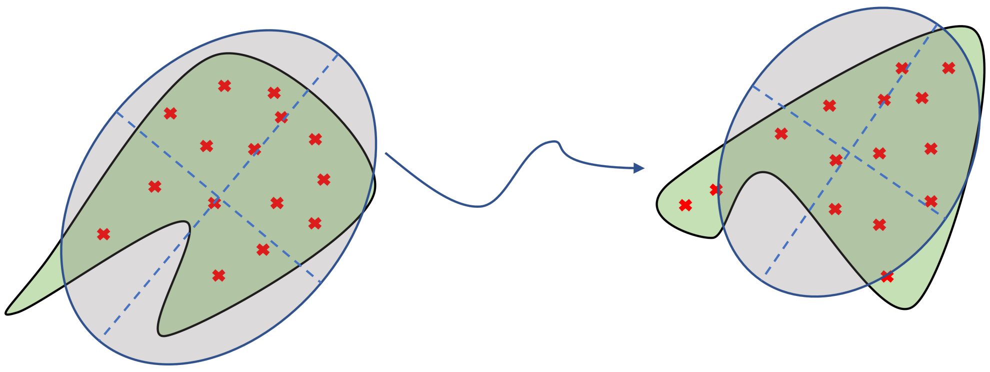


Figure 7.3: Illustration of the update for the ensemble Kalman filter. The true nonlinear distribution, green underlying to the left, is characterised by it's first two moments, i.e. mean and covariance, and described by a set of particles randomly sampled from the prior distribution (or approximation). The particles are propagated through the nonlinear state dynamics. The true nonlinear distribution, green underlying to the right, is characterised by the approximations of the expectation and covariance determined statistically from the propagated particles..

7.3.1 Initialisation

We initialise the CD-EnKF by randomly sampling a set of particles, the ensemble, from the prior state distribution, i.e. the initial distribution. We denote the initial ensemble

$$\{\hat{x}_{0|0}^{(i)}\}_{i=1}^{N_p}. \tag{7.30}$$

The initial state distribution serves as the initial guess for the CD-EnKF since the true nonlinear state distribution is usually unknown. A Gaussian distribution with guesses on mean and covariance, similar to the CD-EKF and CD-UKF, may serve as an initial guess to sample from.

7.3.2 Time update

In the time update, we propagate each particle in the ensemble through the system dynamics. We compute the prediction ensemble as the solutions to the initial value problem

$$d\boldsymbol{x}_k^{(i)}(t) = f(\boldsymbol{x}_k^{(i)}(t), u(t), \theta)dt + \sigma(\boldsymbol{x}_k^{(i)}(t), u(t), \theta)d\boldsymbol{\omega}_k(t), \quad x_k^{(i)}(t_k) = \hat{x}_{k|k}^{(i)}, \tag{7.31}$$

for $i \in \{1, 2, \dots, N_p\}$ and $t \in [t_k, t_{k+1}]$. From the predictions, we denote the predicted state ensemble

$$\{\hat{x}_{k+1|k}^{(i)}\}_{i=1}^{N_p}, \tag{7.32}$$

where $\hat{x}_{k+1|k}^{(i)} = x_k^{(i)}(t_{k+1})$ are the solutions to (7.31). We compute the predicted state mean and covariance estimates statistically from the predicted state ensemble, as

$$\hat{x}_{k+1|k} = \frac{1}{N_p} \sum_{i=1}^{N_p} \hat{x}_{k+1|k}^{(i)}, \tag{7.33a}$$

$$P_{k+1|k} = \frac{1}{N_p - 1} \sum_{i=1}^{N_p} \left(\hat{x}_{k+1|k}^{(i)} - \hat{x}_{k+1|k}\right) \left(\hat{x}_{k+1|k}^{(i)} - \hat{x}_{k+1|k}\right)^T. \tag{7.33b}$$

7.3.3 Measurement update

In the measurement update, we compute the predicted measurement ensemble by propagating the predicted state ensemble through the measurement dynamics. We apply the Kalman filter update to each particle in the ensemble by sampling from a measurement distribution defined by a system measurement, y_k^m. We compute the ensemble of predictions

$$\{\hat{z}_{k|k-1}^{m,(i)}\}_{i=1}^{N_p}, \qquad z_{k|k-1}^{m,(i)} = h^m(\hat{x}_{k|k-1}^{(i)}, \theta). \tag{7.34}$$

We compute the predicted mean and covariance estimates of the measurement distribution and cross-covariance of states and measurements, as

$$\hat{y}_{k|k-1}^m = \hat{z}_{k|k-1}^m = \frac{1}{N_p} \sum_{i=1}^{N_p} \hat{z}_{k|k-1}^{m,(i)}, \tag{7.35a}$$

$$R_{zz,k|k-1} = \frac{1}{N_p - 1} \sum_{i=1}^{N_p} \left(\hat{z}_{k|k-1}^{m,(i)} - \hat{z}_{k|k-1}^m\right) \left(\hat{z}_{k|k-1}^{m,(i)} - \hat{z}_{k|k-1}^m\right)^T, \tag{7.35b}$$

$$R_{yy,k|k-1} = R_{zz,k|k-1} + R, \tag{7.35c}$$

$$R_{xy,k|k-1} = \frac{1}{N_p - 1} \sum_{i=1}^{N_p} \left(\hat{x}_{k|k-1}^{(i)} - \hat{x}_{k|k-1}\right) \left(\hat{y}_{k|k-1}^{m,(i)} - \hat{y}_{k|k-1}^m\right)^T. \tag{7.35d}$$

We compute samples from the measurement distribution, as

$$y_k^{m,(i)} = y_k^m + v_k^{(i)}, \tag{7.36}$$

where $v_k^{(i)}$ are realisations of the measurement noise distribution, $\boldsymbol{v}_k \sim \mathcal{N}(0, R(\theta))$, and y_k^m is a system measurement taken at time t_k. We compute the innovations for each particle in the measurement ensemble, as

$$e_k^{(i)} = y_k^{m,(i)} - \hat{z}_{k|k-1}^{m,(i)}, \qquad i \in \{1, 2, \dots, N_p\}. \tag{7.37}$$

We compute the Kalman gain as

$$K_{f_x,k} = R_{xy,k|k-1} R_{yy,k|k-1}^{-1}, \tag{7.38}$$

and compute the filtered state ensemble

$$\{\hat{x}_{k|k}^{(i)}\}_{i=1}^{N_p}, \tag{7.39}$$

where we apply the Kalman filter update to each particle in the ensemble as

$$\hat{x}_{k|k}^{(i)} = \hat{x}_{k|k-1}^{(i)} + K_{f_x,k} e_k^{(i)}. \tag{7.40}$$

We compute the filtered state mean and covariance estimates statistically from the filtered state ensemble as

$$\hat{x}_{k|k} = \frac{1}{N_p} \sum_{i=1}^{N_p} \hat{x}_{k|k}^{(i)}, \tag{7.41a}$$

$$P_{k|k} = \frac{1}{N_p - 1} \sum_{i=1}^{N_p} \left(\hat{x}_{k|k}^{(i)} - \hat{x}_{k|k}\right) \left(\hat{x}_{k|k}^{(i)} - \hat{x}_{k|k}\right)^T. \tag{7.41b}$$

7.4 Continuous-discrete particle filter

In this section, we present a CD-PF for systems in the form (7.1). As mentioned previously, the CD-PF is a particle filter which approximates the state and measurement distributions by a set of randomly sampled particles, the particle set. In the time update, we perform a stochastic simulation of each particle in the particle set. We represent the predicted state distribution by the propagated particle set. We compute the predicted state mean and covariance estimates statistically from the particle set. As such, the time update for the CD-PF is identical to that of the CD-EnKF. In the measurement update, we propagate the particle set through the measurement dynamics and determine the likelihood of each particle being observed, i.e. the likelihood of sampling the measurement particle from the measurement distribution. We then resample the state particle set according to the likelihood. Likely particles may thus appear several time and unlikely particles may be removed entirely from the resampled particle set. The filtered state distribution is represented by the resampled (filtered) particle set. We compute the filtered state mean and covariance statistically from the filtered particle set.

7.4.1 Initialisation

We initialise the CD-PF by randomly sampling a set of particles from the prior state distribution, i.e. the initial distribution. We denote the initial particle set

$$\{\hat{x}_{0|0}^{(i)}\}_{i=1}^{N_p}. \tag{7.42}$$

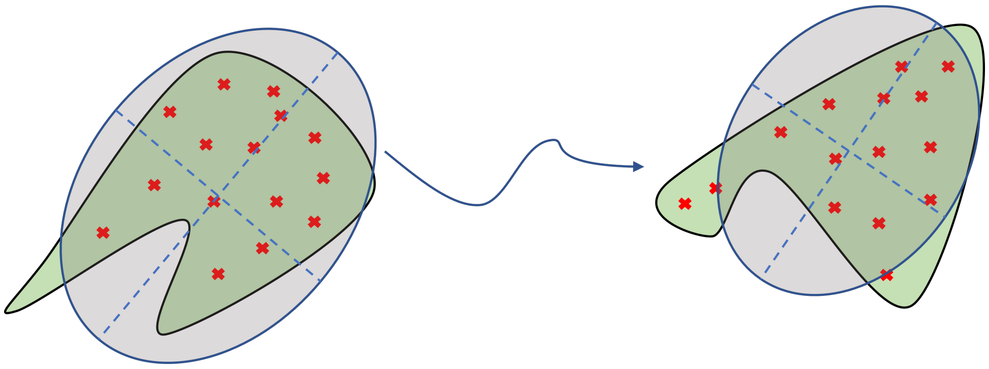


Figure 7.4: Illustration of the update for the particle filter. The true nonlinear distribution, green underlying to the left, is characterised by it's first two moments, i.e. mean and covariance, and described by a set of particles sampled from the prior distribution (or approximation). The particles are propagated through the nonlinear state dynamics. The true nonlinear distribution, green underlying to the right, is characterised by the approximations of the expectation and covariance determined statistically from the propagated particles..

The considerations for the CD-PF are the same as for the CD-EnKF. Since the true initial state distribution may be unknown, the initial distribution serves as the initial guess. A Gaussian distribution with guesses on mean and covariance, similar to the CD-EKF and CD-UKF, may serve as an initial distribution to sample from.

7.4.2 Time update

In the time update, we propagate each particle in the particle set through the stochastic state dynamics. We compute The predicted particle set as the solutions to the initial value problem

$$d\boldsymbol{x}_k^{(i)}(t) = f(\boldsymbol{x}_k^{(i)}(t), u(t), \theta)dt + \sigma(\boldsymbol{x}_k^{(i)}(t), u(t), \theta)d\boldsymbol{\omega}_k(t), \quad x_k^{(i)}(t_k) = \hat{x}_{k|k}^{(i)}, \tag{7.43}$$

for $i \in \{1, 2, \dots, N_p\}$ and $t \in [t_k, t_{k+1}]$. From the predictions, we denote the predicted particle set

$$\{\hat{x}_{k+1|k}^{(i)}\}_{i=1}^{N_p}, \tag{7.44}$$

where $\hat{x}_{k+1|k}^{(i)} = x_k^{(i)}(t_{k+1})$ are the solutions to (7.43). We compute the predicted state mean and covariance estimates statistically from the predicted state particles, as

$$\hat{x}_{k+1|k} = \frac{1}{N_p}\sum_{i=1}^{N_p} \hat{x}_{k+1|k}^{(i)}, \tag{7.45a}$$

$$P_{k+1|k} = \frac{1}{N_p - 1}\sum_{i=1}^{N_p} \left(\hat{x}_{k+1|k}^{(i)} - \hat{x}_{k+1|k}\right)\left(\hat{x}_{k+1|k}^{(i)} - \hat{x}_{k+1|k}\right)^T. \tag{7.45b}$$

7.4.3 Measurement update

In the measurement update, we compute the set of predicted measurement particles by propagating the predicted particle set through the measurement dynamics. We apply a measurement from the system to compute the likelihood for each particle in the measurement set. We resample the state particles according to their respective likelihoods in the measurement distribution. The resampled set is denoted the filtered particle set. We compute the filtered state mean and covariance estimates statistically from the filtered state particle set. We compute the set of particles from the predicted measurement distribution

$$\{\hat{z}_{k|k-1}^{m,(i)}\}_{i=1}^{N_p}, \qquad \hat{z}_{k|k-1}^{m,(i)} = h^m(\hat{x}_{k|k-1}^{(i)}, \theta), \tag{7.46}$$

and compute the innovations from each measurement particle, as

$$e_k^{(i)} = y_k^m - \hat{z}_{k|k-1}^{m,(i)}, \tag{7.47}$$

for $i \in \{1, 2, \ldots, N_p\}$. We compute a set of likelihood weights for each particle, as the likelihood of sampling each particle from the measurement distribution, as

$$\tilde{w}_k^{(i)} = \frac{1}{\sqrt{2\pi^{n_y}\,|R(\theta)|}} \exp\left(-\frac{1}{2}\left(e_k^{(i)}\right)^T R(\theta)^{-1}\left(e_k^{(i)}\right)\right), \tag{7.48}$$

where $|R(\theta)|$ denotes the determinant of the measurement covariance matrix, $R(\theta)$. The likelihood weight are normalised, as

$$w_k^{(i)} = \frac{\tilde{w}_k^{(i)}}{\sum_{j=1}^{N_p} \tilde{w}_k^{(j)}}, \tag{7.49}$$

for $i \in \{1, 2, \ldots, N_p\}$. We resample the particles according to their normalised likelihood weights. For a single realisation of a uniform distribution, $q_1 \sim \mathcal{U}[0,1]$, we compute a set of ordered resampling points

$$q_k^{(i)} = \frac{(i-1) + q_1}{N_p}, \tag{7.50}$$

for $i \in \{1, 2, \dots, N_p\}$. We resample the particles by storing $m^{(i)}$ copies of each particle, $\hat{x}^{(i)}_{k|k-1}$, in the set. The indices for the resampled particles, l, are chosen such that $q^{(l)}_k \in \left]s^{(i-1)}, s^{(i)}\right]$, where $s^{(i)} = \sum_{j=1}^{i} w^{(j)}_k$. This resampling means that particles with relatively high likelihood may appear several times, while particle with relatively low likelihood may disappear. The set of resampled particles (the filtered particle set) is denoted

$$\{\hat{x}^{(i)}_{k|k}\}_{i=1}^{N_p}. \tag{7.51}$$

The filtered state estimate is computed statistically from the filtered particle set, as

$$\hat{x}_{k|k} = \frac{1}{N_p} \sum_{i=1}^{N_p} \hat{x}^{(i)}_{k|k}, \tag{7.52a}$$

$$P_{k|k} = \frac{1}{N_p - 1} \sum_{i=1}^{N_p} \left(\hat{x}^{(i)}_{k|k} - \hat{x}_{k|k}\right) \left(\hat{x}^{(i)}_{k|k} - \hat{x}_{k|k}\right)^T. \tag{7.52b}$$

7.5 Discussion

In this section, we discuss the different state estimation methods presented in this chapter.

CD-EKF

As mentioned previously, the CD-EKF is essentially the application of the linear Kalman filter to a local linearisation of a nonlinear system. As a result of this, the performance of the CD-EKF is largely dictated by the error associated with the linearisation. In the time update, the linearisation error is dependent on the discretisation interval. As such, the error introduced in the time update can be reduced by increasing the internal linearisation points between samples. The error introduced in the measurement update can cause the CD-EKF to diverge for highly nonlinear systems. The CD-EKF is a computationally efficient state estimation algorithm in most applications. The computational efficiency is largely dictated by the covariance update in the time update. For non-stiff systems, i.e. applying explicit numerical integration schemes, a disadvantage of the CD-EKF is that it requires the computation of the Jacobian. For stiff system, i.e. applying implicit numerical integration schemes, the numerical integration also requires the computation of the Jacobian, so the CD-EKF does not present increased complexity in that regard.

CD-UKF

Like the CD-EKF, the CD-UKF is also a computationally efficient algorithm. The nonlinear distributions are represented by a set of deterministically sampled particles. The particles are propagated through the true nonlinear state and measurement dynamics. The CD-UKF has higher order convergence than the CD-EKF. The particle representation means that it better represents the nonlinearities of the state and measurement distributions. This introduces statistical error, but removes the error associated with linearisation. The CD-UKF better represents nonlinearities in the state and measurement distributions then the CD-EKF, but can still fail for highly-nonlinear systems.

CD-EnKF

The CD-EnKF is a particle filter based on the filtering strategy of the Kalman filter. The CD-EnKF represents the state and measurement distributions with a set of randomly sampled particles. The CD-EnKF has Gaussian assumptions in the measurement update, but can be effectively applied to highly nonlinear systems in most cases. The CD-EnKF is a particle filter and thus suffers from issues with coverage for high-dimensional state distributions, i.e. the curse of dimensionality. This can make the CD-EnKF computationally inefficient for high-dimensional systems if great statistical accuracy is required.

CD-PF

The CD-PF is a particle filter which does not have assumptions on the distributions. This means that the CD-PF can be effectively applied to arbitrary nonlinear systems. It can be advantageous to apply the CD-PF for state estimation in systems, where the distributions are bounded and can thus not be assumed Gaussian with high accuracy. This means that the CD-PF has seen many applications in robotics, where the initial particle distribution may fill up a physical space, i.e. a potentially highly nonlinear distribution. As long as the likelihood can be computed, the CD-PF can be applied. Similar to the CD-EnKF, the CD-PF suffers from coverage issues for high-dimensional state distributions, i.e. the curse of dimensionality. This can make the CD-PF computationally inefficient for high-dimensional systems if great statistical accuracy is required.

7.6 Summary

In this chapter, we presented four methods of state estimation in nonlinear systems involving SDEs. We presented the CD-EKF which applies the Kalman filter updates to a local linearisation of the nonlinear system. We presented the CD-UKF which represent the nonlinear state and measurement distributions by a set of deterministically sampled particles, so-called sigma-points. We presented the two particle filters; the CD-EnKF and th CD-PF. The CD-EnKF applies the Kalman filter update to the ensemble of particles. The CD-PF applies likelihood resampling to the particle set to represent the filtered state distribution. The state estimates are determined statistically from the particle sets.

CHAPTER 8

State Estimation for Nonlinear Systems involving Stochastic Differential Algebraic Equations

In this chapter, we present the CD-EKF for state estimation in nonlinear systems involving SDAEs. We consider SDAE models in the form

$$d\boldsymbol{x}(t) = f(\boldsymbol{x}(t), \boldsymbol{y}(t), u(t), \theta)dt + \sigma(\boldsymbol{x}(t), \boldsymbol{y}(t), u(t), \theta)d\boldsymbol{\omega}(t), \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \tag{8.1a}$$

$$0 = g(\boldsymbol{x}(t), \boldsymbol{y}(t), \theta), \tag{8.1b}$$

$$\boldsymbol{y}^m(t_k) = h^m(\boldsymbol{x}(t_k), \boldsymbol{y}(t_k), \theta) + \boldsymbol{v}(t_k, \theta), \tag{8.1c}$$

where $\boldsymbol{x}(t)$ are state variables, $\boldsymbol{y}(t)$ are algebraic variables, $u(t)$ are manipulated variables, $\boldsymbol{y}^m(t_k)$ are measurement variables, and θ are model parameters. The process noise $\boldsymbol{\omega}(t)$ is a standard Wiener process, i.e. $d\boldsymbol{\omega}(t) \sim \mathcal{N}(0, Idt)$, and $\boldsymbol{v}(t_k, \theta) \sim \mathcal{N}(0, R(\theta))$ is the normally distributed measurement noise.

As mentioned in the introduction to Chapter 7, the CD-EKF is a direct extension of the original Kalman filter applied to nonlinear systems. The CD-EKF applies the Kalman filter update to a local linearisation of the nonlinear system and has been widely applied in industry due to its close relation to the linear Kalman filter. Formulating the CD-EKF for SDAEs makes it possible to include information from measurements which are governed by algebraic variables, e.g. chemical equilibrium reactions.

The chapter is structured as follows: in section 8, we present the CD-EKF for nonlinear systems involving SDAEs. We apply the implicit function theorem to express the algebraic variables as functions of the states. We formulate the CD-EKF by applying the sensitivities arising from the implicit description. Finally, we present a summary of the chapter in section 8.2.

8.1 Continuous-discrete extended Kalman filter

In this section, we consider the CD-EKF for systems in the form (8.1).

8.1.1 Initialisation

We initialise the CD-EKF with initial guesses for the state mean and covariance

$$\hat{x}_{0|0} \in \mathbb{R}^{n_x}, \qquad P_{0|0} \in \mathbb{R}^{n_x \times n_x}. \tag{8.2}$$

Given an initial state, we compute the algebraic variables at the initial time,

$$\hat{y}_{0|0} \in \mathbb{R}_{n_y}, \tag{8.3}$$

as the solution to the algebraic equations,

$$0 = g(\hat{x}_{0|0}, \hat{y}_{0|0}, \theta). \tag{8.4}$$

Given an initial state covariance, we compute the covariance of the algebraic variables at the initial time, as

$$P_{y,0|0} = \Phi_{yx} P_{0|0} \Phi_{yx}^T, \tag{8.5}$$

where Φ_{yx} denotes the sensitivities of the algebraic variables wrt. to the states. The sensitivities arise from the implicit function theorem as the solution to

$$\frac{\partial g_0}{\partial y} \Phi_{yx} = -\frac{\partial g_0}{\partial x}, \tag{8.6}$$

where

$$\Phi_{yx} = \frac{\partial y}{\partial x}, \tag{8.7a}$$

$$\frac{\partial g_0}{\partial y} = \frac{\partial g}{\partial y}(\hat{x}_{0|0}, \hat{y}_{0|0}, \theta), \tag{8.7b}$$

$$\frac{\partial g_0}{\partial x} = \frac{\partial g}{\partial x}(\hat{x}_{0|0}, \hat{y}_{0|0}, \theta). \tag{8.7c}$$

8.1.2 Time update

In the time update, we compute the one-step predictions of the predicted state mean and covariance estimates as the solution to the initial value problem defined by the expectation of the stochastic state dynamics, i.e. an ODE because (8.1) is a martingale.

Consider the initial value problem for $t \in [t_k, t_{k+1}]$

$$\frac{d\hat{x}_k}{dt}(t) = f(\hat{x}_k(t), \hat{y}_k(t), u(t), \theta), \qquad \hat{x}_k(t_k) = \hat{x}_{k|k}, \tag{8.8a}$$

$$0 = g(\hat{x}_k(t), \hat{y}_k(t), \theta). \tag{8.8b}$$

We compute the predicted state and algebraic variable mean estimates as the solution to (8.8) at time $t = t_{k+1}$, as

$$\hat{x}_{k+1|k} = \hat{x}_k(t_{k+1}), \qquad \hat{y}_{k+1|k} = \hat{y}_k(t_{k+1}). \tag{8.9}$$

We compute the predicted state covariance estimate as the solution to the integral expression defined in terms of the state sensitivities

$$P_k(t) = \Phi_{xx}(t, t_k) P_k(t_k) \Phi_{xx}^T(t, t_k) + \int_{t_k}^{t} \Phi_{xx}(t, s) \sigma_k(s) \sigma_k^T(s) \Phi_{xx}^T(t, s) ds, \tag{8.10}$$

where

$$\sigma_k(t) = \sigma(x_k(t), y_k(t), u(t), \theta). \tag{8.11}$$

We compute the predicted state covariance estimate as the solution to (8.10) at time $t = t_{k+1}$,

$$P_{k+1|k} = P_k(t_{k+1}), \tag{8.12}$$

and compute the predicted algebraic variable mean estimate, $\hat{y}_{k+1|k}$, as the solution to the algebraic equation,

$$0 = g(\hat{x}_{k+1|k}, \hat{y}_{k+1|k}, \theta). \tag{8.13}$$

We compute the predicted algebraic variable covariance estimate as

$$P_{y,k+1|k} = \Phi_{yx} P_{k+1|k} \Phi_{yx}^T, \tag{8.14}$$

and obtain the sensitivities as the solution to

$$\frac{d\Phi_{xx}}{dt}(t, s) = \frac{\partial f}{\partial x} \Phi_{xx}(t, s) + \frac{\partial f}{\partial y} \Phi_{yx}(t, s), \qquad \Phi_{xx}(s, s) = I, \tag{8.15a}$$

$$0 = \frac{\partial g}{\partial x} \Phi_{xx}(t, s) + \frac{\partial g}{\partial y} \Phi_{yx}(t, s), \tag{8.15b}$$

where the partial derivatives of the state dynamics and algebraic expressions are

$$\frac{\partial f_k}{\partial x} = \frac{\partial f}{\partial x}(\hat{x}_k(t), \hat{y}_k(t), u_k, \theta), \qquad \frac{\partial f_k}{\partial y} = \frac{\partial f}{\partial x}(\hat{x}_k(t), \hat{y}_k(t), u_k, \theta), \tag{8.16a}$$

$$\frac{\partial g_k}{\partial x} = \frac{\partial g}{\partial x}(\hat{x}_k(t), \hat{y}_k(t), \theta), \qquad \frac{\partial g_k}{dy} = \frac{\partial g}{\partial y}(\hat{x}_k(t), \hat{y}_k(t), \theta). \tag{8.16b}$$

The sensitivities are

$$\Phi_{xx}(t, s) = \frac{\partial \hat{x}_k(t)}{\partial \hat{x}_k(s)}, \qquad \Phi_{yx}(t, s) = \frac{\partial \hat{y}_k(t)}{\partial \hat{x}_k(s)}. \tag{8.17}$$

Numerical details

Within each sampling interval, $t \in [t_k, t_{k+1}]$, we approximate the solution to (8.8) and (8.10) with N_{Int} equidistant subintervals of size $h = T_s/N_{Int}$. As such, we describe the internal solutions to the first and second moments of the state as

$$\hat{x}_{k,n} = \hat{x}_k(t_k + nh), \qquad P_{k,n} = P_k(t_k + nh), \tag{8.18}$$

and describe the solution to the first and second moments of the algebraic variables as

$$\hat{y}_{k,n} = \hat{y}_k(t_k + nh), \qquad P_{y,k,n} = P_{y,k}(t_k + nh). \tag{8.19}$$

We discretise the differential algebraic equation using Euler's implicit method, as

$$0 = \hat{x}_{k,n+1} - \hat{x}_{k,n} - f_k(\hat{x}_{k,n+1}, \hat{y}_{k,n+1})h, \tag{8.20a}$$

$$0 = g(\hat{x}_{k,n+1}, \hat{y}_{k,n+1}, \theta), \tag{8.20b}$$

where $f_k(x, y) = f(x, y, u_k, \theta)$. We describe the right-hand side of the algebraic equations, (8.20), as the residual

$$R(\hat{x}_{k,n+1}, \hat{y}_{k,n+1}) = \begin{bmatrix} \hat{x}_{k,n+1} - \hat{x}_{k,n} - f_k(\hat{x}_{k,n+1}, \hat{y}_{k,n+1})h \\ g(\hat{x}_{k,n+1}, \hat{y}_{k,n+1}, \theta) \end{bmatrix}, \tag{8.21}$$

and obtain the state and algebraic variables, $\hat{x}_{k,n+1}$ and $\hat{y}_{k,n+1}$, as the solution to

$$0 = R(\hat{x}_{k,n+1}, \hat{y}_{k,n+1}). \tag{8.22}$$

We apply Newton's method to obtain the solution to (8.22). For the state covariance (8.10), we apply Euler's implicit method to approximate the solution to (8.16). We obtain the sensitivities as the solution to

$$\begin{bmatrix} I - \frac{\partial f_{k,n+1}}{\partial x} h & -\frac{\partial f_{k,n+1}}{\partial y} h \\ \frac{\partial g_{k,n+1}}{\partial x} & \frac{\partial g_{k,n+1}}{\partial y} \end{bmatrix} \begin{bmatrix} \Phi_{xx}(t_{k,n+1}, t_{k,n}) \\ \Phi_{yx}(t_{k,n+1}, t_{k,n}) \end{bmatrix} = \begin{bmatrix} I \\ 0 \end{bmatrix}, \tag{8.23}$$

where

$$\frac{\partial f_{k,n+1}}{\partial x} = \frac{\partial f}{\partial x}(\hat{x}_{k,n+1}, \hat{y}_{k,n+1}, u_k, \theta), \tag{8.24a}$$

$$\frac{\partial f_{k,n+1}}{\partial y} = \frac{\partial f}{\partial y}(\hat{x}_{k,n+1}, \hat{y}_{k,n+1}, u_k, \theta), \tag{8.24b}$$

$$\frac{\partial g_{k,n+1}}{\partial x} = \frac{\partial g}{\partial x}(\hat{x}_{k,n+1}, \hat{y}_{k,n+1}, \theta), \tag{8.24c}$$

$$\frac{\partial g_{k,n+1}}{\partial y} = \frac{\partial g}{\partial y}(\hat{x}_{k,n+1}, \hat{y}_{k,n+1}, \theta). \tag{8.24d}$$

We apply a left rectangular rule to approximate the integral in (8.10). We obtain the state covariance as

$$P_{k,n+1} = \Phi_{xx}(t_{k,n+1}, t_k)\tau_{k,n}\Phi_{xx}^T(t_{k,n+1}, t_k), \tag{8.25a}$$

$$\tau_{k,n} = P_{k,n} + \sigma_k(\hat{x}_{k,n}, \hat{y}_{k,n})\sigma_k^T(\hat{x}_{k,n}, \hat{y}_{k,n})h, \tag{8.25b}$$

where $P_{k,0} = P_{k|k}$. The covariance of the algebraic variables are obtained as

$$P_{y,k,n+1} = \Phi_{yx,k,n+1}P_{k,n+1}\Phi_{yx,k,n+1}^T, \tag{8.26}$$

where we apply the implicit function theorem to compute the sensitivities as the solution to

$$\frac{\partial g_{k,n+1}}{\partial y}\Phi_{yx,k,n+1} = -\frac{\partial g_{k,n+1}}{\partial x}. \tag{8.27}$$

The solutions over the sampling interval are

$$\hat{x}_{k+1|k} = \hat{x}_{k,N_{Int}}, \qquad P_{k+1|k} = P_{k,N_{Int}}, \tag{8.28a}$$

$$\hat{y}_{k+1|k} = \hat{y}_{k,N_{Int}}, \qquad P_{y,k+1|k} = P_{y,k,N_{Int}}. \tag{8.28b}$$

8.1.3 Measurement update

In the measurement update, we apply the filtering strategy of the linear Kalman filter on a local linearisation of the nonlinear system. We apply the implicit function theorem to compute filtered estimates of the algebraic variables and compute the innovation and its covariance, as

$$e_k = y_k^m - \hat{y}_{k|k-1}^m, \tag{8.29a}$$

$$R_{e,k} = C_k P_{k|k-1} C_k^T + R(\theta), \tag{8.29b}$$

where y_k^m is a measurement taken at time $t = t_k$. We obtain the one-step prediction of the measurement as

$$\hat{y}_{k|k-1}^m = h^m(\hat{x}_{k|k-1}, \hat{y}_{k|k-1}, \theta), \tag{8.30}$$

and apply the implicit function theorem to compute the matrix C_k, as

$$C_k = \frac{dh_{k|k-1}^m}{dx} \tag{8.31a}$$

$$= \frac{\partial h_{k|k-1}^m}{\partial x} + \frac{\partial h_{k|k-1}^m}{\partial y}\frac{\partial y_{k|k-1}}{\partial x}, \tag{8.31b}$$

where

$$\frac{\partial h_{k|k-1}^m}{\partial x} = \frac{\partial h^m}{\partial x}(\hat{x}_{k|k-1}, \hat{y}_{k|k-1}, \theta), \tag{8.32a}$$

$$\frac{\partial h_{k|k-1}^m}{\partial y} = \frac{\partial h^m}{\partial y}(\hat{x}_{k|k-1}, \hat{y}_{k|k-1}, \theta). \tag{8.32b}$$

We compute the sensitivity, $\frac{\partial y_{k|k-1}}{\partial x}$, as the solution to

$$\frac{\partial g_{k|k-1}}{\partial y} \frac{d y_{k|k-1}}{dx} = - \frac{\partial g_{k|k-1}}{\partial x}, \tag{8.33}$$

where

$$\frac{\partial g_{k|k-1}}{\partial x} = \frac{\partial g}{\partial x}(\hat{x}_{k|k-1}, \hat{y}_{k|k-1}, \theta), \tag{8.34a}$$

$$\frac{\partial g_{k|k-1}}{\partial y} = \frac{\partial g}{\partial y}(\hat{x}_{k|k-1}, \hat{y}_{k|k-1}, \theta). \tag{8.34b}$$

The Kalman gain is

$$K_{f_x,k} = P_{k|k-1} C_k^T R_{e,k}^{-1}. \tag{8.35}$$

We compute the filtered state mean and covariance estimates as

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_{f_x,k} e_k, \tag{8.36a}$$

$$P_{k|k} = \left(I - K_{f_x,k} C_k\right) P_{k|k-1} \left(I - K_{f_x,k} C_k\right)^T + K_{f_x,k} R K_{f_x,k}^T. \tag{8.36b}$$

We obtain the filtered algebraic mean estimate, $\hat{y}_{k|k}$, as the solution to

$$0 = g(\hat{x}_{k|k}, \hat{y}_{k|k}, \theta), \tag{8.37}$$

and compute the filtered algebraic variable covariance estimate, $P_{y,k|k}$, as

$$P_{y,k|k} = \frac{\partial y_{k|k}}{\partial x} P_{k|k} \left(\frac{\partial y_{k|k}}{\partial x} \right)^T. \tag{8.38}$$

We apply the implicit function theorem to obtain the sensitivities, $\frac{\partial y_{k|k}}{\partial x}$, as the solution to

$$\frac{\partial g_{k|k}}{\partial y} \frac{\partial y_{k|k}}{\partial x} = - \frac{\partial g_{k|k}}{\partial x}. \tag{8.39}$$

8.2 Summary

In this chapter, we presented the CD-EKF for nonlinear system involving SDAEs. We formulated the CD-EKF for nonlinear systems involving SDAEs by expressing the algebraic variables as a function of the state. We formulated the CD-EKF by applying the sensitivities arising from the implicit description. We presented procedures for predicted and filtered estimates mean and covariance for state and algebraic variables. For the time update, we presented numerical details relevant to the implementation of the state estimation algorithm.

Part III

Economic Model Predictive Control

CHAPTER 9 Economic Model Predictive Control

The work presented in this chapter is partially based on the papers listed in Appendices D, F, A, and C. In this chapter, we present ENMPC algorithms applied to systems described by nonlinear SDAEs. The description generalises to system described by SDEs. ENMPC describes full state feedback model-based control algorithms applied to nonlinear systems which directly optimises economic measures of performance, e.g. maximising revenue, minimising raw materials or power cost, or maximising sustainable energy use [40, 105]. Figure 9.1 presents an overview of an ENMPC system. In the application of ENMPC, we take a measurement from ta system and pass it to the state estimator. The state estimator computes an estimate of the systems states at the time of the measurement. The state estimate is passed to the optimiser. In the optimiser, an EOCP is solved numerically given the current state estimate to produce an optimal open-loop control strategy over a finite horizon. We implemented the first control action of the optimal open-loop control strategy in the system. A new measurement is taken from the system and the control-loop repeats. We consider continuous-discrete nonlinear SDAE models in the form

$$\begin{aligned}
d\boldsymbol{x}(t) &= f(\boldsymbol{x}(t), \boldsymbol{y}(t), u(t), \theta)dt + \sigma(\boldsymbol{x}(t), \boldsymbol{y}(t), \theta)d\boldsymbol{\omega}(t), \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, && (9.1)\\
0 &= g(\boldsymbol{x}(t), \boldsymbol{y}(t), \theta), && (9.2)\\
\boldsymbol{z}^m(t) &= g^m(\boldsymbol{x}(t), \boldsymbol{y}(t), \theta), && (9.3)\\
\boldsymbol{y}^m(t_k) &= h^m(\boldsymbol{x}(t_k), \boldsymbol{y}(t_k), \theta) + \boldsymbol{v}(t_k, \theta), && (9.4)
\end{aligned}$$

where $\boldsymbol{x}(t) \in \mathbb{R}^{n_x}$ are state variables, $\boldsymbol{y}(t) \in \mathbb{R}^{n_y}$ are algebraic variables, $u(t) \in \mathbb{R}^{n_u}$ are manipulated variables, $\boldsymbol{z}^m(t) \in \mathbb{R}^{n_{z,m}}$ are output variables, $\boldsymbol{y}^m(t_k) \in \mathbb{R}^{n_{y,m}}$ are measurement variables, and $\theta \in \mathbb{R}^{n_\theta}$ are model parameters. The process noise, $\boldsymbol{w}(t) \in \mathbb{R}^{n_\omega}$, is a standard Wiener process, i.e. $d\boldsymbol{\omega}(t) \sim \mathcal{N}(0, Idt)$, and $\boldsymbol{v}(t_k, \theta) \sim \mathcal{N}(0, R(\theta))$ is the normally distributed measurement noise. $f(\cdot) : (\boldsymbol{x}(t), \boldsymbol{y}(t), u(t), \theta) \longrightarrow \mathbb{R}^{n_x}$ is the state drift function, $\sigma(\cdot) : \mathbb{R}^{n_\theta} \longrightarrow \mathbb{R}^{n_x \times n_\omega}$ is the state diffusion function, $g(\cdot) : (\boldsymbol{x}(t), \boldsymbol{y}(t), \theta) \longrightarrow \mathbb{R}^{n_y}$ is the algebraic function, $g^m(\cdot) : (\boldsymbol{x}(t), \boldsymbol{y}(t), \theta) \longrightarrow \mathbb{R}^{n_{z,m}}$ is the output function, and $h^m(\cdot) : (\boldsymbol{x}(t_k), \boldsymbol{y}(t_k), \theta) \longrightarrow \mathbb{R}^{n_{y,m}}$ is the measurement function.

The chapter is structured as follows: in section 9.1, we describe optimal control.

We present an OCP for nonlinear systems involving SDAEs in Bolza form. Furthermore, we present classical measures of performance, e.g. target tracking and input regularisation, as well as economic performance measures, e.g. product revenue and input cost. Furthermore, we present hard constraints on independent variables, soft constraints for dependent variables, and formulate an extended OCP. In section 9.2, we present a conceptual overview of ENMPC and define the control system in terms of the state estimator and EOCP. Section 9.3 summarises the work presented in the chapter.

9.1 Optimal control

In this section, we present a brief overview of optimal control. We formulate OCPs for nonlinear systems involving DAEs. We present examples of objective and constraint functions relevant to the production of SCP.

The objective of optimal control, or dynamic optimisation, is to determine open-loop control strategies for optimal operation of a dynamical system with respect to one or more performance measures. The objective function of the OCP describes the performance measure, e.g. a tracking error, a penalty on the rate-of-movement of the manipulated variables, or a measure of profit or cost. The constraints represent the dynamics of the operated system as well as operational and physical limitations, e.g. minimum and maximum flow rates a pump can handle, the maximum volume of a reactor vessel, or non-negativity of physical variables such as mass or concentration. Finally, we describe a direct simultaneous approach for formulating OCPs as NLPs for numerical solution.

9.1.1 Optimal control problem

Consider an OCP in Bolza form

$$\min_{[x(t);y(t);u(t)]_{t_0}^{t_f}} \quad \phi = \int_{t_0}^{t_f} l(t, x(t), y(t), u(t), \theta)dt + \hat{l}(x(t_f), y(t_f), \theta), \tag{9.5}$$

subject to

$$\begin{aligned}
x(t_0) &= x_0, && && (9.6a)\\
\frac{dx}{dt} &= f(x(t), y(t), u(t), \theta), && t_0 \leq t \leq t_f, && (9.6b)\\
0 &= g(x(t), y(t), \theta), && t_0 \leq t \leq t_f, && (9.6c)\\
z^m(t) &= g^m(x(t), \theta), && t_0 \leq t \leq t_f, && (9.6d)\\
c_{lb}(t) &\leq c(t, x(t), y(t), u(t), \theta) \leq c_{ub}(t), && t_0 \leq t \leq t_f, && (9.6e)
\end{aligned}$$

where $x(t)$ are state variables, $y(t)$ are algebraic variables, $u(t)$ are manipulated variables, and θ are parameters. (9.5) describes the objective function of the OCP. The

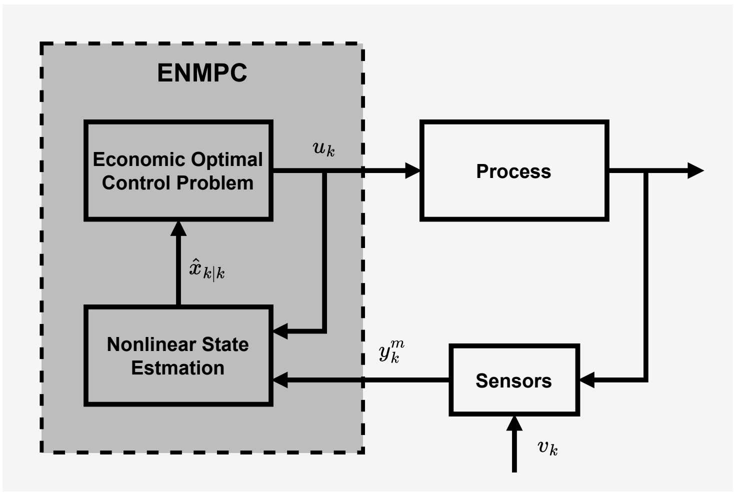


Figure 9.1: Illustration of the economic nonlinear model predictive control. Sensors pass measurements from the process to the estimation block, where a nonlinear state estimation algorithm computes an estimate of the current state and covariance. The state information is passed to the optimisation block, where an economic optimal control problem is solved to yield an optimal open-loop control strategy. The current control action is implemented in the process and the control-loop repeats.

functions $l(\cdot)$ and $\hat{l}(\cdot)$ are the Lagrange and Mayer terms, respectively. (9.6a)-(9.6d) describe the nonlinear system dynamics. (9.6e) describes general constraints operational and variable constraints, e.g. non-negativity for for states and input or volume and flow equipment constraints.

9.1.2 Zero-order hold parametrisation

For the applications discussed in this work, we consider a zero-order hold parametrisation of the manipulated variables. For the control horizon $t \in [t_0, t_f]$, we define the final time as

$$t_f = t_0 + \sum_{k=1}^{N} t_k, \qquad t_k = kT_s, \tag{9.7}$$

where T_s is the time-interval defining a series of equidistant control intervals, or sampling intervals, for $t_k \leq t < t_{k+1}$. We define for this equidistant grid, a zero-order hold parametrisation of the manipulated variables

$$u(t) = u_k, \qquad \text{for} \qquad t_k \leq t < t_{k+1}, \tag{9.8}$$

where u_k are constant values of the manipulated inputs within each control intervals. As such, we note that we can define the controlled inputs in terms of the set of constant input levels

$$\{u_k\}_{k=0}^{N-1}. \tag{9.9}$$

9.1.3 Measures of performance (objectives)

In optimal control, dynamical systems may be optimised with respect to several performance measures. In this section, we describe objective function terms in Bolza form to describe performance of dynamical systems. We consider objective functions in the form

$$\phi = \sum_{i \in \mathcal{O}} \alpha_i \phi_i, \qquad 1 = \sum_{i \in \mathcal{O}} \alpha_i, \tag{9.10}$$

where $\mathcal{O}$ is the set of objectives, α_i are objective weights, and ϕ_i are objectives. We consider objectives for; setpoint tracking, input rate-of-movement penalisation, input economy, product economy, as well as hard and soft constraints in this section.

Setpoint tracking

For tracking an output to a target trajectory, i.e. setpoint, we define the Lagrange term

$$\phi_z = \int_{t_0}^{t_f} l_z(t, x(t), u(t), \theta)dt + \hat{l}_z(x(t_f), y(t_f), \theta). \tag{9.11}$$

The Lagrange function is

$$l_z(t, x(t), u(t), \theta) = \|z(t) - \bar{z}(t)\|_{Q_z}^2, \tag{9.12}$$

where $\bar{z}(t)$ is the target trajectory and $\|\cdot\|_{Q_z}^2$ is a weighted 2-norm, defining the Euclidean distance between the output and the target. The Mayer term is

$$\hat{l}_z(x(t_f), y(t_f), \theta) = 0. \tag{9.13}$$

Input rate-of-movement penalisation

For penalising the rate-of-movement of the manipulated variables, we define the Lagrange term

$$\phi_{\Delta u} = \int_{t_0}^{t_f} l_{\Delta u}(t, x(t), u(t), \theta)dt + \hat{l}_{\Delta u}(x(t_f), y(t_f), \theta) \tag{9.14}$$

$$= \sum_{k=0}^{N-1} \int_{t_k}^{t_{k+1}} l_{\Delta u,k}(t, x(t), u(t), \theta)dt + \hat{l}_{\Delta u}(x(t_f), y(t_f), \theta). \tag{9.15}$$

The Lagrange function is

$$l_{\Delta u,k}(t, x(t), u(t), \theta) = \|\Delta u_k\|_{Q_{\Delta u}}^2, \qquad \Delta u_k = u_k - u_{k-1}. \tag{9.16}$$

The input u_{-1} is not part of the decision variables, but is defined as the current control action implemented before time $t = t_0$. The Mayer term is

$$\hat{l}_{\Delta u}(x(t_f), y(t_f), \theta) = 0. \tag{9.17}$$

Input economy

We define the input economy objective

$$\begin{aligned} \phi_{u,\text{eco}} = & \int_{t_0}^{t_f} l_{u,\text{eco}}(t, x(t), y(t), u(t), \theta) dt \\ & + \hat{l}_{u,\text{eco}}(x(t_f), y(t_f), \theta) \qquad (9.18a) \\ = & \sum_{k=0}^{N-1} \int_{t_k}^{t_{k+1}} l_{u,\text{eco},k}(t, x(t), y(t), u_k, \theta) dt \\ & + \hat{l}_{u,\text{eco}}(x(t_f), y(t_f), \theta). \qquad (9.18b) \end{aligned}$$

The Lagrange function is

$$l_{u,\text{eco},k}(t, x(t), y(t), u(t), \theta) = p_{u,\text{eco}} u_k, \tag{9.19}$$

where $p_{u,\text{eco}}$ is the unit cost per unit time of the inputs. The Mayer term is

$$\hat{l}_{u,\text{eco}}(x(t_f), y(t_f), \theta) = 0. \tag{9.20}$$

This is an example of an economic objective.

Product economy

Consider the U-loop reactor defined in Chapter 2. For a product harvested from the reactor at a rate of $F(t)$, where $F(t) \in u(t)$, we define the product economy objective

$$\phi_{P,\text{eco}} = \int_{t_0}^{t_f} l_{P,\text{eco}}(t, x(t), y(t), u(t), \theta) dt + \hat{l}_{P,\text{eco}}(x(t_f), y(t_f), \theta). \tag{9.21}$$

The Lagrange function is

$$l_{P,\text{eco}}(t, x(t), y(t), u(t), \theta) = -p_P F(t) \bar{c}_P(t), \tag{9.22}$$

where p_P is the unit price per unit weight of the product, P, $F(t)$ is the harvest rate from the top tank, and $c_P(t)$ is the concentration of the product, P. The Mayer term is

$$\hat{l}_{P,\text{eco}}(x(t_f), y(t_f), \theta) = p_P \left(c_P(t_0)V + \int_0^L \bar{c}_P(t_0, z)Adz \right) \\ - p_P \left(c_P(t_f)V + \int_0^L \bar{c}_P(t_f, z)Adz \right), \tag{9.23}$$

where V is the volume of the top tank, A is the cross-sectional area of the U-loop leg, L is the length of the U-loop leg, and $c_P(t_0)$ and $\bar{c}(t_0, z)$ are the initial product concentrations of the top tank and U-loop leg, respectively. The Lagrange term defines the value of the product harvested from the reactor, i.e. the negative cost for minimisation. The Mayer term defines the cost-to-go, by subtracting the value of the initial product and including the value of the product remaining in the reactor at the final time. The cost-to-go ensures that the reactor is not emptied for additional profit at the end of the control horizon. This is an example of an economic objective.

9.1.4 Constraints

Hard constraints

It is often the case, that independent variables, e.g. manipulated inputs in the case of control systems, are constrained in the OCP. This limits the search space in the resulting NLP, as well as corresponding to physical or operational limitations in the system, e.g. minimum and maximum flow-rates of a pump, minimum or maximum actuator change, or nonnegativity of physical quantities. We may represent such box constraints on inputs, as

$$c_{u,lb}(t) \leq u(t) \leq c_{u,ub}(t), \tag{9.24}$$

or rate-of-movement of the inputs, as

$$c_{lb,\Delta u}(t) \leq \frac{du}{dt}(t) \leq c_{ub,\Delta u}(t). \tag{9.25}$$

Similarly, systems often present physical or operational constraints on dependent variables, e.g. states, algebraic variables, or outputs. We may represent such constraints on the state and algebraic variables, as

$$c_{lb,xy}(t) \leq c_{xy}(x(t), y(t), \theta) \leq c_{ub,xy}(t). \tag{9.26}$$

However, in real-time applications the dependent variables cannot always be guaranteed to remain within such hard constraints, as diffusion may drive those variables into infeasible regions. Therefore, it can be beneficial to introduce soft constraints instead of the hard constraints defined in (9.26).

Soft constraints

We introduce soft constraints to approximate the same restrictions on the values of dependent variables as described in (9.26), while removing issues related to infeasibility. Consider the modified constraints on the state variables

$$c_{lb,xy}(t) - p(t) \leq c_{xy}(x(t), y(t), \theta) \leq c_{ub,xy}(t) + q(t), \tag{9.27a}$$
$$0 \leq p(t), \tag{9.27b}$$
$$0 \leq q(t). \tag{9.27c}$$

The variables p and q are slack variables introduced to increase in value if the state and algebraic variables violate the hard constraints defined in (9.26). The introduction of p and q thus increases the number of variables in the OCP, but removes issues regarding feasibility, as the values of p and q can simply increase freely to account for violations of the constraints. However, as mentioned previously, the slack variables can increase freely and the softening, (9.27), therefore does not approximate the hard constraints defined in (9.26). We introduce a penalty of the slack variables, as

$$\phi_{pq} = \int_{t_0}^{t_f} l_{qp}(t, x(t), y(t), u(t), \theta)dt + \hat{l}_{pq}(x(t_f), y(t_f), \theta). \tag{9.28}$$

The Lagrange function is a linear and quadratic penalty on the slack variables

$$l_{pq}(t, x(t), y(t), u(t), \theta) = l_p(t, x(t), y(t), u(t), \theta) + l_q(t, x(t), y(t), u(t), \theta), \tag{9.29}$$

where

$$l_p(t, x(t), y(t), u(t), \theta) = \|p(t)\|^2_{Q_{p,2}} + |p(t)|_{Q_{p,1}}, \tag{9.30a}$$
$$l_q(t, x(t), y(t), u(t), \theta) = \|q(t)\|^2_{Q_{q,2}} + |q(t)|_{Q_{q,1}}. \tag{9.30b}$$

The Mayer term is

$$\hat{l}_{pq}(x(t_f), y(t_f), \theta) = 0. \tag{9.31}$$

The penalty on the slack variables, (9.28), defines an optimum for $q = p = 0$, i.e. where the hard constraints, (9.26), are satisfied. If we include other measures of performance in the objective with the slack penalties, a sufficiently large value of scaling weights $Q_{p,2}$, $Q_{p,1}$, $Q_{q,2}$, and $Q_{q,1}$ will make the OCP prefer feasibility in the hard constrained problem over fulfilment of the remaining performance measures. If the optimum lies on constraints, (9.26), the optimum in the soft constrained case will violate the hard constraints to the smallest degree possible. The choice of scaling determines the degree to which the constraints are violated. However, the problem remains feasible. Note that one may also introduce several soft constraints on the same variable to increase the strength of the penalty gradually. This can be useful if small violations are not detrimental to performance, but larger violations are, e.g.

in diabetes related model-based control solutions [106, 107]. The OCP with soft constraints, (9.27) and (9.28), is

$$\min_{[x(t);y(t);u(t);p(t);q(t)]_{t_0}^{t_f}} \quad \phi, \tag{9.32}$$

subject to

$$
\begin{aligned}
x(t_0) &= x_0, && && (9.33a)\\
\frac{dx}{dt} &= f(x(t),y(t),u(t),\theta), && t_0 \le t \le t_f, && (9.33b)\\
0 &= g(x(t),y(t),\theta), && t_0 \le t \le t_f, && (9.33c)\\
z^m(t) &= g^m(x(t),\theta), && t_0 \le t \le t_f, && (9.33d)\\
c_{lb}(t) &\le c(t,x(t),y(t),u(t),\theta) \le c_{ub}(t), && t_0 \le t \le t_f, && (9.33e)\\
c_{lb,s}(t) - p(t) &\le c_s(t,x(t),y(t),u(t),\theta) \le c_{ub,s}(t) + q(t), && t_0 \le t \le t_f, && (9.33f)\\
0 &\le p(t), && t_0 \le t \le t_f, && (9.33g)\\
0 &\le q(t), && t_0 \le t \le t_f, && (9.33h)
\end{aligned}
$$

where the function $c_{lb,s}$ and $c_{ub,s}$, (9.33f), are the softened constraints in the form (9.27) and the objective function ϕ includes the slack penalty objectives, (9.28).

9.1.5 Temporal discretisation

We apply a temporal discretisation of the continuous-time OCP presented in (9.5)-(9.6) to formulate a numerical optimisation problem, i.e. an NLP. We apply a simultaneous approach to the formulation of the NLP.

Direct simultaneous approach

In the simultaneous approach, we include the numerical integration scheme directly as constraints in the NLP, and integrate the objective function numerically with the same points. For consistency, we apply the same numerical integration scheme in the discretisation of the dynamical equations and the objective integrals. We apply the implicit Euler scheme for numerical integration of the dynamical constraints and the right-rectangular rule for numerical integration of the objective Lagrange terms. We define the sets of discrete state, algebraic, and manipulated variables

$$\{\{x_{k,n}\}_{n\in\mathcal{N}_k}\}_{k\in\mathcal{N}_c}, \qquad \{\{y_{k,n}\}_{n\in\mathcal{N}_k}\}_{k\in\mathcal{N}_c}, \qquad \{u_k\}_{k\in\mathcal{N}_c}, \tag{9.34}$$

where $\mathcal{N}_k = \{0,1,\ldots,N_k\}$ is the set of internal integration step within each sampling interval and $\mathcal{N}_c = \{0,1,\ldots,N-1\}$ is the finite set of sampling times in the controller, i.e. N is the control horizon. The evolution of the states and algebraic variables are defined for the set $\mathcal{N}_{k-1} = \{0,1,\ldots,N_z-1\}$, as

$$x_{k,n+1} = x_{k,n} + f(x_{k,n+1}, y_{k,n+1}, u_k, \theta)\Delta t_{k,n}, \qquad n \in \mathcal{N}_{k-1}, \qquad k \in \mathcal{N}_c, \tag{9.35}$$

and the continuity constraints are

$$x_{k+1,0} = x_{k,N_k}, \qquad k \in \mathcal{N}_c. \tag{9.36}$$

The initial value for the system is the state estimate at time $t = t_0$

$$x_{0,0} = \hat{x}_{0|0}. \tag{9.37}$$

For simplicity in notation, we define the dynamical constraints in 9.35 in residual form as well as the algebraic equations, as

$$0 = D(x_{k,n+1}, x_{k,n}, y_{k,n}, y_{k,n+1}, u_k, \theta), \qquad n \in \mathcal{N}_{k-1}, \qquad k \in \mathcal{N}_c, \tag{9.38}$$

where

$$D(x_{k,n+1}, x_{k,n}, y_{k,n}, y_{k,n+1}, u_k, \theta) = \begin{bmatrix} R(x_{k,n+1}, x_{k,n}, y_{k,n}, y_{k,n+1}, u_k, \theta) \\ g(x_{k,n+1}, x_{k,n}, y_{k,n+1}, \theta) \end{bmatrix}. \tag{9.39}$$

We note that this notation introduces redundant variables in the continuity constraints, which can be eliminates in the implementation. We similarly approximate the Lagrange term of the objective function with the right-rectangular rule (equivalent to implicit Euler), as

$$\begin{aligned}
\phi &\approx \Phi && (9.40a)\\
&= \Phi(\{\{x_{k,n}, y_{k,n}\}_{n\in\mathcal{N}_k}, u_k\}_{k\in\mathcal{N}_c}, \theta) && (9.40b)\\
&= \Phi_L + \Phi_M && (9.40c)\\
&= \Phi_L(\{\{x_{k,n}, y_{k,n}\}_{n\in\mathcal{N}_k}, u_k\}_{k\in\mathcal{N}_c}, \theta) + \Phi_M(x_{N-1,N_k}, y_{N-1,N_{k-1}}, \theta), && (9.40d)
\end{aligned}$$

where the Lagrange term is

$$\Phi_L = \sum_{k\in\mathcal{N}_c} \sum_{n\in\mathcal{N}_{k-1}} l(t_{k,n+1}, x_{k,n+1}, y_{k,n+1}, u_k, \theta)\Delta t_{k,n}, \tag{9.41}$$

and the Mayer term is

$$\Phi_M = \hat{l}(x_{N-1,N_k}, y_{N-1,N_k}, \theta). \tag{9.42}$$

We define the NLP for numerical solution of the OCP defined in (9.5)-(9.6), as

$$\min_{\{\{x_{k,n}, y_{k,n}\}_{n\in\mathcal{N}_k}, u_k\}_{k\in\mathcal{N}_c}} \quad \Phi = \Phi_L + \Phi_M, \tag{9.43}$$

subject to

$$\begin{aligned}
x_{0,0} &= \hat{x}_{0|0}, && && (9.44a)\\
x_{k+1,0} &= x_{k,N_z}, && && k \in \mathcal{N}_c, \quad (9.44b)\\
0 &= D(x_{k,n+1}, x_{k,n}, y_{k,n}, y_{k,n+1}, u_k, \theta), && n \in \mathcal{N}_{k-1}, && k \in \mathcal{N}_c, \quad (9.44c)\\
z_{k,n} &= g^m(x_{k,n}, y_{k,n}, \theta), && n \in \mathcal{N}_k, && k \in \mathcal{N}_c, \quad (9.44d)\\
c_{lb,k,n} &\leq c(t_{k,n}, x_{k,n}, y_{k,n}, u_k, \theta) \leq c_{ub,k,n}, && n \in \mathcal{N}_k, && k \in \mathcal{N}_c. \quad (9.44e)
\end{aligned}$$

We apply an algorithm of choice for solving the NLP defined by (9.43)-(9.44), e.g. interior-point algorithm [50] or sequential quadratic programming algorithm [49].

9.2 Economic model predictive control

In this section, we present a conceptual overview of ENMPC. As mentioned previously, ENMPC is a full state feedback control strategy in which a we take a measurement from a system and pass it to a state estimator. The state estimator provides predicted and filtered estimate of the system states. Given the state estimate, we solve an EOCP for an optimal open-loop control strategy. We implement the first control action of the solution to the EOCP in the system. At the following sampling time, we take a new measurement repeat the control-loop.

System description

Consider a system governed by a SDAE model in the form (9.1). We describe the evolution of the system in discrete sampling intervals, as

$$\boldsymbol{z}_{i+1}^{s} = F(\boldsymbol{z}_i^s, u_i, \boldsymbol{\omega}_i^s, \theta^s), \tag{9.45a}$$
$$\boldsymbol{y}_i^{m,s} = h^m(\boldsymbol{z}_i^s, \theta^s) + \boldsymbol{v}_i^s, \tag{9.45b}$$

where the variable $\boldsymbol{z}_i^s = (\boldsymbol{x}_i^s, \boldsymbol{y}_i^s)$ are the state and algebraic variables, $F^s(\cdot)$ is a function evolving the system one sampling time forward in time according to to state and algebraic dynamics described in (9.1), $g^m(\cdot)$ are measurement dynamics, and $\boldsymbol{v}_{ki}^s \sim \mathcal{N}(0, R^s(\theta^s))$ is normally distributed measurement noise.

Controller description

We describe the control system in terms of the nonlinear state estimator and EOCP solver in the same discrete sampling intervals, as

$$z_{i+1}^{c} = \kappa(z_i^c, u_i, y_{i+1}^{m,s}, \theta^c), \tag{9.46a}$$
$$u_i = \lambda(z_i^c, \theta^c), \tag{9.46b}$$

where $z_i^c = (x_i^c, y_i^x)$ is a filtered estimate of state and algebraic variables at time $t = t_i$ computed by the state estimation function, $\kappa(\cdot)$, and u_i is the control action to be implemented at time $t = t_i$ computed by the EOCP function $\lambda(\cdot)$.

ENMPC algorithm

We describe ENMPC in terms of the four base operations at time $t = t_k$:
1) we take a measurement is taken from the system, as

$$y_k^{m,s} = h^m(z_k^s, \theta^s) + v_k^s(\theta^s), \tag{9.47}$$

2) we compute filtered estimates of state and algebraic variables from the measurement, as

$$z_k^c = \kappa(z_{k-1}^c, u_{k-1}, y_k^{m,s}, \theta^c), \tag{9.48}$$

3) we compute an optimal control action based on the information from the state estimate, as

$$u_k = \lambda(z_k^x, \theta^c), \tag{9.49}$$

4) we implement the optimal control action in the system and let it progress forward on sampling time, as

$$z_{k+1}^s = F(z_k^s, u_k, \omega_k^s, \theta^s), \tag{9.50}$$

5) repeat from 1).

9.3 Summary

In this chapter, we presented ENMPC and described EOCP. We introduced OCPs for nonlinear systems involving DAEs. We formulated the OCP. We describe classical and economic measures of performance, e.g. target tracking and production revenue. We described hard constraints on independent variables, i.e. manipulated inputs, and soft constraints on the dependent variables, e.g. states and algebraic variables. We formulated an extended OCP for the including of soft constraints. Finally, we presented a conceptual description of ENMPC and described the implementation algorithmically.

Part IV

Numerical Examples

CHAPTER 10 Pilot-Scale U-loop Bioreactor for Single-Cell Protein Production

This chapter is a summary of the work presented in the publications listed in Appendices D, E, and F. In this chapter, we present economic optimising control solutions for SCP production in a U-loop bioreactor. We formulate and solve an EOCP for revenue maximisation of the SCP production and minimisation of raw-material cost. We present and implement a CD-EKF for nonlinear systems involving SDEs. We perform a numerical experiment of state estimation using the CD-EKF on the U-loop bioreactor and demonstrate that we can estimate uncertain parameters online. We present and implement ENMPC for SCP production in the U-loop bioreactor. We apply the CD-EKF and EOCP mentioned previously in the ENMPC. Additionally, we apply a proportional (P) controller to stabilise the process between control actions determined by the ENMPC. Finally, we perform a numerical closed-loop experiment with ENMPC for SCP production in the U-loop bioreactor. Figure 10.1 illustrates the ENMPC applied to the U-loop bioreactor.

The chapter is structured as follows: in section 10.1, we present a summary of the publication listed in Appendix D. The publication presents an EOCP for profit maximisation of SCP production in a U-loop bioreactor. In section 10.2, we preset a summary of the publication listed in Appendix E. The publication presents the CD-EKF, and we perform a numerical experiment applying the CD-EKF for state estimation for SCP production in a U-loop bioreactor. In section 10.3, we present a summary of the publication listed in Appendix F. In the publication, we present an ENMPC, and we perform a closed-loop numerical experiment with ENMPC for SCP production in a U-loop bioreactor. Finally, section 10.4 summarises the work presented in the chapter.

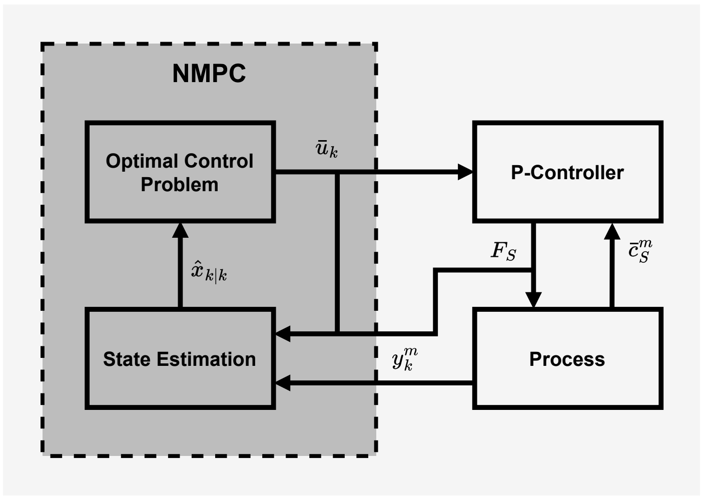


Figure 10.1: Illustration of the economic nonlinear model predictive control systems applied to the U-loop bioreactor in the publication listed in Appendix F.

10.1 U-loop economic optimal control

This section is a summary of the publication listed in Appendix D. In this paper, we formulate an EOCP for SCP production in a U-loop bioreactor. We apply the methanol based growth model described in Chapter 3. We describe the growth model by the stoichiometry

$$1.366S + 0.600O \longrightarrow X, \qquad r_1(c). \tag{10.1}$$

We apply the U-loop bioreactor model described in Chapter 2 with 20 discrete volumes in the spatial discretisation of the PFR. We formulate an ODE model in the form

$$dx(t) = f(x(t), u(t), d(t), \theta)dt, \qquad x(t_0) = x_0. \tag{10.2}$$

The model has 83 states and 3 manipulated inputs. We formulate an EOCP with the objective of minimising input cost and maximising product revenue, i.e. optimal profit, as

$$\min_{[x(t)]_{t_0}^{t_f}, \{u_k\}_{k=0}^{N-1}} \quad \phi = \phi_{X,\text{eco}} + \phi_{u,\text{eco}}, \tag{10.3}$$

subject to

$$x(t_0) = \hat{x}_{0|0}, \tag{10.4a}$$

$$dx(t) = f(x(t), u(t), \theta)dt,, \qquad t \in [t_0, t_f], \tag{10.4b}$$

$$u(t) = u_k, \qquad t \in [t_k, t_{k+1}], \qquad k \in \{0, 1, \ldots, N-1\}. \tag{10.4c}$$

The performance measure $\phi_{X,\text{eco}}$ is the SCP production revenue maximisation metric, (9.21)-(9.23), and the performance measure $\phi_{u,\text{eco}}$ is the input cost minimisation metric described in (9.18)-(9.20). The objective ϕ describes the maximisation of profit associated with production, i.e. revenue minus expenses. We note that there are other economic metric which could be considered in this context for a more complete picture of profit, e.g. energy consumption or cooling. We apply two different direct simultaneous approaches to discretise the EOCP; with and without explicit continuity constraints. We apply an implicit Euler numerical integration scheme, i.e. right rectangular rule, in the temporal discretisation of the EOCP. The NLP arising from the temporal discretisation of the EOCP has 102600 decision variables and 100800 residual equations when continuity constraints are included and 52200 decision variables and 50400 residual equations in the reduced case. We solve the NLPs in less than 30 seconds for sampling intervals of 3 minutes, indicating that the system is real-time feasible. Figure 10.2 illustrates the optimal control profiles computed as the solution to the NLPs.

10.2 U-loop state estimation

This section is a summary of the publication listed in Appendix E. In this paper, we formulate a continuous-discrete SDE model for the U-loop reactor described in Chapter 2. We apply the methanol based growth kinetics described in (3.7). We extend the ODE model resulting from the growth kinetics and U-loop reactor dynamics with the introduction of stochastic parameters, as

$$d\boldsymbol{\theta}_i(t) = \kappa_{\theta_i}\left(\bar{\theta}_i - \boldsymbol{\theta}_i(t)\right) + \sigma_{\theta_i} d\boldsymbol{\omega}_{\theta_i}(t), \tag{10.5}$$

where $\boldsymbol{\theta}_i(t)$ is the new stochastic state representing an uncertain parameter, $\bar{\theta}_i$ is the nominal value for the parameter, κ_{θ_i} describes the magnitude of the drift term, i.e. how much the parameter is driven toward the nominal value, and σ_{θ_i} describes the

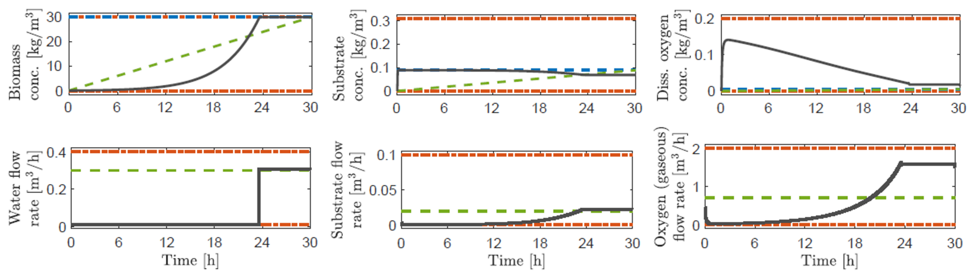


Figure 10.2: Solution to economic optimal control problem for single-cell protein production in a U-loop bioreactor.

magnitude of the diffusion term, i.e. how much the parameter diffuses away from the nominal value. We introduce four uncertain model parameters, as

$$d\boldsymbol{\mu}_{\max}(t) = \kappa_{\mu_{\max}}\left(\bar{\mu}_{\max} - \boldsymbol{\mu}_{\max}(t)\right) + \sigma_{\mu_{\max}} d\boldsymbol{\omega}_{\mu_{\max}}(t), \tag{10.6a}$$

$$d\boldsymbol{\gamma}_S(t) = \kappa_{\gamma_S}\left(\bar{\gamma}_S - \boldsymbol{\gamma}_S(t)\right) + \sigma_{\gamma_S} d\boldsymbol{\omega}_S(t), \tag{10.6b}$$

$$d\boldsymbol{\gamma}_O(t) = \kappa_{\gamma_O}\left(\bar{\gamma}_O - \boldsymbol{\gamma}_O(t)\right) + \sigma_{\gamma_O} d\boldsymbol{\omega}_O(t), \tag{10.6c}$$

$$d\boldsymbol{C}_{F,S}(t) = \kappa_{C_{F,S}}\left(\bar{C}_{F,S} - \boldsymbol{C}_{F,S}(t)\right) + \sigma_{C_{F,S}} d\boldsymbol{\omega}_{C_{F,S}}(t). \tag{10.6d}$$

Coupling the SDE model resulting from the introduction of uncertain parameters described in (10.6) with a state estimator provides online parameter estimates for the modelled parameters. We introduce a measurement model which measures dissolved oxygen concentration in the U-loop leg at three separate points. Furthermore, we introduce a stabilising P-controller to track the substrate concentration to the optimal value as described in section 3.1. The P-controller assumes we can measure the substrate concentration online, and is formulates as

$$\tilde{F}_S = F_S^* + K_{c_S}\left(\bar{C}_S^* - \bar{C}_S\right), \tag{10.7}$$

where $\tilde{F}_S$ is the implemented substrate flow-rate, F_S^* is the optimal open-loop control strategy for the substrate flow-rate, $\bar{C}_S$ is the measured substrate concentration in the top tank, and $\bar{C}_S^*$ is the optimal substrate concentrations as described in 3.1. We perform a numerical example with the open-loop control strategy determined by the EOCP presented in section 10.1 and with the stabilising P-controller. Figure 10.3 presents the state estimates for the numerical experiment, and Figure 10.4 presents the online estimates of the modelled parameters from the numerical experiment.

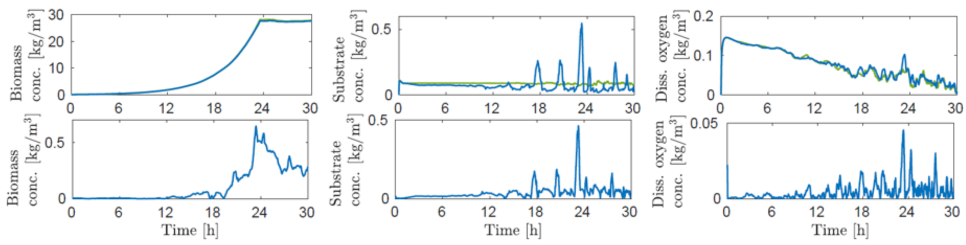


Figure 10.3: Results of numerical experiment of state estimation for the U-loop bioreactor with CD-EKF. This figures describes the estimates of the state variables.

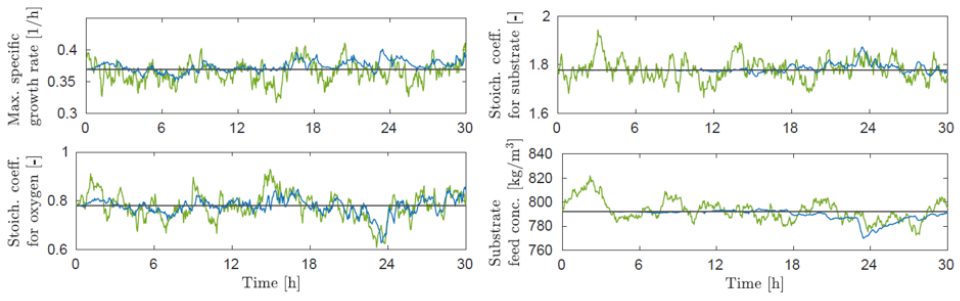


Figure 10.4: Results of numerical experiment of state estimation for the U-loop bioreactor with CD-EKF. This figures describes the estimates of the parameter variables.

10.3 U-loop nonlinear model predictive control

This section is a summary of the publication listed in Appendix F. In this paper, we present an ENMPC for SCP production in a U-loop bioreactor. We formulate a continuous-discrete SDE model based on the methanol growth model described in section 3.1 and the U-loop reactor model described in Chapter 2. We measure the dissolved oxygen at three points in the U-loop leg. We extend the model with stochastic models describing the uncertain parameters described in section 10.2. We also apply the stabilising P-controller described in section 10.2 and introduce clipping at minimum and maximum substrate flow rates, as

$$\tilde{F}_S = F_S^* + K_{c_S}\left(\bar{C}_S^* - \bar{C}_S\right), \tag{10.8a}$$

$$F_S = \max\{F_{S,\min}, \min\{\tilde{F}_S, F_{S,\max}\}\}. \tag{10.8b}$$

In the ENMPC, we apply the CD-EKF described in Appendix E and solve the EOCP using the direct simultaneous approach presented in Appendix D. Furthermore, we introduce an input rate-of-movement penalty objective in the OCP, as described in 9.1. The EOCP is defined as

$$\min_{[x(t),u(t)]_{t_k}^{t_{k+N}}} \quad \phi = \phi_{X,\text{eco}} + \phi_{u,\text{eco}} + \phi_{\Delta u} + \phi_s, \tag{10.9}$$

subject to

$$x(t_0) = \hat{x}_{0|0}, \tag{10.10a}$$

$$dx(t) = f(x(t), u(t), \theta)dt,, \quad t \in [t_k, t_{k+N}], \tag{10.10b}$$

$$u(t) = u_{k+j|k}, \quad t \in [t_{k+j}, t_{k+j+1}], \quad j \in \{0, 1, \dots, N-1\}, \tag{10.10c}$$

$$u_{\min} \le u_{k+j|k} \le u_{\max}, \quad j \in \{0, 1, \dots, N-1\}. \tag{10.10d}$$

The performance measures $\phi_{X,\text{eco}}$ and $\phi_{u,\text{eco}}$ are as defined in (10.3). The performance measure $\phi_{\Delta u}$ is the input rate-of-movement penalty described in (9.14)-(9.17) and ϕ_s is the penalty on the slack variables introduced by the including of soft constraints as defined in (9.27)-(9.30). The numerical experiment was conducted over a 30 hours with the NLP being solved once every hour over a 20 hour control horizon. The P-controller stabilises the process between sampling times every minute. The maximum computation time of the NLP was 70 seconds, which shows that the ENMPC is feasible for real-time application. Figure 10.5 describes the states and manipulated variables resulting from the numerical experiment. Figure 10.6 describes the estimates of the modelled parameters computed by the CD-EKF.

10.4 Summary

In this chapter, we presented advanced process control tools for economic optimising control for SCP production in a U-loop bioreactor. The chapter was a summary of the publications listed in Appendices D, E, and F. We formulated and solved an EOCP for profit maximisation of SCP production in the U-loop reactor. We described the revenue in terms of the harvested biomass and cost in terms of the input raw material expenses. Additionally, we included Mayer terms describing the value of the initial and final biomass content of the U-loop reactor. This removed the tendency for the ENMPC to empty the reactor at the final time-step in the control horizon. We presented the CD-EKF and a model extension including uncertain parameters in the

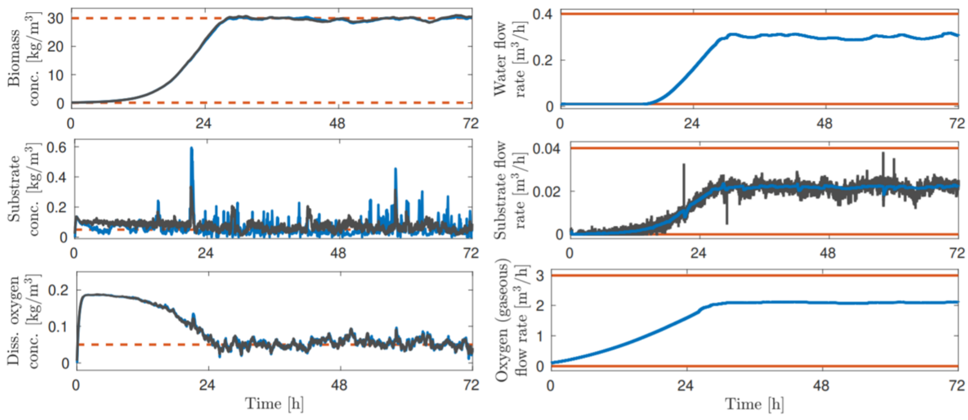


Figure 10.5: Numerical closed-loop nonlinear model predictive control experiment for the U-loop bioreactor. This figure describes the states and manipulated inputs.

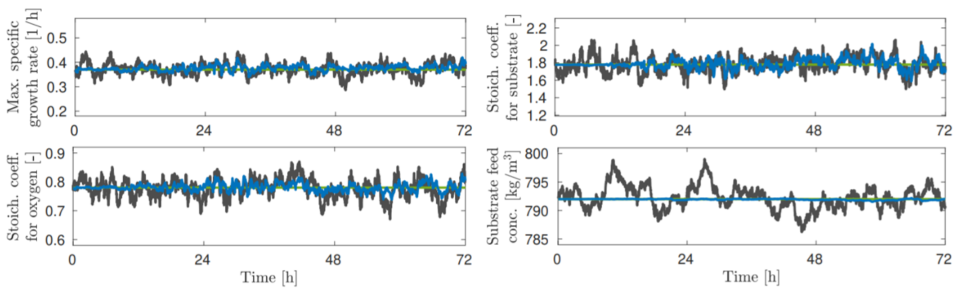


Figure 10.6: Numerical closed-loop nonlinear model predictive control experiment for the U-loop bioreactor. This figure describes the online estimates of the modelled parameters.

state space. We applied the CD-EKF to estimate states and modelled parameters in an numerical experiment for SCP production in the U-loop reactor with a stabilising P-controller. Finally, we presented ENMPC and applied it in a numerical experiment with the U-loop reactor. We demonstrated that the ENMPC system could successfully stabilise the process through the start up and at high productivity while maximising profit. Furthermore, the implementation was demonstrated to be real-time feasible.

CHAPTER 11 Laboratory-Scale Continuous Stirred Tank Reactor for Single-Cell Protein Production

This chapter is partially based on the work presented in the publication listed in Appendix A. In this chapter, we present economic optimising control for SCP production in a laboratory-scale fermenter. We present a growth models based on methane as carbon source and ammonium as nitrogen source. We include chemical equilibrium reactions to determine the pH-value and concentrations of components relevant to growth, e.g. ammonium. This results in a nonlinear system involving SDAEs. We present an EOCP to maximise profit in the nonlinear system. We apply a direct simulataneous approach and an implicit Euler temporal discretisation to formulate a nonlinear NLP. We compute the optimal trajectory as the solution to the NLP. We present the CD-EKF and extend the SCP production model with an uncertain model parameter. Finally, we apply the optimal trajectory from the EOCP in an open-loop numerical experiment for state estimation with the CD-EKF. The state and parameter estimates successfully converge to the true values in the numerical experiment. Figure 11.1 is an image of the physical laboratory-scale fermenter.

The chapter is structured as follows: in section 11.1, we present a summary of the paper listed in Appendix A. In the paper, we present a model for growth of *Methylococcus capsulatus* (Bath) and chemical equilibrium in a laboratory-scale fermentor. In section 11.2, we present the CD-EKF for state estimation in nonlinear systems involving SDAEs. We apply the CD-EKF to the laboratory-scale fermenter in an open-loop numerical experiment. Finally, 11.3 summarises the work presented in the chapter.

11.1 Modelling and economic optimal control

The work presented in this section is a summary of the paper listed in Appendix A. In this paper, we present a growth model for SCP production of *M. capsulatus* and a CSTR model describing a laboratory-scale fermenter. The resulting model is a DAE model in the form

$$\begin{aligned} dx(t) &= f(x(t), y(t), u(t), \theta)dt, \qquad x(t_0) = x_0, && (11.1a)\\ 0 &= g(x(t), y(t), \theta). && (11.1b) \end{aligned}$$

The growth model is presented in section 3.5, as

$$\begin{aligned} \mathrm{CH_4 + O_2 + H_2O} &\longrightarrow \mathrm{CO_2 + 4\,H_3O^+}, && r_1, && (11.2a)\\ \mathrm{CH_4 + O_2} + \frac{2}{10}\,\mathrm{NH_4{}^+} &\longrightarrow \mathrm{CH_{1.8}O_{0.5}N_{0.2}} + \frac{15}{10}\,\mathrm{H_2O}, && r_2. && (11.2b) \end{aligned}$$

In addition to the growth kinetics, we include the chemical equilibrium reactions

$$\begin{aligned} \mathrm{H_2O\,(aq) + H_2O\,(aq)} &\rightleftharpoons \mathrm{H_3O^+(aq) + OH^-(aq)}, && K_{e,W}, && (11.3a)\\ \mathrm{NH_3(aq) + H_2O\,(aq)} &\rightleftharpoons \mathrm{NH_4{}^+(aq) + OH^-(aq)}, && K_{e,N}, && (11.3b)\\ \mathrm{CO_2(aq) + H_2O\,(aq)} &\rightleftharpoons \mathrm{H_2CO_3(aq)}, && K_{e,C1}, && (11.3c)\\ \mathrm{H_2CO_3(aq) + H_2O\,(aq)} &\rightleftharpoons \mathrm{HCO_3{}^-(aq) + H_3O^+(aq)}, && K_{e,C2}, && (11.3d)\\ \mathrm{HCO_3{}^-(aq) + H_2O\,(aq)} &\rightleftharpoons \mathrm{CO_3{}^{2-}(aq) + H_3O^+(aq)}, && K_{e,C3}, && (11.3e) \end{aligned}$$

as well as the strong acid and base reactions for nitric acid and sodium hydroxide to control pH-value in the reactor

$$\begin{aligned} \mathrm{HNO_3(aq) + H_2O\,(aq)} &\longrightarrow \mathrm{NO_3{}^-(aq) + H_3O^+(aq)}, && (11.4a)\\ \mathrm{NaOH\,(aq)} &\longrightarrow \mathrm{Na^+(aq) + OH^-(aq)}. && (11.4b) \end{aligned}$$

Figure 11.2 illustrates the gas-liquid CSTR model describing the laboratory-scale fermenter. We define an EOCP for optimal biomass production, as

$$\min_{[x(t),y(t)]_{t_0}^{t_f},\{u_k\}_{k=0}^{N-1}} \quad \phi = \phi_{X,\mathrm{eco}} + \phi_{u,\mathrm{eco}}, \qquad (11.5)$$

subject to

$$\begin{aligned} x(t_0) &= \hat{x}_{0|0}, && && (11.6a)\\ \frac{dx}{dt}(t) &= f(x(t), y(t), u(t), \theta),, && t \in [t_0, t_f], && (11.6b)\\ 0 &= g(x(t), y(t), \theta), && t \in [t_0, t_f], && (11.6c)\\ u(t) &= u_k, && t \in [t_k, t_{k+1}], \quad k \in \{0, 1, \dots, N-1\}. && (11.6d) \end{aligned}$$

Figure 11.1: Image of laboratory-scale fermenter for single-cell protein production.

The performance measure $\phi_{X,\text{eco}}$ is the SCP production revenue maximisation matrix described in (9.21)-(9.23), but where the Mayer term is modified to only include the CSTR biomass content. The performance measure $\phi_{u,\text{eco}}$ is the input cost minimisation metric described (9.18)-(9.20). We introduce a parametrisation of the algebraic variables, as

$$y(t) = 10^{-a(t)}, \tag{11.7}$$

such that the parametric variable, $a(t)$, corresponds to the negative logarithm of $y(t)$, i.e. $a_C(t) = -\log_{10}(y_C(t)) = pC$ for an algebraic component, C. This means, that for an algebraic variable describing the concentration of hydronium ion, we are describing the pH-value directly with the parametric variable, $a(t)$. We apply the direct simultaneous approach described in section 9.1 and solve the resulting NLP using IPOPT in Matlab. We compute model derivatives and formulate the NLP using the automatic differentiation and optimisation toolbox Casadi [108]. Figure 11.3 describes a set of key performance indicators for the solution to the EOCP. Figure 11.4 describes the states and algebraic variables for the solution to the EOCP.

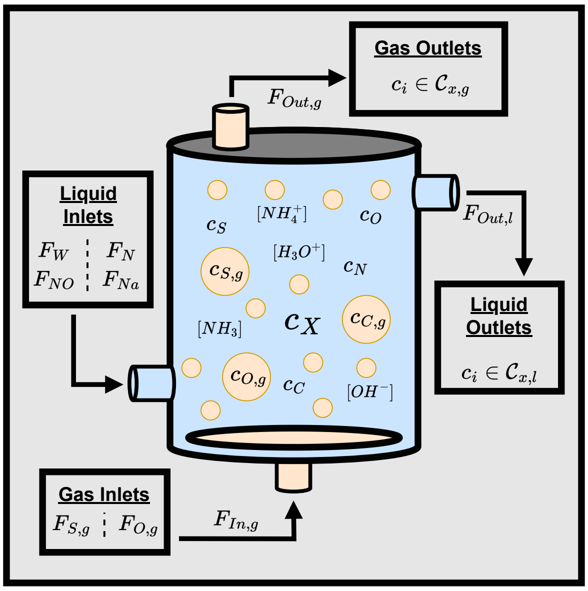


Figure 11.2: Illustration of the laboratory-scale continuous stirred tank reactor for single-cell protein production.

11.2 State estimation

In this section, we apply the CD-EKF as presented in Chapter 8 to the laboratory-scale bioreactor described in Appendix A. We introduce uncertainty in the DAE model by introducing the uncertain parameter for the maximum specific growth rate, as

$$d\boldsymbol{\mu}_{\max}(t) = \kappa_{\mu_{\max}} (\bar{\mu}_{\max} - \boldsymbol{\mu}_{\max}(t)) + \sigma_{\mu_{\max}} d\boldsymbol{\omega}_{\mu_{\max}}(t), \tag{11.8}$$

where $\kappa_{\mu_{\max}}$ is a scaling parameter for the drift term, $\bar{\mu}_{\max}$ is the nominal value for the maximum growth rate, $\sigma_{\mu_{\max}}$ is the diffusion parameter, and the process noise, $\boldsymbol{\omega}_{\mu_{\max}}(t)$, is a standard Wiener process, i.e. $d\boldsymbol{\omega}_{\mu_{\max}}(t) \sim \mathcal{N}(0, Idt)$. We introduce measurements of the pH-value and dissolved oxygen from the reactor at equidistant discrete sampling intervals of length T_s. From this, we formulate an SDAE model in the form

$$d\boldsymbol{x}(t) = f(\boldsymbol{x}(t), \boldsymbol{y}(t), u(t), \theta)dt + \sigma(\boldsymbol{x}(t), \boldsymbol{y}(t), \theta)d\boldsymbol{\omega}(t), \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \tag{11.9}$$

$$0 = g(\boldsymbol{x}(t), \boldsymbol{y}(t), \theta), \tag{11.10}$$

$$\boldsymbol{y}^m(t_k) = h^m(\boldsymbol{x}(t_k), \boldsymbol{y}(t_k), \theta) + \boldsymbol{v}(t_k, \theta), \qquad \boldsymbol{v}(t_k, \theta) \sim \mathcal{N}(0, R(\theta)). \tag{11.11}$$

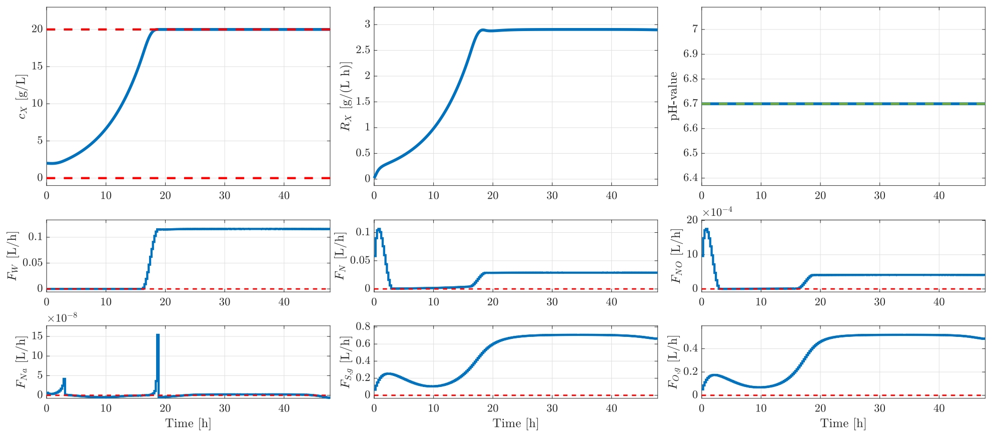


Figure 11.3: Key performance indicators for solution to the economic optimal control problem for single-cell protein production in a laboratory-scale continuous stirred tank reactor.

To improve the conditioning of the residual Jacobian, (8.22), applied in simulation and state estimation, we introduce the linear scaling described in 6.2, as

$$S_g = \operatorname{diag}(s_g), \qquad s_g = \left[10^7, 10^7, 10^4, 10^8, 10^{10}, 10^0, 10^0, 10^0\right], \tag{11.12a}$$
$$S_y = \operatorname{diag}(s_y), \qquad s_y = \left[10^0, 10^0, 10^0, 10^0, 10^0, 10^0, 10^0, 10^0\right]. \tag{11.12b}$$

We solve the EOCP formulated in Appendix A for a horizon of 48 hours. We apply the direct simultaneous approach and implicit Euler numerical integration scheme as described in Chapter 9 to formulate an NLP. We perform a numerical experiment implementing the open-loop control strategy computed as the solution the NLP. We measure pH and dissolved oxygen concentration in the reactor every 30 minutes, i.e. $T_s = 30$ minutes. For the numerical experiment, we define a simulation and an estimation model to simulate plant-model mismatch. We initialise the simulation

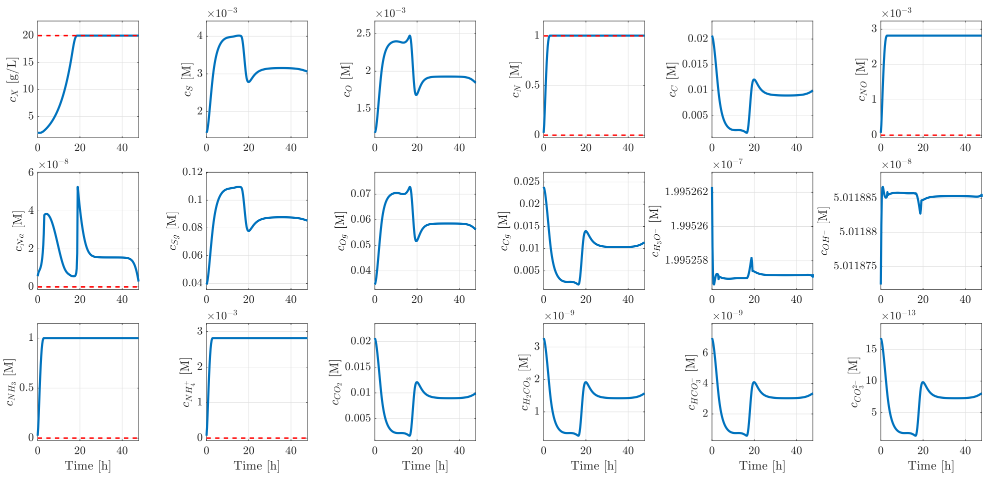


Figure 11.4: States for solution to the economic optimal control problem for single-cell protein production in a laboratory-scale continuous stirred tank reactor.

and estimation models, as

$$x_0 = \begin{bmatrix} 8.12 \cdot 10^{-2} \\ 1.43 \cdot 10^{-3} \\ 1.18 \cdot 10^{-3} \\ 2.37 \cdot 10^{-2} \\ 2.07 \cdot 10^{-2} \\ 6.67 \cdot 10^{-5} \\ 5.30 \cdot 10^{-8} \\ 3.92 \cdot 10^{-2} \\ 3.46 \cdot 10^{-2} \\ 2.39 \cdot 10^{-2} \\ 2.28 \cdot 10^{-2} \end{bmatrix}, \qquad p_{0|0} = \begin{bmatrix} 5.00 \cdot 10^{-4} \\ 5.00 \cdot 10^{-7} \\ 5.00 \cdot 10^{-7} \\ 1.00 \cdot 10^{-4} \\ 1.00 \cdot 10^{-5} \\ 1.00 \cdot 10^{-6} \\ 1.00 \cdot 10^{-12} \\ 1.00 \cdot 10^{-4} \\ 1.00 \cdot 10^{-4} \\ 1.00 \cdot 10^{-4} \\ 1.00 \cdot 10^{-2} \end{bmatrix}, \tag{11.13}$$

where the estimation model is initialised with state $\hat{x}_{0|0} = x_0 + 0.5x_0$ and state covariance $P_{0|0} = \text{diag}\left(p_{0|0}\right)$. The measurement covariances were chosen identically for the simulation and estimation models, as

$$R^s(\theta^s) = R^e(\theta^e) = \begin{bmatrix} 5.00 \cdot 10^{-2} & \\ & 1.00 \cdot 10^{-8} \end{bmatrix}, \tag{11.14}$$

where R^s is the measurement noise covariance for the simulation model and R^e is the measurement noise covariance for the estimation model. Table 11.1 list the parameters for the simulation and estimation models, respectively. Figure 11.5 describes the estimated states and algebraic variables including uncertainty for the numerical experiment. We see that the CD-EKF provides reliable estimates of the states and algebraic variables almost immediately and converges to the true values after between 1 and 10 hours. Figure 11.6 describes the estimate of the uncertain parameter, i.e. the maximum growth rate, and the measured variables, i.e. pH and dissolved oxygen concentration. Notably, the parameter estimate converges to the true value after around 10 hours, corresponding well to the observed convergence in the states and algebraic variables. The estimates of the measured variables converge after 1-2 hours. It was evident from the simulation study that the initial guess for the state covariance was of crucial importance to the convergence of the estimates. If the covariance was chosen too large, the estimates would converge very slowly, and if the covariances were chosen too small, the estimates would not converge at all.

11.3 Summary

In this chapter, we presented a summary of the paper listed in Appendix A and conducted a numerical experiment of state estimation for a system described by nonlinear SDAEs. In section 11.1, we presented a summary of the laboratory-scale fermentation process for SCP. We briefly outlined the model equations and described the EOCP. In section 11.2, we extended the model of the laboratory-scale fermenter by introducing

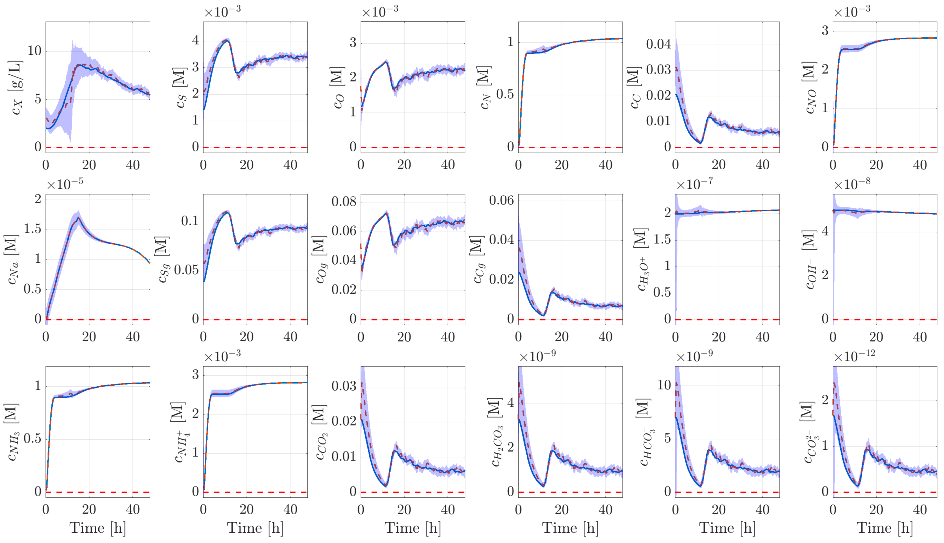


Figure 11.5: Estimates of state and algebraic variables in the numerical experiment for the laboratory-scale bioreator.

Table 11.1: Parameters in the simulation and estimation models in the numerical experiment of single-cell protein production the laboratory-scale bioreactor.

	Simulation model	Estimation model	
Symbol	Value	Value	Unit
$\bar{\mu}_{\max}$	$2.28 \cdot 10^{-1}$	$2.00 \cdot 10^{-1}$	1/h
$\kappa_{\mu_{\max}}$	1.00	$1.00 \cdot 10^{-1}$	-
$\sigma_{\mu_{\max}}$	$5.00 \cdot 10^{-2}$	$5.00 \cdot 10^{-2}$	-
α	$2.00 \cdot 10^{-2}$	$2.00 \cdot 10^{-2}$	-
δ	$2.00 \cdot 10^{-2}$	$2.00 \cdot 10^{-2}$	-
m	$9.80 \cdot 10^{-5}$	$9.80 \cdot 10^{-5}$	1/h
K_N	$1.30 \cdot 10^{-3}$	$1.30 \cdot 10^{-3}$	mol/L
$K_{N,ox}$	$3.30 \cdot 10^{-3}$	$3.30 \cdot 10^{-3}$	mol/L
K_S	$7.50 \cdot 10^{-5}$	$7.50 \cdot 10^{-5}$	mol/L
K_O	$5.50 \cdot 10^{-5}$	$5.50 \cdot 10^{-5}$	mol/L
$K_{e,W}$	$1.00 \cdot 10^{-14}$	$1.00 \cdot 10^{-14}$	-
$K_{e,N}$	$5.62 \cdot 10^{-10}$	$5.62 \cdot 10^{-10}$	-
$K_{e,C1}$	$1.58 \cdot 10^{-7}$	$1.58 \cdot 10^{-7}$	-
$K_{e,C2}$	$4.27 \cdot 10^{-7}$	$4.27 \cdot 10^{-7}$	-
$K_{e,C3}$	$4.79 \cdot 10^{-11}$	$4.79 \cdot 10^{-11}$	-
$c_{\text{In},N}$	5.88	5.88	mol/L
$c_{\text{In},Na}$	1.00	1.00	mol/L
$c_{\text{In},NO}$	1.00	1.00	mol/L
c_{In,S_g}	$1.90 \cdot 10^{-1}$	$1.90 \cdot 10^{-1}$	mol/L
c_{In,O_g}	$1.90 \cdot 10^{-1}$	$1.90 \cdot 10^{-1}$	mol/L
k_La_S	$3.89 \cdot 10^{2}$	$3.89 \cdot 10^{2}$	1/h
k_La_O	$3.71 \cdot 10^{2}$	$3.71 \cdot 10^{2}$	1/h
k_La_C	$3.26 \cdot 10^{2}$	$3.26 \cdot 10^{2}$	1/h
H_S^{pc}	$7.05 \cdot 10^{2}$	$7.05 \cdot 10^{2}$	(atm L)/mol
H_O^{pc}	$7.59 \cdot 10^{2}$	$7.59 \cdot 10^{2}$	(atm L)/mol
H_C^{pc}	$2.99 \cdot 10^{2}$	$2.99 \cdot 10^{2}$	(atm L)/mol
T	$3.15 \cdot 10^{2}$	$3.15 \cdot 10^{2}$	K
V	1.00	1.00	L

uncertainty in the parameter describing the maximum growth rate. We preformed an open-loop numerical experiment applying the resulting SDAE model and applied the state estimation method described in Chapter 8. We demonstrated that all estimates computed by the CD-EKF converged to the true values of the state after around 10 hours in the simulation.

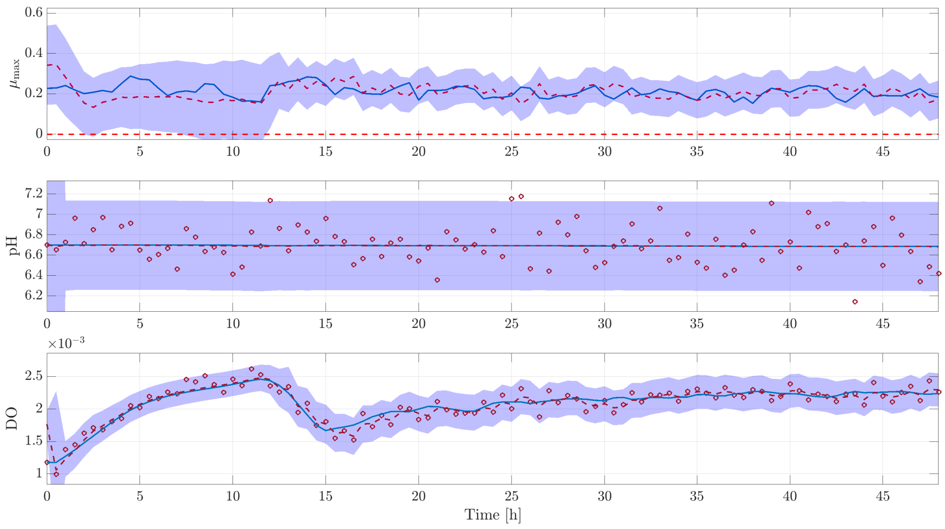


Figure 11.6: Estimates of uncertain parameter and measured variables in the numerical experiment for the laboratory-scale bioreator.

Part V

Controller Performance Quantification and Tuning

CHAPTER 12 High-Performance Monte Carlo Simulation

This chapter is a summary of the publication listed in Appendix C. In this paper, we present a high-performance Monte Carlo simulation toolbox for tuning and uncertainty quantification (UQ) for closed-loop nonlinear systems involving stochastic differential equations (SDEs). We achieve large-scale computational feasibility of the Monte Carlo approach by parallel implementation in C for shared memory architectures. We present a test system of single-cell protein (SCP) production in a fed-batch bioreactor and test four different controller for optimal biomass production; 1) an open-loop bang-bang control strategy as described in [83], 2) a proportional-integral-derivative (PID) controller, a PID control with clipping, and a nonlinear model predictive controller (NMPC). We perform 30,000 closed-loop simulations applying the open-loop control strategy in approximately 1 second and 30,000 closed-loop simulations tuning the two PID control strategies in approximately 8 second. We perform 1,000 closed-loop simulations for the NMPC in approximately 30 minutes. We show close to linear scaling on a single NUMA node with 16 cores and a scale-up of 27.3 times on two NUMA nodes with a total of 32 cores. The toolbox has been applied in a number of applications already for controller performance in diabetes treatment [109, 110], for performance quantification of optimisation and control algorithms [111, 112], and for controller matching [113].

The chapter is structured as follows: in section 12.1, we present formulation and implementation of a high-performance Monte Carlo simulation toolbox for UQ in closed-loop systems. In section 12.2, we present a numerical study of the performance of four controller; an open-loop controller, two PIDs, and an NMPC, on a fed-batch bioreactor. We perform 30,000 Monte Carlo simulation for the open-loop and PID controllers and 1,000 simulations for the NMPC. We quantify the performances of the controllers using the numerical simulation results, i.e. UQ. Finally, in section 12.3, we summarise the work presented in the chapter.

12.1 Closed-loop simulations

In this section, we consider closed-loop simulation of nonlinear systems involving SDEs in the form

$$
\begin{aligned}
d\boldsymbol{x}(t) &= f(t, \boldsymbol{x}(t), u(t), d(t), \theta)dt + \sigma(t, \boldsymbol{x}(t), u(t), \theta)d\boldsymbol{\omega}(t), && (12.1a)\\
\boldsymbol{z}(t) &= g^m(t, \boldsymbol{x}(t), \theta), && (12.1b)\\
\boldsymbol{y}^m(t_k) &= h^m(t_k, \boldsymbol{x}(t_k), \theta) + \boldsymbol{v}(t_k, \theta), && (12.1c)
\end{aligned}
$$

where

$$
\boldsymbol{x}(t_0) \sim \mathcal{N}(\bar{x}_0, P_0), \qquad d\boldsymbol{\omega}(t) \sim \mathcal{N}(0, Idt), \qquad \boldsymbol{v}(t_k, \theta) \sim \mathcal{N}(0, R(\theta)), \qquad (12.2)
$$

and where $\boldsymbol{x}(t)$ are states, $u(t)$ are manipulated variables, $d(t)$ are unmeasured disturbances, $\boldsymbol{y}^m(t_k)$ are measured variables, $\boldsymbol{z}(t)$ are output variables, and $\theta)$ are parameters. The process noise $\boldsymbol{\omega}(t)$ is a standard Wiener process and $\boldsymbol{v}(t_k, \theta)$ is normally distributed measurement noise. $f(\cdot)$ is the state drift function, $\sigma(\cdot)$ is the state diffusion function, $g^m(\cdot)$ is the output function, and $h^m(\cdot)$ is the measurement function. We describe discretised formulations of the system and controller and present a compact description of closed-loop systems using these formulations.

12.1.1 System formulation

We describe the system in discrete time, using the explicit-explicit or implicit-explicit numerical integration schemes for SDE models in the form presented in (12.1), as

$$
\begin{aligned}
x_{k+1} &= \Psi(t_k, x_k, u_k, d_k, \omega_k, \theta), && (12.3a)\\
y_k &= h^m(t_k, x_k, \theta), && (12.3b)\\
z_k &= g^m(t_k, x_k, \theta), && (12.3c)
\end{aligned}
$$

where $\Psi(\cdot)$ is the chosen numerical integration scheme and ω_k is a set of N_t noise realisation of the diffusion $\Delta\omega_k \sim \mathcal{N}(0, I\Delta t)$, where N_t is the number of internal steps within each sampling time of the simulation.

12.1.2 Controller formulation

We describe the controller in terms of three base operations; state estimation, control, and prediction. We consider controllers in the form

$$
\begin{aligned}
x_{k+1}^c &= \kappa\left(t_k, x_k^c, y_{k+1}^m, u_k, \theta\right), && (12.4a)\\
u_k &= \lambda\left(t_k, x_k^c, \theta\right), && (12.4b)\\
z_k^{m,c} &= \mu\left(t_k, x_k^c, \theta\right), && (12.4c)
\end{aligned}
$$

where x_k^c is the filtered state estimate, y_k^m is a measurement taken at time $t = t_k$, and $z_k^{m,c}$ are predictions. $\kappa(\cdot)$ is the state estimator, $\lambda(\cdot)$ is the controller, and $\mu(\cdot)$ is the predictor. We apply four control strategies in the closed-loop simulations; open-loop, PID, PID with clipping, and NMPC.

Open-loop control

We consider an open-loop controller in the form

$$\begin{aligned} u_k &= \lambda(t_k, x_k^c, \theta) && (12.5a) \\ &= \bar{u}_k, && (12.5b) \end{aligned}$$

where $\bar{u}_k$ is the pre-computed optimal open-loop bang-bang control strategy described in [83]. The state estimator, $\kappa(\cdot)$, and predictor, $\mu(\cdot)$, are not defined for the open-loop controller.

PID

We consider a PID controller in the form

$$\begin{aligned} u_k &= \lambda(t_k, x_k^c, \theta) && (12.6a) \\ &= \bar{u}_k + P_k + I_k + D_k. && (12.6b) \end{aligned}$$

$\bar{u}_k$ is the nominal control action and

$$\begin{aligned} P_k &= K_p e_k, && (12.7a) \\ I_k &= I_{k-1} + T_s K_i e_k, && (12.7b) \\ D_k &= -\frac{K_d}{T_s}\left(y_k^{m,F} - y_{k-1}^{m,F}\right), && (12.7c) \end{aligned}$$

where $e_k = \bar{y}_k^m - y_k^{m,F}$ is the tracking error, $\bar{y}_k^m$ is the target, and $y_k^{m,F}$ is a filtered measurement computed by the discrete low-pass filter

$$\begin{aligned} y_k^{m,F} &= \kappa(t_{k-1}, y_{k-1}^{m,F}, y_k^m, u_{k-1}, \theta) && (12.8a) \\ &= (1-\alpha) y_{k-1}^{m,F} + \alpha y_k^m. && (12.8b) \end{aligned}$$

The predictor, $\mu(\cdot)$, is not defined for the PID controller.

PID with clipping

We consider a PID controller with clipping in the form

$$\begin{aligned} u_k &= \lambda(t_k, x_k^c, \theta) && (12.9a) \\ &= \max\{u_{\min}, \min\{u_{\max}, \tilde{u}_k\}\}, \qquad \tilde{u}_k = \bar{u}_k + P_k + I_k + D_k, && (12.9b) \end{aligned}$$

where $u_{\min}$ and $u_{\max}$ are the minimum and maximum inputs, respectively. The proportional term, P_k, integral term, I_k, and derivative term D_k, are computed as presented in (12.7).

NMPC

In the NMPC, we consider a controller, $\lambda(\cdot)$, defined by the OCP

$$\min_{[x(t);u(t)]_{t_k}^{t_{k+N}}} \quad \phi = \alpha_z \phi_z + \alpha_{\Delta u} \phi_{\Delta u}, \tag{12.10}$$

subject to

$$\begin{aligned}
x(t_k) &= \hat{x}_{k|k}, && && && (12.11a)\\
\frac{dx}{dt} &= f(t, x(t), u(t), d(t), \theta), && t_k \leq t \leq t_{k+N}, && && (12.11b)\\
z^m(t) &= g^m(t, x(t), \theta), && t_k \leq t \leq t_{k+N}, && && (12.11c)\\
u(t) &= u_{k+j|k}, && t_k \leq t \leq t_{k+N}, && j \in \mathcal{N}_c, && (12.11d)\\
d(t) &= d_{k+j|k}, && t_k \leq t \leq t_{k+N}, && j \in \mathcal{N}_c, && (12.11e)\\
u_{\min} &\leq u_{k+j|k} \leq u_{\max}, && && j \in \mathcal{N}_c, && (12.11f)\\
\Delta u_{\min} &\leq \Delta u_{k+j|k} \leq \Delta u_{\max}, && && j \in \mathcal{N}_c, && (12.11g)
\end{aligned}$$

where $\mathcal{N}_c = \{0, 1, \ldots, N_c - 1\}$ defines the N_c step control horizon of discrete sampling intervals of length T_s. We apply the direct simultaneous approach and discretise the system dynamics and integrals using the right rectangular rule, i.e. implicit Euler. We apply IPOPT to solve the resulting NLP. We consider the the state estimator, $\kappa(\cdot)$, to be the CD-EKF as described in Chapter 7 the publication. The predictions, $\mu(\cdot)$, are computed by the CD-EKF given the optimal input trajectory computed as the solution to the NLP.

12.2 Numerical experiments

In this section, we present numerical experiments. We present an example system of SCP production in a fed-batch reactor. We perform a series of scaling experiments to evaluate the performance of the toolbox. We conduct numerical experiments testing the different control strategies. We perform 30,000 closed-loop simulations for the open-loop and PID strategies and 1,000 closed-loop simulation for the NMPC.

12.2.1 Fed-batch reactor

We consider a fed-batch reactor on the form presented in the technical report listed in Appendix G. We consider an SDE model in the form

$$\begin{aligned}
\frac{dV}{dt}(t) &= e^T F(t) + \sigma_V d\boldsymbol{\omega}_V(t), && V(t_0) = V_0, && (12.12a)\\
\frac{dn}{dt}(t) &= C_{\text{In}} F(t) + R(c)V + \sigma_n \boldsymbol{\omega}_n(t), && n(t_0) = n_0, && (12.12b)
\end{aligned}$$

where $V(t)$ is the reactor volume and $n(t) = [n_X, n_S]^T$ are the masses of biomass and substrate in the reactor, respectively, and $F(t) = [F_W(t), F_S(t)]^T$ are the manipulated input flow-rates. σ_V is the diffusion parameter for the volume and σ_n is a diagonal matrix of diffusion parameters for the masses. C_{In} are the inlet concentrations and $R(c)$ are the productions.

Stoichiometry and kinetics

We describe the growth of biomass in the fed-batch reactor by the stoichiometry

$$1.777S \longrightarrow X, \qquad\qquad r(c), \tag{12.13}$$

where S is the substrate, X is the biomass, and $r(c)$ is the rate of reaction. The system is described by the stoichiometric matrix

$$S = \begin{array}{cc} \mathrm{X} & \mathrm{S} \\ \left[1\right. & \left.-1.777\right] \end{array} \; \mathrm{r}_1 \quad . \tag{12.14}$$

We rate of reaction is

$$r(c) = \mu(c) c_X, \tag{12.15}$$

where $\mu(c)$ is the growth rate and c_X is the biomass concentration in the reactor. The growth rate is

$$\mu(c) = \mu_{\max} \mu_S(c), \tag{12.16}$$

where $\mu_{\max}$ is the maximum growth rate and $\mu_S(c)$ is the specific growth rate on substrate. We apply Monod-Haldane growth kinetics to describe the specific growth rate on substrate, as

$$\mu_S(c) = \frac{c_S}{K_S + c_S + c_S^2/K_{S,I}}, \tag{12.17}$$

where K_S is the saturation constant and $K_{S,I}$ is the inhibition constant. We note that the Monod-Haldane expression gives rise to an optimal substrate concentration for growth, as

$$c_S^* = \sqrt{K_S K_{S,I}}. \tag{12.18}$$

We elaborate further on the optimal substrate concentration in Chapter 3.

12.2.2 Scaling

Monto Carlo simulation provides an excellent problem for parallelisation, as each individual simulation is independent of all other simulation, i.e. a so-called embarrassingly parallel problem. Due to this, we expect the problem to scale almost linearly with the number of cores. We perform numerical scaling experiments on one and two NUMA nodes with each 16 cores. We run 10,000 closed-loop simulations in each experiment and benchmark the results against a serial implementation in Matlab of the same system. We observe a 2300 times speed-up on 32 cores for the C implementation compared to the serial Matlab implementation and a 27.3 times speed-up relative to a a serial C version. We observe almost linear scaling on a single NUMA node of 16 cores and a slight decrease in scaling on two NUMA nodes. The decrease in scaling on several NUMA nodes is expected, as the implementation is not optimised for parallelisation on multiple NUMA nodes. Figure 12.1 illustrates the results of the numerical scaling experiments.

12.2.3 PID tuning

We apply the Monte Carlo simulation toolbox to choose optimal values of the gains of the PID controller; K_p, K_i, and K_d. We generate 1,000 process noise realisations. We choose 101 equidistant values for each constant $K_j \in [0, 100]$ for $j \in \{p, i, d\}$. We conduct a total of $3 \times 101,000$ closed-loop simulations for the chosen PID tunings and determine the optimal tuning in terms of the maximal biomass concentrations at the final time. We optimise the gains sequentially, i.e. a type of coordinate descent approach. The closed-loop simulations has a wall-time of ~ 7.5 seconds. The optimal PID gains are

$$K_p = 85, \qquad K_i = 3, \qquad K_d = 0. \tag{12.19}$$

Figure 12.2 describes the tuning profiles generates by the Monte Carlo simulations.

12.2.4 Performance quantification

We apply the Monte Carlo simulation toolbox to determine performance distributions for the different control strategies. We perform 30,000 simulations applying the open-loop control strategy in 0.78 seconds and plot the biomass in the reactor at the final time for all realisations. We perform 30,000 closed-loop simulations applying a PID controller with sub-optimal tuning in 0.77 seconds and plot the biomass in the reactor at the final time for all realisations. We perform 30,000 closed-loop simulations applying a PID controller with the optimal tuning determined in this work in 0.76 seconds and plot the biomass in the reactor at the final time for all realisations. We perform 1,000 closed-loop simulations applying the NMPC in ~ 30 minutes and plot the biomass in the reactor at the final times for all realisations. We do not observe the same scaling profile for the NMPC, as dynamic memory-allocation is being done inside the optimiser, i.e. IPOPT. Figure 12.3 presents the UQ of the different control strategies applied to the fed-batch reactor example.

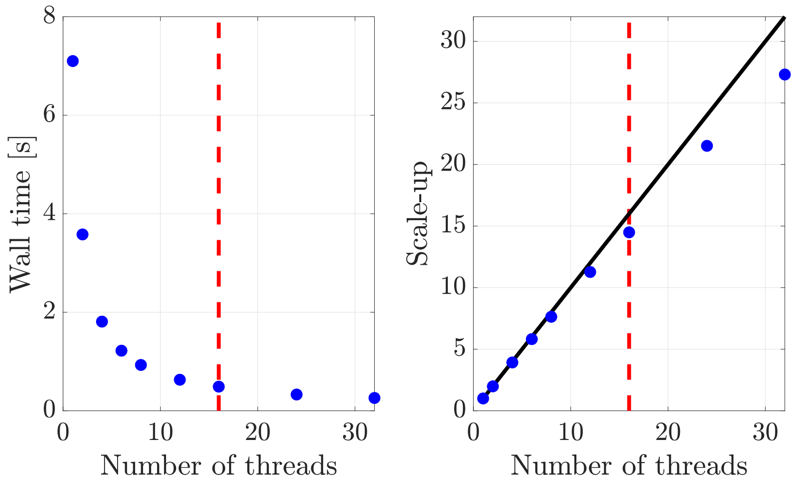


Figure 12.1: Results of the numerical experiments to determine the scaling of the parallel Monte Carlo simulation toolbox. Each numerical experiment runs 10,000 simulations for the example fed-batch reactor for single-cell protein production.

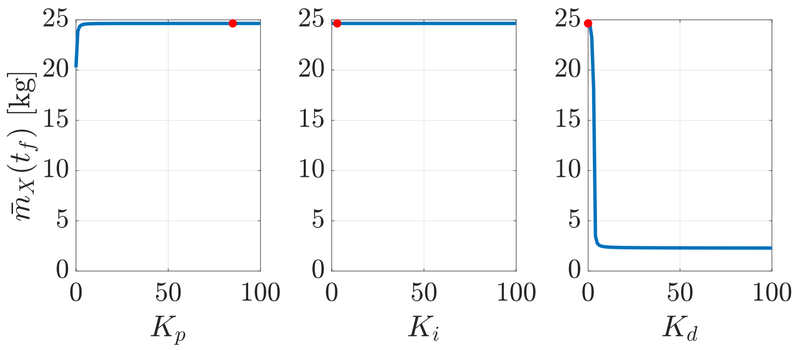


Figure 12.2: Optimal tuning for PID controller determined from 303,000 simulation using the Monte Carlo simulation toolbox.

12.3 Summary

In this chapter, we presented a summary of the publication listed in Appendix C. We presented a formulation of Monte Carlo simulation for closed-loop systems. We presented a high-performance implementation of a Monte Carlo simulation toolbox in C. We used Openmp to parallelise the independent simulations. We presented a scaling experiment, demonstrating that the toolbox has close to linear scaling on 16

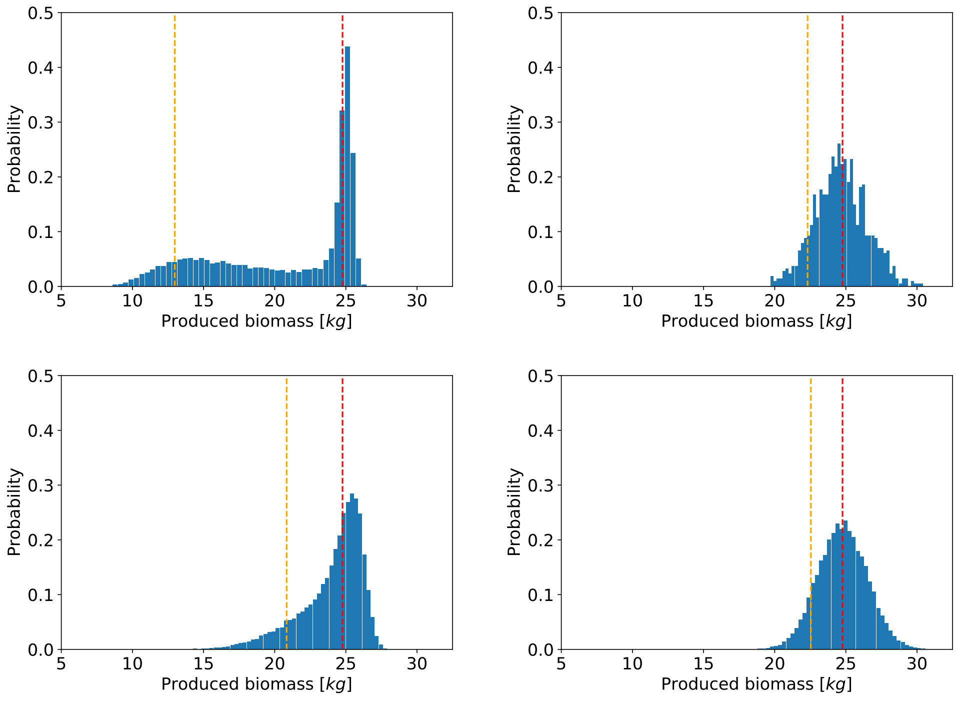


Figure 12.3: Distributions resulting from the Monte Carlo simulations applying open-loop, PID, tuned PID, and NMPC. **Top left:** Distribution of biomass at final time for 30,000 closed-loop simulation applying the open-loop control strategy. The simulations took 0.78 seconds. **Top right:** Distribution of biomass at final time for 1,000 closed-loop simulations applying NMPC. The simulations took approximately 30 minutes. **Bottom left:** Distribution of biomass at final time for 30,000 closed-loop simulations applying an untuned PID controller. The simulations took 0.77 seconds. **Bottom right:** Distribution of biomass at final time for 30,000 closed-loop simulations applying an optimally tunes PID controller. The simulations took 0.76 seconds.

cores on a single NUMA node and 27.3 times scale-up on 32 cores across two individual NUMA nodes. We formulated a fed-batch bioreactor example for numerical experiments. On the bioreactor, we conducted 303,000 closed-loop simulations in $\sim$ 7.5 seconds to determine an optimal tuning for a PID controller. We performed UQ on four different control strategies applied to the bioreactor example. We performed 30,000 simulations applying open-loop and two PID control strategies in less than 1 second and 1,000 closed-loop simulations applying NMPC in $\sim$ 30 minutes.

Finally, we quantified the performance of the control strategies statistically from the simulation study.

Part VI

Conclusions and Suggestions for Future Work

CHAPTER 13

Conclusion

In this chapter, we present conclusions and suggestions for future work based on the work presented in this thesis.

13.1 Conclusions

In this section, we present conclusions for the work presented in this thesis. Firstly, we recall the main objectives of the work as described in the introduction to the thesis, as

1. describe growth models for *Methylococcus capsulatus* (Bath) for the purpose of single-cell protein production,
2. describe a dynamical model for the U-loop bioreactor,
3. formulate and test state estimation methods in nonlinear systems involving stochastics,
4. formulate a nonlinear model predictive control system for direct optimising of economic performance measures,
5. and formulate methods of investigating performance of nonlinear control solutions.

In this work, we presented a set of growth models for the methanotropic bactria *Methylococcus capsulatus* (Bath). We presented model based on methanol as carbon source an nitric acid as nitrogen source and we presented a model based on methane as carbon source and ammonia as nitrogen source. Based on a metabolic study of the microorganism, we presented a growth model based on methane as carbon source, which describes anabolic reactions from several nitrogen sources; ammonia, nitrite, nitrate, and molecular nitrogen. Additionally, we discussed the model formulation for catabolic repression in this model and suggested updates for the model formulation to better describe the dynamics. We presented a reduced model based on the same metabolic study with methane and carbon source and ammonium as nitrogen source. Furthermore, we presented models describing chemical equilibrium reactions in the context of single-cell protein production. We presented models describing a novel U-loop bioreactor and a laboratory-scale continuous stirred tank bioreactor (CSTR). We formulate continuous-discrete nonlinear systems involving SDEs and SDAE describing SCP production processes.

We presented state estimation methods for the purpose of monitoring and control in the nonlinear systems presented in this work. We presented four methods for state estimation in continuous-discrete nonlinear systems involving SDEs; the continuous-discrete extended Kalman filter (CD-EKF), unscented Kalman filter (CD-UKF), ensemble Kalman filter (CD-EnKF), and a continuous-discrete particle filter (CD-PF). We discussed advantages and disadvantages of each method, and where they might be applied most successfully. Additionally, we presented the CD-EKF for continuous-discrete nonlinear systems involving SDAEs. This was especially relevant when chemical equilibrium reactions were introduced in the system descriptions to include measurement information from common online sensors, e.g. pH-value.

We presented an economic nonlinear model predictive control (ENMPC) formulation involving the state estimation methods discussed previously, as well as economic optimal control problems (EOCP). We presented objectives in which the productivity of the production process is directly optimised. The SCP production process resulted in a stiff and unstable system. We applied a direct simultaneous approach to handle system instability and discretised the continuous-time OCP with an implicit Euler numerical integration scheme, i.e. right rectangular rule, to handle stiffness of the high-dimensional system. We demonstrated in the control study, that the ENMPC could stabilise start-up and maintain high productivity in the U-loop bioreactor. Furthermore, we demonstrated that the ENMPC is real-time feasible for control of the U-loop bioreactor.

We presented numerical experiments for modelling, economic optimal control, state estimation using the CD-EKF, and a closed-loop simulation of ENMPC for SCP production in a U-loop bioreactor. In the U-loop bioreactor experiments, we observed stable start-up and high productivity over the control horizon. Additionally, we presented numerical experiments of modelling, economic optimal control, and state estimation for SCP production in a laboratory-scale bioreactor including chemical equilibrium reactions. In the numerical experiments, we observed good estimates of states, algebraic variables, and modelled parameters including available measurements of dissolved oxygen and pH-value.

Finally, we presented a high-performance Monte Carlo simulation toolbox for quantification of controller performance in closed-loop systems. The toolbox was implemented in C and applied Openmp for parallel computing in high-performance closed-loop simulation with control applied to stochastic systems. We applied the toolbox on a numerical experiment for SCP production in an FBR and show a 2300 times speed-up relative to a benchmark serial Matlab implementation. We applied the Monte Carlo simulation toolbox for uncertainty quantification of controller performance for open-loop, PID, PID with clipping, and NMPC control strategies applied to the fed-batch bioreactor example.

13.2 Suggestions for future work

In this section, we present suggestions for future work based on, and related to, the work presented in this thesis on SCP production. We focus particularly on suggestions regarding

- modelling,
- estimation,
- and implementation and testing.

Modelling

In the application of monitoring and control systems, it is important to incorporate as much of the measurement information as possible in the reconstruction of the model states. It is for this reason, that we have included chemical equilibrium reactions in the model. pH-value is a common and readily available online measurement in chemical and biochemical systems, and it should therefore also be included in the model. We suggest that models describing chemical equilibrium reactions are investigated to formulate models involving SDAEs. In this work, we have focused on models describing chemical equilibrium reactions by applying thermodynamic equilibrium constants, but we suggest to also investigate formulations based on minimisation of Gibbs free energy. Consider the optimisation problem for the minimisation of Gibbs free energy in chemical equilibrium systems

$$\min_{n} \quad \phi = G(n), \tag{13.1}$$

subject to

$$An = b, \tag{13.2a}$$

$$n \geq 0, \tag{13.2b}$$

where $G(n)$ is the Gibbs free energy in the system for the mole numbers of the modelled components, n. We may formulate a set of algebraic equations arising from the 1st order necessary optimality conditions, as

$$0 = \nabla \mathcal{L}(n, \mu), \tag{13.3a}$$

$$0 = An - b, \tag{13.3b}$$

and then use a solver with a nonnegativity constraint on the mole number, n, to ensure that the nonnegativity constraint is satisfied in the solution, i.e. $n \geq 0$. It is suggested that a close dialogue is maintained with the process experts in the development of new fermentation models, such that they accurately reflect the pilot-scale reactor setup operated in Kalundborg by Unibio A/S. The pilot-scale U-loop bioreactor is ideal for testing the feasibility of monitoring and control systems.

Estimation

We have formulated and implemented a number of state estimation algorithms in this work. However, they have not been yet implemented as tools which can be directly utilised by operators. Monitoring tools utilising state estimation based on available online and offline measurements could potentially be of great benefit to process operators. Instead of basing decisions on measured variables only, monitoring tools would give operators information about internal unmeasured variables as well, e.g. substrate concentrations and concentrations of inhibiting components, not only at the present time, but also predictively as the process evolves in time. We suggest that such tools are utilised actively in process operation. Furthermore, we suggest that parameter estimation methods based on the state estimators are investigated, and that the laboratory-scale setup is used for identification of kinetic parameters in the growth model. We suggest formulating methods for maximum likelihood estimation (MLE) and maximum a posteriori estimation (MAPE) for the estimation of kinetic parameters.

Implementation and testing

Finally, we suggest that the methods and control systems developed in this work are implemented and tested on physical bioreactors. The results suggest that the control solutions can increase productivity and stability of the U-loop bioreactor significantly. Given a model which accurately reflect the pilot-scale reactor setup in Kalundborg, this would be an ideal location to test both monitoring and control systems. In this context, the importance of closed dialogue with process operators and experts should again be emphasised in the development of models and control systems which accurately reflect process dynamics and the demands of operators.

Bibliography

[1] United Nation - Department of Economic and Social affairs. *Sustainable developement goals (SDGs).* February 2023. URL: https://sdgs.un.org/goals.

[2] Giulia Sesini, Cinzia Castiglioni, and Edoardo Lozza. "New trends and patterns in sustainable consumption: A systematic review and research agenda." In: *Sustainability* 12.15 (2020), page 5935. DOI: 10.3390/su12155935.

[3] Christina Hartmann and Michael Siegrist. "Consumer perception and behaviour regarding sustainable protein consumption: A systematic review." In: *Trends in Food Science & Technology* 61 (2017), pages 11–25. DOI: 10.1016/j.tifs.2016.12.006.

[4] Food and Agriculture Organization of the United Nations (FAO). *FAOSTAT.* January 2023. URL: https://www.fao.org.

[5] David Fraser. "Could animal production become a profession?" In: *Livestock Science* 169 (2014), pages 155–162. DOI: 10.1016/j.livsci.2014.09.017.

[6] S. R. L. Smith. "Single cell protein." In: *Philosophical Transactions of the Royal Society of London. B, Biological Sciences* 290.1040 (1980), pages 341–354.

[7] Jeremy BC Jackson, Michael X Kirby, Wolfgang H Berger, Karen A Bjorndal, Louis W Botsford, Bruce J Bourque, Roger H Bradbury, Richard Cooke, Jon Erlandson, James A Estes, et al. "Historical overfishing and the recent collapse of coastal ecosystems." In: *science* 293.5530 (2001), pages 629–637. DOI: 10.1126/science.1059199.

[8] Naglaa F Soliman, Dalia MM Yacout, Mahmoud A Hassaan, et al. "Responsible fishmeal consumption and alternatives in the face of climate changes." In: *International Journal of Marine Science* 7.15 (2017). DOI: 10.5376/ijms.2017.07.0015.

[9] A.C. Fanatico, K. Arsi, I. Upadhyaya, J. Morales Ramos, D. Donoghue, and A.M. Donoghue. "Sustainable Fish and Invertebrate Meals for Methionine and Protein Feeds in Organic Poultry Production1." In: *Journal of Applied Poultry Research* 27.4 (2018), pages 437–448. DOI: 10.3382/japr/pfy037.

[10] Elizabeth Barona, Navin Ramankutty, Glenn Hyman, and Oliver T Coomes. "The role of pasture and soybean in deforestation of the Brazilian Amazon." In: *Environmental Research Letters* 5.2 (2010), page 024002. DOI: 10.1088/1748-9326/5/2/024002.

[11] M. Van der Spiegel, M. Y. Noordam, and H. J. Van der Fels-Klerx. “Safety of novel protein sources (insects, microalgae, seaweed, duckweed, and rapeseed) and legislative aspects for their application in food and feed production.” In: *Comprehensive reviews in food science and food safety* 12.6 (2013), pages 662–678. DOI: 10.1111/1541-4337.12032.

[12] María-José Sánchez-Muros, Fernando G Barroso, and Francisco Manzano-Agugliaro. “Insect meal as renewable source of food for animal feeding: a review.” In: *Journal of Cleaner Production* 65 (2014), pages 16–27. DOI: 10.1016/j.jclepro.2013.11.068.

[13] Harinder PS Makkar, Gilles Tran, Valérie Heuzé, and Philippe Ankers. “State-of-the-art on use of insects as animal feed.” In: *Animal feed science and technology* 197 (2014), pages 1–33.

[14] R. A. Leng, J. H. Stambolie, and R. Bell. “Duckweed-a potential high-protein feed resource for domestic animals and fish.” In: *Livestock Research for Rural Development* 7.1 (1995), page 36.

[15] Jay J Cheng and Anne-M Stomp. “Growing duckweed to recover nutrients from wastewaters and for production of fuel ethanol and animal feed.” In: *Clean–Soil, Air, Water* 37.1 (2009), pages 17–26. DOI: 10.1002/clen.200800210.

[16] AT Nasseri, S Rasoul-Amini, MH Morowvat, Y Ghasemi, et al. “Single cell protein: production and process.” In: *American Journal of food technology* 6.2 (2011), pages 103–116. DOI: 10.3923/ajft.2011.103.116.

[17] Gour Suman, Mathur Nupur, Singh Anuradha, and Bhatnagar Pradeep. “Single cell protein production: a review.” In: *Int. J. Curr. Microbiol. App. Sci* 4.9 (2015), pages 251–262.

[18] P. Ravindra. “Value-added food:: Single cell protein.” In: *Biotechnology advances* 18.6 (2000), pages 459–479. DOI: 10.1016/S0734-9750(00)00045-8.

[19] John Villadsen, Jens Nielsen, and Gunnar Lidén. *Bioreaction engineering principles*. Springer Science+Business Media, 2011. ISBN: 9781441996879.

[20] S. R. Tannenbaum, R. I. Mateles, and G. R. Capco. “Processing of bacteria for production of protein concentrations.” In: *World protein resources*. Edited by Aaron M. Altschul. Volume 57. Adcances in chemistry, 1966, pages 254–260. ISBN: 9780841200586. DOI: 10.1021/ba-1966-0057.ch018.

[21] Allen I. Laskin. “Chapter 7 - Single cell protein.” In: volume 1. Annual reports on fermentation processes. Elsevier, 1977, pages 151–180. DOI: 10.1016/B978-0-12-040301-1.50012-9.

[22] R. B. Vasey and K. A. Powell. “Single-cell protein.” In: *Biotechnology and Genetic Engineering Reviews* 2.1 (1984), pages 285–311. DOI: 10.1080/02648725.1984.10647802.

[23] R. I. Mateles and S. R. Tannenbaum. “Single-cell protein.” In: *Economic botany* 22 (1968), pages 42–50. DOI: 10.1007/BF02897743.

[24] GL Solomons and John H Litchfield. "Single cell protein." In: *Critical Reviews in Biotechnology* 1.1 (1983), pages 21–58. DOI: 10.3109/07388558309082578.

[25] Anneli Ritala, Suvi T Häkkinen, Mervi Toivari, and Marilyn G Wiebe. "Single cell protein—state-of-the-art, industrial landscape and patents 2001–2016." In: *Frontiers in microbiology* 8 (2017), page 2009. DOI: 10.3389/fmicb.2017.02009.

[26] Richard S Hanson and Thomas E Hanson. "Methanotrophic bacteria." In: *Microbiological reviews* 60.2 (1996), pages 439–471. DOI: 10.1128/mr.60.2.439-471.1996.

[27] Unibio A/S. *Amino acid profile.* February 2023. URL: https://www.unibio.dk/end-product/amino-acid-profile/.

[28] Christian Lieven, Leander A. H. Petersen, Sten Bay Jørgensen, Krist V. Gernaey, Markus J. Herrgard, and Nikolaus Sonnenschein. "A genome-scale metabolic model for Methylococcus capsulatus (Bath) suggests reduced efficiency electron transfer to the particulate methane monooxygenase." In: *Frontiers in microbiology* 9 (2018), page 2947. DOI: 10.3389/fmicb.2018.02947.

[29] Leander A. H. Petersen. "Single cell protein production in U-loop bioreactors: fundamentals, modeling and control." PhD thesis. Technical University of Denmark, 2019.

[30] Ebbe Busch Larsen. "U-shape and/or nozzle-u-loop fermentor and method of carrying out a fermentation process." WO2000070014A1.

[31] Oscar Andreés Prado-Rubio, John Bagterp Jørgensen, and Sten Bay Jørgensen. "Systematic model analysis for single cell protein (SCP) production in a U-loop reactor." In: *Computer aided chemical engineering* 28 (2010), pages 319–324. DOI: 10.1016/S1570-7946(10)28054-9.

[32] Leander A. H. Petersen, John Villadsen, Sten Bay Jørgensen, and Krist V. Gernaey. "Mixing and mass transfer in a pilot scale U-loop bioreactor." In: *Biotechnology and bioengineering* 114.2 (2017), pages 344–354. DOI: 10.1002/bit.26084.

[33] Dres Foged Olsen, John Bagterp Jørgensen, John Villadsen, and Sten Bay Jørgensen. "Modeling and Simulation of Single Cell Protein Production." In: *IFAC Proceedings Volumes* 43.6 (2010), pages 502–507. DOI: 10.3182/20100707-3-BE-2012.0099.

[34] Leander A. H. Petersen, Christian Leiven, Subir K. Nandy, John Villadsen, and Sten Bay Jørgensen. "Dynamics investigation and modeling of the nitrogen cometabolism in Metylococcus capsulatus (Bath)." In: *Boitechnology and bioengineering* 116.11 (2019), pages 2884–2895. DOI: 10.1002/bit.27113.

[35] Leander A. H. Petersen, B. Wayne Bequette, Sten Bay Jørgensen, John Villadsen, Ib Christensen, and Krist V. Gerneay. “Modeling and system identification of an unconventional bioreactor used for single cell protein production.” In: *Chemical engineering journal* 390 (2020), page 124438. DOI: `10.1016/j.cej.2020.124438`.

[36] André Drejer, Tobias K. S. Ritschel, Sten Bay Jørgensen, and John Bagterp Jørgensen. “Economic optimizing control for single-cell protein production in a U-loop reactor.” In: *proceedings of the 27th European Symposium on Computer Aided Process Engineering (ESCAPE).* Barcelona, Spain. October 1-5, 2017, pages 1759–1764. DOI: `10.1016/B978-0-444-63965-3.50295-6`.

[37] Dres Foged Olsen, John Bagterp Jørgensen, John Villadsen, and Sten Bay Jørgensen. “Optimal Operating Points for SCP Production in the U-Loop Reactor.” In: *IFAC Proceedings Volumes* 43.5 (2010), pages 499–504. DOI: `10.3182/20100705-3-BE-2011.00083`.

[38] Unibio A/S. *Single-cell protein production using U-loop fermenter - upstream and downstream.* February 2023. URL: `https://www.unibio.dk/technology/introduction/`.

[39] J.B. Rawlings. “Tutorial overview of model predictive control.” In: *IEEE Control Systems Magazine* 20.3 (2000), pages 38–52. DOI: `10.1109/37.845037`.

[40] James Blake Rawlings, David Q. Mayne, and Moritz Diehl. *Model predictive control: theory, computation, and design.* 2nd edition. Nob Hill Publishing Madison, WI, 2017. ISBN: 9780975937730.

[41] Rudolf Emil Kalman. “A new approach to linear filtering and prediction problems.” In: *Journal of Basic Engineering* 82.1 (1960), pages 35–45. DOI: `10.1115/1.3662552`.

[42] Rudolf Emil Kalman. “Contributions to the theory of optimal control.” In: *Bol. Soc. Mat. Mexicana* 5.2 (1960), pages 102–119. DOI: `10.1109/9780470544334`.

[43] Martin S. Andersen, Joachim Dahl, and Lieven Vandenberghe. *CVXOPT: python software for convex optimisation.* February 2023. URL: `http://cvxopt.org/`.

[44] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. “OSQP: an operator splitting solver for quadratic programs.” In: *Mathematical Programming Computation* 12.4 (2020), pages 637–672. DOI: `10.1007/s12532-020-00179-2`.

[45] Donald Goldfarb and Ashok Idnani. “A numerically stable dual method for solving strictly convex quadratic programs.” In: *Mathematical programming* 27.1 (1983), pages 1–33. DOI: `10.1007/BF02591962`.

[46] Hans Joachim Ferreau, Christian Kirches, Andreas Potschka, Hans Georg Bock, and Moritz Diehl. “qpOASES: A parametric active-set algorithm for quadratic programming.” In: *Mathematical Programming Computation* 6 (2014), pages 327–363. DOI: `10.1007/s12532-014-0071-1`.

[47] Gurobi Optimization, LLC. *Gurobi optimizer reference manual.* 2023. URL: https://www.gurobi.com.

[48] Mosek ApS. *The MOSEK optimization toolbox for Matlab manual. Version 10.0.* 2023. URL: https://www.mosek.com.

[49] Jorge Nocedal and Stephen Wright. *Numerical optimization.* Edited by Thomas V. Mikosch, Sidney I. Resnick, and Stephen M. Robinson. 2nd edition. Springer Series in Operation Research and Financial Engineering. Springer Science+Business Media, 2006. ISBN: 0387303030.

[50] Lorenz T Biegler and Victor M Zavala. "Large-scale nonlinear programming using IPOPT: An integrating framework for enterprise-wide dynamic optimization." In: *Computers & Chemical Engineering* 33.3 (2009), pages 575–582. DOI: 10.1016/j.compchemeng.2008.08.006.

[51] Andrew H. Jazwinski. *Stochastic processes and filtering theory.* Dover Publication, Inc., 2007. 376 pages. ISBN: 0486462749.

[52] Richard S. Bucy and Peter D. Joseph. *Filtering for stochastic processes with applications to guidance.* American Mathematical Society, 2005. ISBN: 0821837826.

[53] Zhe Sage Chen. "Bayesian filtering: from Kalman filters to particle filters, and beyond." In: *Statistics: A Journal of Theoretical and Applied Statistics* 182.1 (2003), pages 1–69. DOI: 10.1080/02331880309257.

[54] Paul Frogerais, Jean-Jacques Bellanger, and Lotfi Senhadji. "Various ways to compute the continuous-discrete extended Kalman filter." In: *IEEE Transactions on Automatic Control* 57.4 (2011), pages 1000–1004. DOI: 10.1109/TAC.2011.2168129.

[55] Simon J. Julier. "The scaled unscented transformation." In: *proceedings of the 2002 American Control Conference (ACC).* Anchorage, A.K., USA. May 8-10, 2002, pages 4555–4559. DOI: 10.1109/ACC.2002.1025369.

[56] Simon J. Julier and Jeffrey K. Uhlmann. "New extension of the Kalman filter to nonlinear systems." In: *proceedings of Signal processing, sensor fusion, and target recognition VI (SPIE).* Orlando, Florida, USA. April 21-25, 1997, pages 182–193. DOI: 10.1117/12.280797.

[57] Simon J. Julier and Jeffrey K. Uhlmann. "Unscented filtering and nonlinear estimation." In: *Proceedings of the IEEE* 92.3 (2004), pages 401–422. DOI: 10.1109/JPROC.2003.823141.

[58] Francesco De Vivo, Alberto Brandl, Manuela Battipede, and Piero Gili. "Joseph covariance formula adaptation to square-root sigma-point Kalman filters." In: *Nonlinear Dynamics* 88.3 (2017), pages 1969–1986. DOI: 10.1007/s11071-017-3356-x.

[59] Eric A. Wan and Rudolph van der Merwe. "The unscented Kalman filter for nonlinear estimation." In: *proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (AS-SPCC)*. Lake Louise, A.B., Canada. October 4, 2000, pages 153–158. DOI: 10.1109/ASSPCC.2000.882463.

[60] Rambabu Kandepu, Bjarne Foss, and Lars Imsland. "Applying the unscented Kalman filter for nonlinear state estimation." In: *Journal of Process Control* 18.7-8 (2008), pages 753–768. DOI: 10.1016/j.jprocont.2007.11.004.

[61] Renato Zanetti and Kyle J. DeMars. "Joseph formulation of unscented and quadrature filters with application to consider states." In: *Journal of Guidance, Control, and Dynamics* 36.6 (2013), pages 1860–1864. DOI: 10.2514/1.59935.

[62] Steven Gillijns, O. Barrero Mendoza, Jaganath Chandrasekar, B. L. R. De Moor, D. S. Bernstein, and A. Ridley. "What is the ensemble Kalman filter and how well does it work?" In: *proceedings of the 2006 American Control Conference (ACC)*. Minneapolis, M.N., USA. June 14-16, 2006, pages 4448–4453. DOI: 10.1109/ACC.2006.1657419.

[63] Geir Evensen. "Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics." In: *Journal of Geophysical Research: Oceans* 99.C5 (1994), pages 10143–10162. DOI: 10.1029/94JC00572.

[64] Michael Roth, Carsten Fritsche, Gustaf Hendeby, and Fredrik Gustafison. "The ensemble Kalman filter and its relations to other nonlinear filters." In: *proceedings of the 23rd European Signal Processing Conference (EUSIPCO)*. Nice, France. August 31-September 4, 2015, pages 1236–1240. DOI: 10.1109/EUSIPCO.2015.7362581.

[65] Inge Myrseth and Henning Omre. "Hierarchical ensemble Kalman filter." In: *SPE Journal* 15.2 (2010), pages 569–580. DOI: 10.2118/125851-PA.

[66] Eric L. Haseltine and James B. Rawlings. "Critical evaluation of extended Kalman filtering and moving-horizon estimation." In: *Industrial & Engineering Chemistry Research* 44.8 (2005), pages 2451–2460. DOI: 10.1021/ie034308l.

[67] Aditya Tulsyan, Rohit B. Gopaluni, and Swanand R. Khare. "Particle filtering without tears: a primer for beginners." In: *Computers & Chemical Engineering* 95 (2016), pages 130–145. DOI: 10.1016/j.compchemeng.2016.08.015.

[68] Arnaud Doucet and Adam Michael Johansen. *A tutorial on particle filtering and smoothing: Fifteen years later*. 2009.

[69] Arjun V. Shenoy, Jagadeesan Prakash, K. B. McAuley, Vinay Prasad, and Sirish L. Shah. "Practical issues in the application of the particle filter for estimation of chemical processes." In: *proceedings of the 18th IFAC World Congress*. Italy, Milano. August 28-September 2, 2011, pages 2773–2778. DOI: 10.3182/20110828-6-IT-1002.03272.

[70] John Bagterp Jørgensen. "A critical discussion of the continuous-discrete extended Kalman filter." In: *proceedings of the 6th European Congress of Chemical Engineering (ECCE).* Copenhagen, Denmark. September 16-20, 2007.

[71] Rudolph van der Merwe, Arnaud Doucet, Nando De Freitas, and Eric A. Wan. "The unscented particle filter." In: *proceedings of the 13th Advances in Neural Information Processing Systems (NIPS).* Vancouver, B.C., Canada. December 3-8, 2001, pages 584–590.

[72] Matthias Katzfuss, Jonathan R Stroud, and Christopher K Wikle. "Understanding the ensemble Kalman filter." In: *The American Statistician* 70.4 (2016), pages 350–357. DOI: 10.1080/00031305.2016.1141709.

[73] M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking." In: *IEEE Transactions on Signal Processing* 50.2 (2002), pages 174–188. DOI: 10.1109/78.978374.

[74] Tobias Kasper Skovborg Ritschel and John Bagterp Jørgensen. "Nonlinear filters for state estimation of UV flash processes." In: *proceedings of the 2018 IEEE Conference on Control Technology and Applications (CCTA).* Copenhagen, Denmark. August 21-24, 2018, pages 1753–1760. DOI: 10.1109/CCTA.2018.8511532.

[75] Lorenz T. Biegler. *Nonlinear programming: concepts, algorithms, and applications to chemical processes.* SIAM, 2010. ISBN: 9780898717020.

[76] Robert J. Davis Mark E. Davis. *Fundamentals of chemical reaction engineering.* Courier Corporation, 2012. ISBN: 9781628704372.

[77] Robin Smith. *Chemical process design and intergration.* John Wiley & Sons, Ltd, 2005. ISBN: 0471486817.

[78] H. Scott Folger. *Elements of chemical engineering.* 4th edition. Pearson education limited, 2014. ISBN: 9781292026169.

[79] Peter Victor Danckwerts. "Continuous flow systems: distribution of residence times." In: *Chemical engineering science* 2.1 (1953), pages 1–13. DOI: 10.1016/0009-2509(53)80001-1.

[80] Rolf Sander. *Henry's Law Constants.* January 2023. URL: http://www.henrys-law.org/henry/.

[81] Rolf Sander. "Compilation of Henry's law constants (version 4.0) for water as solvent." In: *Atmospheric Chemistry and Physics* 15.8 (2015), pages 4399–4981. DOI: 10.5194/acp-15-4399-2015.

[82] Lars Norbert Petersen and John Bagterp Jørgensen. "Real-time economic optimization for a fermentation process using model predictive control." In: *2014 European Control Conference (ECC).* Strasbourg, France. June 24-27, 2014, pages 1831–1836. DOI: 10.1109/ECC.2014.6862270.

[83] Thomas Emil Ryde, Morten Ryberg Wahlgreen, Marcus Krogh Nielsen, Steen Hørsholt, Sten Bay Jørgensen, and John Bagterp Jørgensen. “Optimal feed trajectories for fedbatch fermentation with substrate inhibition kinetics.” In: *proceedings of the International Symposium on Advanced Control of Chemical Processes (ADCHEM)*. Venice, Italy. June 13-16, 2021, pages 318–323. DOI: `10.1016/j.ifacol.2021.08.261`.

[84] Eskild Shcroll-Fleischer and Thomas Geert Reichenbach Nielsen. ”Implementation of optimizing control for a single-cell protein plant.” MSc thesis. Technical University of Denmark, 2017.

[85] J. Bevan Ott and Juliana Boerio-Gates. *Chemical Thermodynamics Principles and Applications*. Academic Press, 2000. 664 pages. ISBN: 9781281038463.

[86] William M Haynes. *CRC handbook of chemistry and physics*. CRC press, 2016. 2670 pages. ISBN: 9781315380476.

[87] William M Haynes. *Libre text chemistry*. February 2023. URL: `https://chem.libretexts.org`.

[88] Desmond J. Higham. “An algorithmic introduction to numerical simulation of stochastic differential equations.” In: *SIAM Review* 43.3 (2001), pages 525–546. DOI: `10.1137/S0036144500378302`.

[89] Timothy Sauer. “Numerical solution of stochastic differential equations in finance.” In: *Handbook of computational finance*. Springer, 2011, pages 529–550. ISBN: 978-3-642-17253-3.

[90] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004. 732 pages. ISBN: 0521833787.

[91] Sanjay Mehrotra. “On the Implementation of a Primal-Dual Interior Point Method.” In: *SIAM Journal on Optimization* 2.4 (1992), pages 575–601. DOI: `10.1137/0802028`.

[92] Adi Ben-Israel. “A Newton-Raphson method for the solution of systems of equations.” In: *Journal of Mathematical analysis and applications* 15.2 (1966), pages 243–252. DOI: `10.1016/0022-247x(66)90115-6`.

[93] Keith Meintjes and Alexander P Morgan. “A methodology for solving chemical equilibrium systems.” In: *Applied Mathematics and Computation* 22.4 (1987), pages 333–361. DOI: `10.1016/0096-3003(87)90076-2`.

[94] Tobias Kasper Skovborg. Ritschel, Dimitri Boiroux, Marcus Krogh Nielsen, Jakob Kjøbsted Huusom, Sten Bay Jørgensen, and John Bagterp Jørgensen. “The extended Kalman filter for nonlinear state estimation in a U-loop bioreactor.” In: *proceedings of the 2019 IEEE Conference on Control Technology and Applications (CCTA)*. Hong Kong, China. August 19-21, 2019, pages 920–925. DOI: `10.1109/CCTA.2019.8920643`.

[95] Marco Bonvini, Michael D Sohn, Jessica Granderson, Michael Wetter, and Mary Ann Piette. “Robust on-line fault detection diagnosis for HVAC components based on nonlinear state estimation techniques.” In: *Applied Energy* 124 (2014), pages 156–166. DOI: 10.1016/j.apenergy.2014.03.009.

[96] F.N. Chowdhury, J.P. Christensen, and J.L. Aravena. “Power system fault detection and state estimation using Kalman filter with hypothesis testing.” In: *IEEE Transactions on Power Delivery* 6.3 (1991), pages 1025–1030. DOI: 10.1109/61.85843.

[97] Shen Yin and Xiangping Zhu. “Intelligent particle filter and its application to fault detection of nonlinear system.” In: *IEEE Transactions on Industrial Electronics* 62.6 (2015), pages 3852–3861. DOI: 10.1109/TIE.2015.2399396.

[98] Tobias K. S. Ritschel, Dimitri Boiroux, Marcus Krogh Nielsen, Jakob Kjøbsted Huusom, Sten Bay Jørgensen, and John Bagterp Jørgensen. “Economic optimal control of a U-loop bioreactor using simultaneous collocation-based approaches.” In: *proceedings of the 2019 IEEE Conference on Control Technology and Applications (CCTA)*. Hong Kong, China. August 19-21, 2019, pages 933–938. DOI: 10.1109/CCTA.2019.8920479.

[99] Tobias K. S. Ritschel, Dimitri Boiroux, Marcus Krogh Nielsen, Jakob Kjøbsted Huusom, Sten Bay Jørgensen, and John Bagterp Jørgensen. “Economic nonlinear model predictive control of a U-loop bioreactor.” In: *proceedings of the 2020 European Control Conference (ECC)*. Saint Petersburg, Russia. May 12-15, 2020, pages 208–213. DOI: 10.23919/ECC51009.2020.9143629.

[100] James B. Rawlings and Bhavik R. Bakshi. “Particle filtering and moving horizon estimation.” In: *Computers & Chemical Engineering* 30.10-12 (2006), pages 1529–1541. DOI: 10.1016/j.compchemeng.2006.05.031.

[101] Reé Schneider and Christos Georgakis. “How to not make the extended Kalman filter fail.” In: *Industrial & Engineering Chemistry Research* 52.9 (2013), pages 3354–3362. DOI: 10.1021/ie300415d.

[102] John Bagterp. Jørgensen, Morten Rode. Kristensen, Per Grove. Thomsen, and Henrik. Madsen. “New extended Kalman filter algorithms for stochastic differential algebraic equations.” In: *Assessment and Future Directions of Nonlinear Model Predictive Control*. Edited by Rolf. Findeisen, Frank. Allgöwer, and Lorenz T. Biegler. Springer Berlin Heidelberg, 2007, pages 359–366. ISBN: 9783540726999. DOI: 10.1007/978-3-540-72699-9_29.

[103] Torben Knudsen and John Leth. “A New Continuous Discrete Unscented Kalman Filter.” In: *IEEE Transactions on Automatic Control* 64.5 (2019), pages 2198–2205. DOI: 10.1109/TAC.2018.2867325.

[104] R. Van der Merwe and E.A. Wan. “The square-root unscented Kalman filter for state and parameter-estimation.” In: *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*. Volume 6. 2001, pages 3461–3464. DOI: 10.1109/ICASSP.2001.940586.

[105] Rishi Amrit, James B. Rawlings, and Lorenz T. Biegler. “Optimizing process economics online using model predictive control.” In: *Computers and Chemical Engineering* 58 (2013), pages 334–343. DOI: `10.1016/j.compchemeng.2013.07.015`.

[106] Asbjørn Thode Reenberg, Dimitri Boiroux, Tobias Kasper Skovborg Ritschel, and John Bagterp Jørgensen. “Model Predictive Control of the Blood Glucose Concentration for Critically Ill Patients in Intensive Care Units.” In: *2019 IEEE 58th Conference on Decision and Control (CDC)*. 2019, pages 3762–3769. DOI: `10.1109/CDC40024.2019.9029651`.

[107] Asbjørn Thode Reenberg, Tobias K.S. Ritschel, Emilie B. Lindkvist, Christian Laugesen, Jannet Svensson, Ajenthen G. Ranjan, Kirsten Nørgaard, and John Bagterp Jørgensen. “Nonlinear Model Predictive Control and System Identification for a Dual-hormone Artificial Pancreas.” In: *IFAC-PapersOnLine* 55.7 (2022), pages 915–921. DOI: `https://doi.org/10.1016/j.ifacol.2022.07.561`.

[108] Joel AE Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. “CasADi: a software framework for nonlinear optimization and optimal control.” In: *Mathematical Programming Computation* 11 (2019), pages 1–36. DOI: `10.1007/s12532-018-0139-4`.

[109] Asbjørn Thode Reenberg, Tobias KS Ritschel, Bernd Dammann, and John Bagterp Jørgensen. “High-performance Uncertainty Quantification in Large-scale Virtual Clinical Trials of Closed-loop Diabetes Treatment.” In: *2022 American Control Conference (ACC)*. IEEE. 2022, pages 1367–1372. DOI: `10.23919/ACC53348.2022.9867234`.

[110] Tobias KS Ritschel, Asbjørn Thode Reenberg, and John Bagterp Jørgensen. “Large-scale Virtual Clinical Trials of Closed-loop Treatments for People with Type 1 Diabetes.” In: *arXiv preprint arXiv:2205.01332* (2022). DOI: `10.48550/arXiv.2205.01332`.

[111] Morten Wahlgreen Kaysfeld, Mario Zanon, and John Bagterp Jørgensen. “Performance Quantification of a Nonlinear Model Predictive Controller by Parallel Monte Carlo Simulations of a Closed-loop System.” In: *arXiv preprint arXiv:2212.02197* (2022).

[112] Morten Ryberg Wahlgreen and John Bagterp Jørgensen. “On the Implementation of a Preconditioned Riccati Recursion based Primal-Dual Interior-Point Algorithm for Input Constrained Optimal Control Problems .” In: *IFAC-PapersOnLine* 55.7 (2022), pages 346–351. DOI: `10.1016/j.ifacol.2022.07.468`.

[113] Morten Ryberg Wahlgreen, John Bagterp Jørgensen, and Mario Zanon. “Model Predictive Control Tuning by Monte Carlo Simulation and Controller Matching.” In: *arXiv preprint arXiv:2212.02142* (2022).

Part VII

Appendix

APPENDIX A
Paper I - ECC 2023

Modelling and economic optimal control for a laboratory-scale continuous stirred tank reactor for single-cell protein production

Authors:
Marcus Krogh Nielsen, Jens Dynesen, Jess Dragheim, Ib Christensen, Sten Bay Jørgensen, Jakob Kjøbsted Huusom, Krist V. Gernaey, John Bagterp Jørgensen

Accepted for:
2023 21st IEEE european control conference (ECC). June 13-16, 2023. Bucharest, Romania.

Modelling and Economic Optimal Control for a Laboratory-scale Continuous Stirred Tank Reactor for Single-cell Protein Production

Marcus Krogh Nielsen, Jens Dynesen, Jess Dragheim, Ib Christensen, Sten Bay Jørgensen, Jakob Kjøbsted Huusom, Krist V. Gernaey, John Bagterp Jørgensen

***Abstract*— In this paper, we present a novel kinetic growth model for the micro-organism *Methylococcus capsulatus* (Bath) that couples growth and pH. We apply growth kinetics in a model for single-cell protein production in a laboratory-scale continuous stirred tank reactor inspired by a physical laboratory fermentor. The model contains a set of differential algebraic equations describing growth and pH-dynamics in the system. We present a method of simulation that ensures non-negativity in the state and algebraic variables. Additionally, we introduce linear scaling of the algebraic equations and variables for numerical stability in Newton's method. Finally, we conduct a numerical experiment of economic optimal control for single-cell protein production in the laboratory-scale reactor. The numerical experiment shows non-trivial input profiles for biomass growth and pH tracking.**

I. Introduction

Single-cell protein (SCP) provides an alternative source of protein for the feed and food industries to meet the growing demand for protein in the coming decades [1]. Methanotrophs are bacteria capable of metabolising methane as their source of carbon. Methane is a cheap source of carbon. *Methylococcus capsulatus* (Bath) are methanotrophic bacteria with high protein content that is well-suited for production of SCP [2].

M. capsulatus is aerobic and growth therefore involves fixation of methane and oxygen gas. A U-loop bioreactor has been developed for SCP production. It has been demonstrated that the U-loop bioreactor have good mixing and mass transfer properties [3]. Early work on kinetic modelling for growth of *M. capsulatus* shows process instability for high biomass concentrations [4]–[6]. Advanced process control techniques have been applied in numerical experiments for monitoring [7] and to increase stability and productivity in both open- and closed-loop [8], [9]. The metabolism of *M. capsulatus* is well-described in the literature and a metabolic network model exists [10]. Metabolic knowledge and process data show that the micro-organism can metabolise several nitrogen sources: ammonium, nitrate, nitrite, and molecular nitrogen. A kinetic growth model exists for growth on these

[*]This work is funded by Innovation Fund Denmark (9065-00269B).
Marcus Krogh Nielsen and John Bagterp Jørgensen are with the Department of Applied Mathematics and Computer Science and Sten Bay Jørgensen, Jakob Kjøbsted Huusom, and Krist V. Gernaey are with the Department of Chemical and Biochemical Engineering at the Technical University of Denmark, Kgs. Lyngby, Denmark. Jens Dynesen, Jess Dragheim, Ib Christensen, and Marcus Krogh Nielsen are with Unibio A/S, Roskilde, Denmark. email: {mkrni,jbjo}@dtu.dk

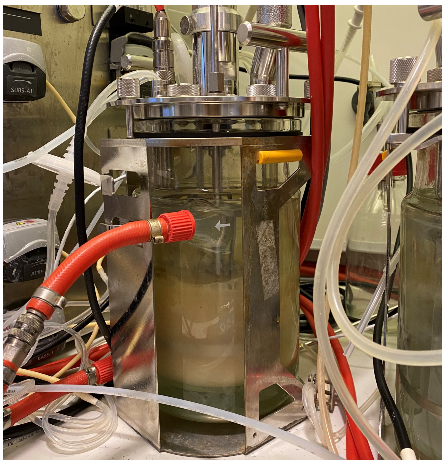


Fig. 1. Laboratory-scale reactor for fermention of *M. capsulatus*.

nitrogen sources with methane as carbon source [11]. Kinetic parameters for the model are estimated from experimental data [12]. The micro-organism metabolises ammonium rather than ammonia directly, pH-dynamics are therefore highly relevant to model precise concentrations of growth-related chemical components. A model including equilibrium reaction in the growth of *M. capsulatus* exists, but the kinetics do not depend explicitly on the algebraic variables [13]. To the best of our knowledge, models coupling growth kinetics and pH-dynamics for *M. capsulatus* as presented in this work have not yet been introduced in the literature.

In this paper, we present a novel growth model for SCP production by cultivation of the micro-organism *M. capsulatus*. In the model, methane is the carbon source and ammonium is the sole nitrogen source, i.e. growth on nitrite, nitrate, and molecular nitrogen is not included in the model. As such, we apply chemical equilibrium reactions to describe relevant pH-dynamics. We model the fermentation process in a laboratory-scale continuous stirred tank reactor (CSTR), based on a physical laboratory-scale fermentor. Fig 1 shows the laboratory reactor. The resulting set of differential algebraic equations (DAEs) model reactor dynamics, growth kinetics, and pH equilibrium dynamics. Optimal control, optimal design of experiments, and parameter estimation motivate the development of the model presented in this work.

The paper is organised as follows. Section II describes modelling of growth kinetics, pH equilibrium dynamics, as well as reactor dynamics for a CSTR. Section III presents methods for numerical solution of DAEs with non-negativity constraints as well as scaling of algebraic equations and variables for numerical stability. Section IV describes an economic optimal control problem (OCP) for SCP production in a CSTR. Section V presents a numerical experiment of optimal biomass production in a laboratory-scale CSTR. Section VI presents conclusions.

II. Modelling

In this section, we describe a mathematical model for SCP production in a CSTR. The model is a system of DAEs in the form

$$\frac{dx}{dt}(t) = f(x(t), y(t), u(t), \theta), \qquad x(t_0) = x_0, \tag{1a}$$
$$0 = g(x(t), y(t), \theta), \tag{1b}$$

where $f(\cdot)$ are state dynamical equations, $g(\cdot)$ are algebraic expressions, t is time, $x(t) \in \mathbb{R}^{n_x}$ are states, $y(t) \in \mathbb{R}^{n_y}$ are algebraic variables, $u(t) \in \mathbb{R}^{n_u}$ are manipulated variables, and θ are system parameters.

A. Variable definitions

We define the sets

$$\mathcal{C}_{x,l} = \{X, N, NO, Na\}, \quad \mathcal{C}_{x,g} = \{S, O, C\}, \tag{2a}$$
$$\mathcal{C}_x = \mathcal{C}_{x,l} \cup \mathcal{C}_{x,g}, \tag{2b}$$
$$\mathcal{C}_y = \{\mathrm{H_3O^+}, \mathrm{OH^-}, \mathrm{NH_3}, \mathrm{NH_4}^+, \mathrm{CO_2}, \mathrm{H_2CO_3}, \mathrm{HCO_3}^-, \mathrm{CO_3}^{2-}\}, \tag{2c}$$
$$\mathcal{C}_{f,l} = \{W, N, NO, Na\}, \quad \mathcal{C}_{f,g} = \{S, O\}, \tag{2d}$$
$$\mathcal{C}_f = \mathcal{C}_{f,l} \cup \mathcal{C}_{f,g}. \tag{2e}$$

$\mathcal{C}_{x,l}$ is the set of dissolved state components, $\mathcal{C}_{x,g}$ is the set of state components present as both in gas phase and dissolved in water, $\mathcal{C}_x$ is the set of all state components, such that the components in the state vector, c_x, are the concentrations of the state components, $c_{x,i} = [i]$ for $i \in \mathcal{C}_x$. $\mathcal{C}_y$ is the set of algebraic components, such that the components in the algebraic variable vector, c_y, are the concentrations $c_{y,i}$ for $i \in \mathcal{C}_y$. $\mathcal{C}_{f,l}$ is the set of liquid inlet flows, $\mathcal{C}_{f,g}$ is the set of gaseous inlet flows, and $\mathcal{C}_f$ is the set of all inlet flows, such that the components in the inlet flow vector, F, are F_i for $i \in \mathcal{C}_f$. The state components are

$$X = \mathrm{CH_{1.8}O_{0.5}N_{0.2}}, \quad S = \mathrm{CH_4}, \quad O = \mathrm{O_2}, \tag{3a}$$
$$N = \{\mathrm{NH_3}, \mathrm{NH_4}^+\}, \quad NO = \mathrm{NO_3}^-, \quad Na = \mathrm{Na^+}, \tag{3b}$$
$$C = \{\mathrm{CO_2}, \mathrm{H_2CO_3}, \mathrm{HCO_3}^-, \mathrm{CO_3}^{2-}\}, \tag{3c}$$
$$S_g = \mathrm{CH_4(g)}, \quad O_g = \mathrm{O_2(g)}, \quad C_g = \mathrm{CO_2(g)}. \tag{3d}$$

For the inlet components, W is water.

B. Stoichiometry and kinetics

We describe a catabolic reaction and the anabolism of ammonium (not including ATP and ADP) in *M. capsulatus* in terms of the reactions

$$\mathrm{CH_4 + O_2 + H_2O \longrightarrow CO_2 + 4\,H_3O^+}, \qquad r_1, \tag{4a}$$
$$\mathrm{CH_4 + O_2} + \frac{2}{10}\mathrm{NH_4}^+ \longrightarrow \mathrm{X} + \frac{15}{10}\mathrm{H_2O}, \qquad r_2, \tag{4b}$$

where $X = \mathrm{CH_{1.8}O_{0.5}N_{0.2}}$ is the biomass and $r_i(c)$ for $i \in \{1, 2\}$ are the reaction rates. The modelled stoichiometry is

$$\mathrm{S + O \longrightarrow C}, \qquad r_1(c), \tag{5a}$$
$$\mathrm{S + O + 0.2\,N \longrightarrow X}, \qquad r_2(c). \tag{5b}$$

The reactions in (5) correspond to the stoichiometric matrix

$$\nu = \begin{bmatrix} 0 & -1 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & -1 & -0.2 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{6}$$

The reaction rates are

$$r_i(c) = \mu_i(c) c_X, \qquad i \in \{1, 2\}, \tag{7}$$

where c_X is the biomass concentration. The specific growth rates are

$$\mu_1(c) = \left(\frac{\alpha}{2\delta} + \frac{8}{20}\right)\mu_2(c) + \frac{m}{2\delta}, \tag{8a}$$
$$\mu_2(c) = \mu_{\max}\mu_S(c)\mu_O(c)\mu_N(c), \tag{8b}$$

where $\mu_{\max}$ is the maximum growth rate, α and δ are parameters related to ATP production and consumption, and m is a maintenance constant. The growth dependence for methane, oxygen, and ammonium are

$$\mu_S(c) = \frac{c_S}{K_S\left(1 + \frac{c_{NH_4^+}}{K_{N,ox}}\right) + c_S}, \tag{9a}$$
$$\mu_O(c) = \frac{c_O}{K_O + c_O}, \tag{9b}$$
$$\mu_N(c) = \frac{c_{NH_4^+}}{K_N + c_{NH_4^+}}, \tag{9c}$$

The concentrations of dissolved methane, oxygen, and ammonium are denoted c_i for $i \in \{S, O, NH_4^+\}$, respectively. K_i for $i \in \{S, O, N\}$ are kinetic constants and $K_{N,ox}$ is the ammonium inhibition constant. We note here that the specific growth for methane, $\mu_S(\cdot)$, directly depends on the concentration of an algebraic component, $c_{NH_4^+}$.

C. pH-balance

A set of chemical equilibrium reactions, stoichiometric reactions, and mass and charge balances describe the pH equilibrium dynamics in the system.

Weak acids and bases: The reactions describing weak acid and base equilibrium are

$$2\,\mathrm{H_2O(aq)} \rightleftharpoons \mathrm{H_3O^+(aq) + OH^-(aq)}, \qquad K_{e,W}, \tag{10a}$$
$$\mathrm{NH_3(aq) + H_2O(aq)} \rightleftharpoons \mathrm{NH_4^+(aq) + OH^-(aq)}, \qquad K_{e,N}, \tag{10b}$$
$$\mathrm{CO_2(aq) + H_2O(aq)} \rightleftharpoons \mathrm{H_2CO_3(aq)}, \qquad K_{e,C1}, \tag{10c}$$

$$H_2CO_3(aq) + H_2O(aq) \rightleftharpoons HCO_3^-(aq) + H_3O^+(aq), \qquad K_{e,C2}, \tag{10d}$$

$$HCO_3^-(aq) + H_2O(aq) \rightleftharpoons CO_3^{2-}(aq) + H_3O^+(aq), \qquad K_{e,C3}. \tag{10e}$$

The equilibrium constants for water, $K_{e,W}$, ammonia, $K_{e,N}$, and carbon dioxide, $K_{e,C1}$, $K_{e,C2}$, and $K_{e,C3}$, govern the equilibrium reactions.

Strong acids and bases: The reactions describing strong acid and base dynamics are

$$NaOH(aq) \longrightarrow Na^+(aq) + OH^-(aq), \tag{11a}$$

$$HNO_3(aq) + H_2O(aq) \longrightarrow NO_3^-(aq) + H_3O^+(aq). \tag{11b}$$

The reactions are assumed to be instantaneous and complete.

System of algebraic equations: The chemical equilibrium reactions give rise to the algebraic equations

$$0 = [H_3O^+][OH^-] - K_{e,W}, \tag{12a}$$

$$0 = [NH_4^+][OH^-] - K_{e,N}[NH_3], \tag{12b}$$

$$0 = [H_2CO_3] - K_{e,C1}[CO_2], \tag{12c}$$

$$0 = [HCO_3^-][H_3O^+] - K_{e,C2}[H_2CO_3], \tag{12d}$$

$$0 = [CO_3^{2-}][H_3O^+] - K_{e,C3}[HCO_3^-]. \tag{12e}$$

The mass balance equations are

$$0 = [NH_3] + [NH_4^+] - c_N, \tag{13a}$$

$$\begin{aligned} 0 &= [CO_2] + [H_2CO_3] + [HCO_3^-] \\ &\quad + [CO_3^{2-}] + [CO_2(g)] - c_C. \end{aligned} \tag{13b}$$

The charge balance equations are

$$\begin{aligned} 0 &= [OH^-] + [HCO_3^-] - 2[CO_3^{2-}] \\ &\quad - c_{NO} - [H_3O^+] - [NH_4^+] - c_{Na}. \end{aligned} \tag{14}$$

We denote the resulting algebraic system of 8 equations and 8 variables

$$0 = g(x(t), y(t), \theta). \tag{15}$$

D. Continuous stirred tank reactor

We model the laboratory-scale CSTR in the general form

$$\frac{dc_x}{dt} = \frac{1}{V}\left(C_{In} - c_x e_l^T - c_x e_g^T\right) F + Q(c), \tag{16}$$

where $c_x \in \mathbb{R}^{n_x}$ are concentration of the state components, i.e. c_i for $i \in \mathcal{C}_x$, V is the constant reactor volume, $C_{In} \in \mathbb{R}^{n_x \times n_u}$ are inlet concentrations of each inlet flow, $F \in \mathbb{R}^{n_u}$ are inlet flows, and c are concentrations of all state and algebraic components in the system. The inlet streams $F = [F_l; F_g]$, where F_l and F_g are the liquid and gas inlet streams, respectively. The vectors $e_l \in \{0,1\}^{n_u}$ and $e_g \in \{0,1\}^{n_u}$ are indicator vectors for liquid and gaseous inflows respectively, i.e. e_l is 1 for all liquid flows and 0 for all gaseous flows and e_g is 0 for all liquid flows and 1 for all gaseous flows. The production and mass transfer is defined for components in aqueous solution for $i \in \mathcal{C}_{x,l}$ as

$$Q_i(c) = R_i(c), \tag{17}$$

TABLE I

MODEL PARAMETERS FOR SINGLE-CELL PROTEIN PRODUCTION IN CONTINUOUS STIRRED TANK REACTOR [12].

name	**value**	**unit**	**name**	**value**	**unit**
$\mu_{\max}$	2.28e−1	1/h	m	9.80e−5	1/h
α	2.00e−2	-	δ	2.00e−2	-
K_S	7.50e−5	mol/L	$K_{N,ox}$	3.30e−3	mol/L
K_O	5.50e−5	mol/L	K_N	1.30e−3	mol/L
$K_{e,W}$	1.00e−14	-	$K_{e,N}$	5.62e−10	-
$K_{e,C1}$	1.58e−7	-	$K_{e,C2}$	4.27e−7	-
$K_{e,C3}$	4.79e−11	-	$c_{In,N}$	5.88	mol/L
$c_{In,Na}$	1.00	mol/L	$c_{In,NO}$	1.00	mol/L
$c_{S,g}$	1.90e−1	mol/L	$c_{O,g}$	1.90e−1	mol/L
k_La_S	3.89e+2	1/h	H_S^{pc}	7.05e+2	atm/M
k_La_O	3.71e+2	1/h	H_O^{pc}	7.59e+2	atm/M
k_La_C	3.26e+2	1/h	H_C^{pc}	2.99e+1	atm/M
R	8.21e−2	atm/(K M)	T	3.15e+2	K
V	1.00	L	-	-	-

and for $i \in \mathcal{C}_{x,g}$ as

$$Q_i(c) = R_i(c) + \frac{1}{1-\epsilon} J_{gl,i}(c), \tag{18}$$

and for components in gas-phase for $i \in \mathcal{C}_{x,g}$ as

$$Q_i(c) = -\frac{1}{\epsilon} J_{gl,i}(c). \tag{19}$$

The production rate is

$$R(c) = \nu^T r(c), \tag{20}$$

where $\nu \in \mathbb{R}^{n_r \times n_x}$ is the stoichiometric matrix (6). The gas-liquid mass transfers for $i \in \mathcal{C}_{x,g}$ are defined as

$$J_{gl,i}(c) = k_La_i\left(c_{Sat,i} - c_i\right), \qquad c_{Sat,i} = \gamma_i c_{i,g}, \tag{21}$$

The gas-liquid volume fraction is $\epsilon = F_g/(F_l + F_g)$, k_La_i are mass transfer coefficients, $c_{Sat,i}$ are saturation concentrations, and γ_i gas-liquid ratio. We apply Henry's law to compute the gas-liquid ratio

$$\gamma_i = \frac{RT}{H_i^{pc}}, \tag{22}$$

Henry's constants for chemicals components dissolved in aqueous solution are described in [14]. Table I describes the model parameters.

III. SIMULATION

In this section, we describe Euler's implicit method for numerical solution of DAEs. Additionally, we describe a variation Newton's method, which yields only non-negative solutions.

A. Euler's implicit method

For differential algebraic systems in the form presented in (1), we may discretise with Euler's implicit method as

$$x_{k+1} = x_k + f(x_{k+1}, y_{k+1}, u_k, \theta)\Delta t, \tag{23a}$$

$$0 = g(x_{k+1}, y_{k+1}, \theta). \tag{23b}$$

From this, we define the residual function

$$\begin{aligned} D(z_{k+1}) &= D(x_{k+1}, y_{k+1}) \\ &= \begin{bmatrix} x_{k+1} - x_k - f_k(x_{k+1}, y_{k+1})\Delta t \\ g_k(x_{k+1}, y_{k+1}) \end{bmatrix}, \end{aligned} \tag{24}$$

where $z_k = [x_k; y_k]$, $f_k(x,y) = f(x,y,u_k,\theta)$ and $g_k(x,y) = g(x,y,\theta)$. Solutions to the difference equation, (23), are roots in the residual equation, (24), obtained by solving

$$0 = D(z_{k+1}). \tag{25}$$

For chemical systems modelling concentrations, only non-negative solutions represent solutions in the physical system.

B. Newton's method with asbolute step

Newton's method provides a means of iteratively finding roots given a function and its Jacobian. For a residual function, $D(z)$, the system of equations

$$0 = D(z), \tag{26}$$

can be solved iteratively, where the search direction, Δz, is obtained as the solution of

$$0 = \frac{\partial D}{\partial z}(z)\Delta z + D(z). \tag{27}$$

The Newton step in each iteration, n, is

$$z_{n+1} = z_n + \Delta z_n, \quad \Delta z_n = -\left(\frac{\partial D}{\partial z}(z_n)\right)^{-1} D(z_n). \tag{28}$$

Implicit Euler discretisation of DAE systems in the form (1) results in the residual Jacobian,

$$\frac{\partial D}{\partial z}(z) = \begin{bmatrix} I - \frac{\partial f_k}{\partial x}(x,y)\Delta t & -\frac{\partial f_k}{\partial y}(x,y)\Delta t \\ \frac{\partial g_k}{\partial x}(x,y) & \frac{\partial g_k}{\partial y}(x,y) \end{bmatrix}. \tag{29}$$

In the case where only non-negative solutions to the residual also represent solutions to the physical system, we may apply so-called absolute Newton's method, such that the step becomes

$$z_{n+1} = |z_n + \Delta z_n|. \tag{30}$$

This variation of Newton's method ensures non-negativity in the roots, but does not maintain the convergence properties of the original formulation of the method. The method is empirical, but has been successfully applied to chemical equilibrium systems [15].

C. Scaling algebraic equations and variables

The different scales of the algebraic equations and variables may result in ill-conditioning in the Jacobian of the residual (29). This can lead to numerical inaccuracies and divergence in Newton's method. As such, we introduce a scaling of the algebraic variables and equations

$$\tilde{g}(x(t), \tilde{y}(t), \theta) = S_g g(x(t), S_y \tilde{y}(t), \theta), \tag{31}$$

$$y(t) = S_y \tilde{y}(t). \tag{32}$$

The matrices, $S_g = \text{diag}(s_g)$ and $S_y = \text{diag}(s_y)$, are diagonal scaling matrices. The vectors, $s_g \in \mathbb{R}^{n_y}$ and $s_y \in \mathbb{R}^{n_y}$, define the scaling factors for the algebraic equations and variables, respectively. The resulting scaled Jacobian is

$$\frac{\partial \tilde{g}}{\partial \tilde{y}} = \frac{\partial \tilde{g}}{\partial g}\frac{\partial g}{\partial y}\frac{\partial y}{\partial \tilde{y}} = S_g \frac{\partial g}{\partial y} S_y. \tag{33}$$

We choose S_g and S_y such that the conditioning of the Jacobian (33) is improved.

IV. Economic optimal control problem

In this section, we present a formulation of an economic OCP for SCP production in a CSTR. The presented system is unstable, i.e. simulations with constants or poorly chosen input profiles are likely to lead to washout (zero biomass concentration). As such, we simulate the system using an optimal economic open-loop profile, i.e. solution to an economic OCP. Specifically, we modify and solve the OCP defined for SCP production in a U-loop bioreactor presented by Ritschel et al. [8], [9].

A. Formulation

We consider the economic objective of the OCP

$$\phi = \alpha_{eco}\phi_{eco} + \alpha_{pH}\phi_{pH} + \alpha_{du}\phi_{du}, \tag{34}$$

where

$$\phi_{eco} = \phi_{cost} - \phi_{profit} - \phi_{ctg}. \tag{35}$$

The economic objective terms are

$$\phi_{profit} = \int_{t_0}^{T} p_X c_X(t) e^T F_l(t) dt, \tag{36a}$$

$$\phi_{cost} = \int_{t_0}^{T} p_F^T F(t) dt, \tag{36b}$$

$$\phi_{ctg} = p_X V \left(c_X(T) - c_X(t_0)\right). \tag{36c}$$

The objective term ϕ_{profit} is the value of the harvested biomass over the horizon, ϕ_{cost} is the cost associated with the inlet streams of gaseous and liquid substrates, and ϕ_{ctg} is the cost-to-go. The cost-to-go term is included such that the reactor is not emptied for additional profit at the end of the control horizon. p_X [USD/g] and p_F [USD/L] are the unit values of the biomass and inlets, respectively. $F_l(t)$ [L/h] are the liquid inlet streams, $e^T F_l(t)$ [L/h] is the liquid outlet stream, and V [L] is the reactor volume. We define the pH tracking objective term, as

$$\phi_{pH} = \frac{1}{2}\int_{t_0}^{T} \left\| \overline{pH} - pH(t) \right\|_{Q_{pH}}^2 dt, \tag{37}$$

where $\overline{pH}$ is the target, $pH(t) = -\log_{10}\left([\text{H}_3\text{O}^+]\right)$, and Q_{pH} is a weight matrix. We define the input rate-of-movement objective as

$$\phi_{du} = \frac{1}{2}\int_{t_0}^{T} \left\| \frac{du}{dt}(t) \right\|_{Q_{du}}^2 dt, \tag{38}$$

such that the changes in the manipulated inputs over time, $u(t)$, are penalised quadratically. The OCP is also subject to state and algebraic dynamics, as well as box constraints on independent and dependent variables.

V. Numerical experiment

In this section, we present a simulation example for economic optimal control in a laboratory-scale CSTR for SCP production. Fig. 2 is an illustration of the fermentor.

A. Laboratory fermentor

In the physical laboratory fermentor, operation is separated into a batch phase and a continuous phase.

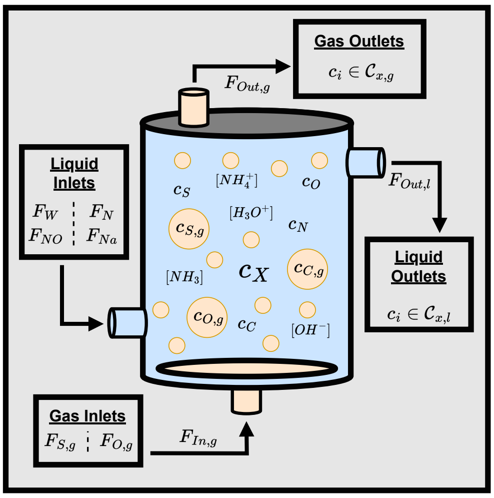


Fig. 2. Illustration of the laboratory-scale continuous stirred tank reactor.

Batch phase: The reactor is initialised from low biomass concentration, $c_X < 0.1$ g/L (inoculum), with substrates present for growth to a biomass concentration of $c_X \geq 2$ g/L. In this phase, only gasses flow in and out of the reactor, i.e. $F_i = 0$ for $i \in \mathcal{C}_{f,l}$ and $F_j \geq 0$ for $j \in \mathcal{C}_{f,g}$.

Continuous phase: The reactor is initialised from a biomass concentration high enough to consistently sustain continuous operation, i.e. $c_X \geq 2$ g/L. In the continuous phase, liquid substrates flow into the reactor. To keep the volume constant, liquid reactor content is harvested at the same rate as the inflow, i.e. $F_i \geq 0$ for $\mathcal{C}_{f,g}$, $F_j \geq 0$ for $i \in \mathcal{C}_{f,l}$, and $F_{Out,l} = \sum_{j \in \mathcal{C}_{f,l}} F_j$.

B. Implementation

We formulate the economic OCP for the continuous phase of the laboratory-scale CSTR. We implement the discretised OCP in Matlab with a simultaneous collocation-based approach [8], [9]. We apply the implicit Euler discretisation scheme (i.e. right rectangular rule). Casadi is applied to solve the resulting nonlinear program [16]. The initial state is $x_0 = [2.00, 2.31\text{e}{-2}, 3.77\text{e}{-2}, 4.03\text{e}{-1}, 9.10\text{e}{-1}, 3.07\text{e}{-3}, 2.23\text{e}{-7}, 6.29\text{e}{-1}, 1.11, 1.05]^T$ [g/L] and the initial input is $u_0 = [1.00\text{e}{-2}, 6.84\text{e}{-5}, 6.71\text{e}{-7}, 5.62\text{e}{-11}, 8.96\text{e}{-3}, 8.53\text{e}{-3}]^T$ [L/h]. The unit prices for substrates are $p_W = 0.00$, $p_N = 1.00\text{e}{-3}$, $p_{NO} = 1.00\text{e}{-1}$, $p_{Na} = 1.00\text{e}{-1}$, $p_{S,g} = 1.00\text{e}{-3}$, $p_{O,g} = 1.00\text{e}{-3}$ [USD/L] and for biomass $p_X = 1.0\text{e}{-2}$ [USD/g]. $\alpha_{eco} = 1.0$, $\alpha_{pH} = 2.0\text{e}{+2}$, and $\alpha_{du} = 1.0$ are the scaling parameters for the objectives in the OCP. The weight matrices for the pH tracking and input rate-of-movement objectives are $Q_{pH} = I$ and $Q_{du} = \text{diag}([1.0, 1.0, 10.0, 10.0, 0.1, 0.1])$, respectively. The algebraic equations are scaled with $s_g = [1.0\text{e}{+7}, 1.0\text{e}{+7}, 1.0\text{e}{+4}, 1.0\text{e}{+8}, 1.0\text{e}{+10}, 1.0\text{e}{+0}, 1.0\text{e}{+0}, 1.0\text{e}{+0}]$ and variables with $s_y = \{1\}^{n_y}$. The algebraic variables are parametrised in the implementation, such that $y(\alpha(t)) = 10^{-\alpha(t)}$, i.e. we describe the pH-value directly as $pH(t) = -\log_{10}(y_{H_3O^+}(t)) = \alpha_{H_3O^+}(t)$. This parametrisation also ensures non-negative during the optimisation. We solve the discretised OCP over a horizon of 48 hours with 200 discrete time-steps. All variables have a lower boundary of 0. The biomass and ammonium concentrations have upper boundaries $c_X \leq 20.0$ g/L and $c_N \leq 1.0$ M (17.03 g/L), respectively.

C. Results

Fig. 3 illustrates the biomass concentration, biomass productivity, pH-value, and liquid and gas inlet streams of the numerical experiment. Fig. 4 illustrates the concentrations of all state and algebraic variables in the numerical experiment. We observe clear separation between growth and production phases in the solution. We note that the nitrogen source, ammonium, is at the upper limit during production. This indicates that the nitrogen source is the limiting substrate during production.

VI. Conclusion

This paper presented a novel growth model for SCP production in a laboratory-scale CSTR. The DAE model describes reactor dynamics, growth kinetics of the micro-organism *M. capsulatus*, as well as pH equilibrium dynamics in the system. The model couples growth and pH as the growth directly depends on the algebraic variables. Finally, we conducted a numerical experiment of economic optimal control for the system. In the numerical experiment, we demonstrated optimal biomass growth and production, while tracking the pH-value in the reactor.

References

[1] OECD, Food, and A. O. of the United Nations, *OECD-FAO Agricultural Outlook 2022-2031*, 2022.

[2] J. Villadsen, J. Nielsen, and G. Lidén, *Bioreaction Engineering Principles*, 3rd ed. Springer Science+Business Media, 2011.

[3] L. A. Petersen, J. Villadsen, S. B. Jørgensen, and K. V. Gernaey, "Mixing and mass transfer in a pilot scale u-loop bioreactor," *Biotechnology and Bioengineering*, vol. 114, no. 2, pp. 344–354, 2017.

[4] D. F. Olsen, J. B. Jørgensen, J. Villadsen, and S. B. Jørgensen, "modeling and simulation of single cell protein production," in *proceedings of the 11th international symposium on computer applications in biotechnology*, Leuven, Belgium, July 7-9, 2010, pp. 502–507.

[5] ——, "Optimal operating points for scp production in the u-loop reactor," in *proceedings of the 9th international symposium on dynamics and control of process systems (DYCOPS)*, Leuven, Belgium, July 5-7, 2010, pp. 485–490.

[6] A. Drejer, T. K. S. Ritschel, S. B. Jørgensen, and J. B. Jørgensen, "Economic optimizing control for single-cell protein production in a U-loop reactor," in *Proceedings of the 27th European Symposium on Computer Aided Process Engineering (ESCAPE)*, Barcelona, Spain. October 1-5, 2017, pp. 1759–1764.

[7] T. K. S. Ritschel, D. Boiroux, M. K. Nielsen, J. K. Huusom, S. B. Jørgensen, and J. B. Jørgensen, "The extended Kalman filter for nonlinear state estimation in a U-loop bioreactor," in *proceedings of the 2019 IEEE Conference on Control Technology and Applications (CCTA)*, Hong Kong, China. August 19-21, 2019, pp. 920–925.

[8] ——, "Economic optimal control of a U-loop bioreactor using simultaneous collocation-based approaches," in *proceedings of the 2019 IEEE Conference on Control Technology and Applications (CCTA)*, Hong Kong, China. August 19-21, 2019, pp. 933–938.

[9] ——, "Economic nonlinear model predictive control of a U-loop bioreactor," in *proceedings of the 2020 European Control Conference (ECC)*, Saint Petersburg, Russia. May 12-15, 2020, pp. 208–213.

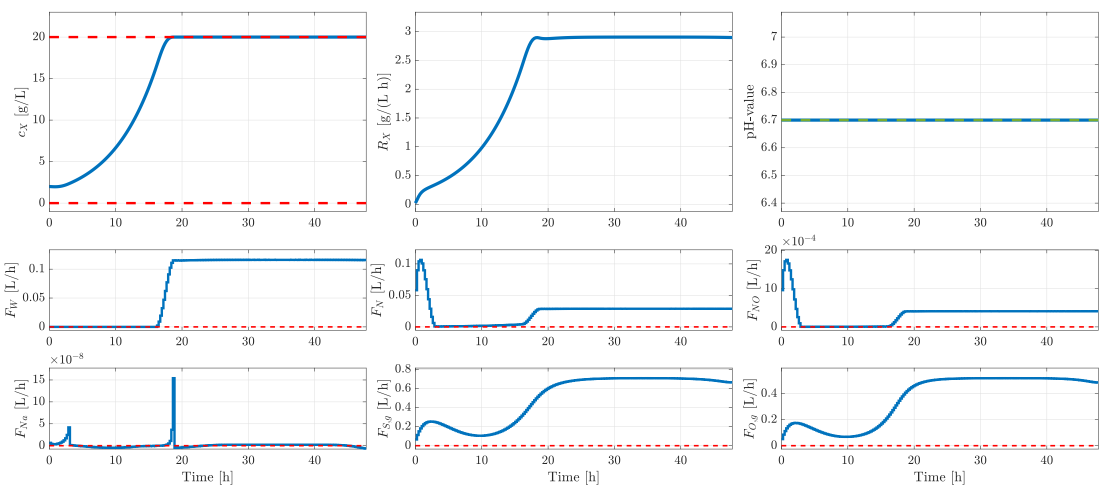


Fig. 3. Solution to economic optimal control problem for single-cell protein production in continuous-stirred tank reactor. The solution shows an initial growth phase of around 18 hours, followed by a production phase with increased inlet flow-rate of water for stable and high productivity.

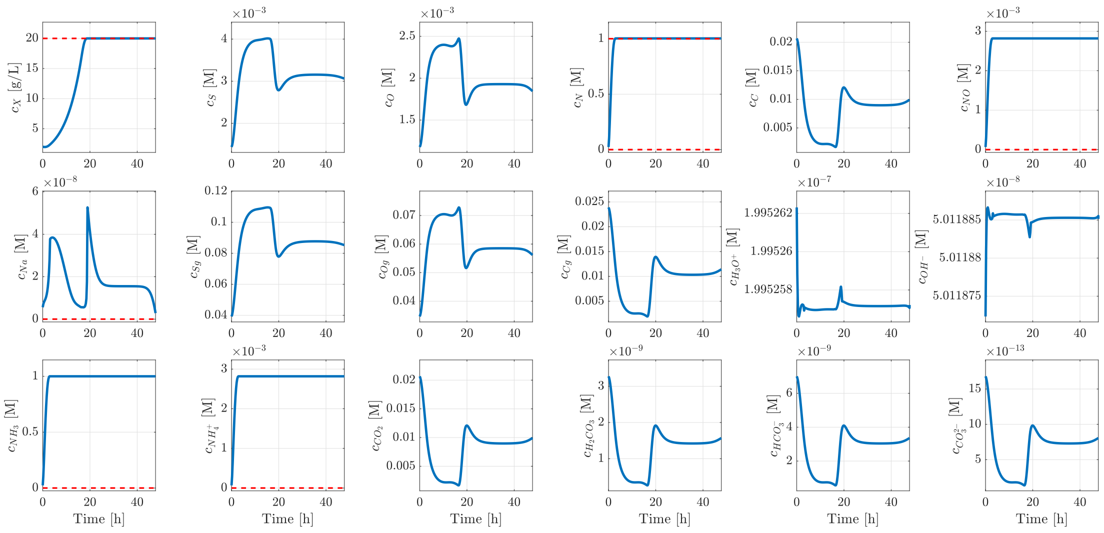


Fig. 4. Concentrations of all state and algebraic variables computed as the solution to the economic optimal control problem.

[10] C. Lieven, L. A. Petersen, S. B. Jørgensen, K. V. Gernaey, M. J. Herrgard, and N. Sonnenschein, "A genome-scale metabolic model for methylococcus capsulatus (bath) suggests reduced efficiency electron transfer to the particulate methane monooxygenase," *Frontiers in microbiology*, vol. 9, p. 2947, 2018.

[11] L. A. Petersen, C. Lieven, S. K. Nandy, J. Villadsen, S. B. Jørgensen, I. Christensen, and K. V. Gernaey, "Dynamic investigation and modeling of the nitrogen cometabolism in methylococcus capsulatus (bath)," *Biotechnology and Bioengineering*, vol. 116, no. 11, pp. 2884–2895, 2019.

[12] L. A. Petersen, B. W. Bequette, S. B. Jørgensen, J. Villadsen, I. Christensen, and K. V. Gernaey, "Modeling and system identification of an unconventional bioreactor used for single cell protein production," *Chemical Engineering Journal*, vol. 390, p. 124438, 2020.

[13] O. A. Prado-Rubio, J. B. Jørgensen, and S. B. Jørgensen, "Systematic model analysis for single cell protein (scp) production in a u-loop reactor," in *Computer aided chemical engineering*. Elsevier, 2010, vol. 28, pp. 319–324.

[14] R. Sander, "Compilation of henry's law constants (version 4.0) for water as solvent," *Atmospheric Chemistry and Physics*, vol. 15, no. 8, pp. 4399–4981, 2015.

[15] K. Meintjes and A. P. Morgan, "A methodology for solving chemical equilibrium systems," *Applied Mathematics and Computation*, vol. 22, no. 4, pp. 333–361, 1987.

[16] J. A. E. Anderson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

APPENDIX B
Paper II - FOCAPO/CPC 2023

State estimation in continuous-discrete-time nonlinear stochastic systems

Authors:
Marcus Krogh Nielsen, Tobias K. S. Ritschel, Ib Christensen, Jess Dragheim, Jakob Kjøbsted Huusom, Krist V. Gernaey, John Bagterp Jørgensen

Accepted for:
2023 foundations for computer-aided process operation/chemical process control (FOCAPO/CPC). January 8-12, 2023. San Antonio, Texas, USA.

State Estimation for Continuous-Discrete-Time Nonlinear Stochastic Systems

Marcus Krogh Nielsen[a,c], Tobias K. S. Ritschel[a], Ib Christensen[c], Jess Dragheim[c],
Jakob Kjøbsted Huusom[b], Krist V. Gernaey[b], John Bagterp Jørgensen[a, 1]

[a] Department of Applied Mathematics and Compute Science, DTU Compute, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark.

[b] Department of Chemical and Biochemical Engineering, DTU Chemical Engineering, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark.

[c] Unibio A/S, DK-4000 Roskilde, Denmark.

Abstract

State estimation incorporates the feedback in optimization based advanced process control systems and is very important for the performance of model predictive control. We describe the extended Kalman filter, the unscented Kalman filter, the ensemble Kalman filter, and a particle filter for continuous-discrete time nonlinear systems involving stochastic differential equations. Continuous-discrete time nonlinear systems is a natural way to model physical systems controlled by digital controllers. We implement the state estimation methods in Matlab, illustrate and evaluate their performance using simulations of the modified four-tank system. This system is non-stiff and the state estimation methods are implemented numerically using an explicit numerical integration scheme. We evaluate the accuracy of the state estimation methods in terms of the mean absolute percentage error over the simulation horizon. Each method successfully estimates the states and unmeasured disturbances of the simulated modified four-tank system. The key contribution is an overview and comparison of state estimation methods for continuous-discrete time nonlinear stochastic systems. This can guide efficient implementations.

Introduction

State estimation is widely applied in advanced process control (APC) systems, e.g. for monitoring, fault-detection, and as part of model predictive control (MPC). The objective of state estimation is to predict and reconstruct the states of a mathematical model using measurements from a physical system. The Kalman filter provides optimal estimates for systems with Gaussian process and measurement noise, but is limited to system with linear dynamics (Kalman, 1960). For nonlinear systems, the exact evolution of the state distribution can be computed as the solution to the Fokker-Planck equation (Kolmogorov's forward equation). However, the Fokker-Planck equation is a partial differential equation where the number of dimensions equal the number of states in the system. The Fokker-Planck equation suffers from the curse of dimensionality and solving it is therefore impractical for systems with more than a few states (Jazwinski, 2007). This paper describes four approximate methods for state estimation, 1) the extended Kalman filter (EKF), 2) the unscented Kalman filter (UKF), 3) the ensemble Kalman filter (EnKF), and 4) a particle filter (PF).

In the EKF, the equations of the original Kalman filter are applied on a local linearisation of a nonlinear system (Rawlings et al., 2017). The EKF is a computationally efficient method, but the quality of the estimates depend on the nonlinearity of the system (Frogerais et al., 2011). Additionally, some stability issues may arise in relation to fixed step-size solutions (Bucy and Joseph, 2005; Jørgensen et al., 2007). In the UKF, an unscented transformation is used as an approximation for the first two moments of the true nonlinear distribution. The unscented transformation is propagated through the nonlinear dynamics and each sigma-point is updated using observations from the physical system (Julier and Uhlmann, 2004). For some systems, the UKF has shown higher accuracy than the EKF, while still being computationally efficient (Wan and van der Merwe, 2000). However, the UKF also suffers from inaccuracy in highly nonlinear systems. For the UKF, some of the issues pertaining to nonlinearity and numerical instability have been addressed in more recent contributions (Kandepu et al., 2008; De Vivo et al., 2017). In the EnKF, a set of particles, the ensemble, is randomly sampled from the state distribution and propagated through the nonlinear system dynamics. Each particle in the ensemble is updated separately using the Kalman filter update when a measurement becomes available. The state estimates are computed statistically from the ensembles (Gillijns et al., 2006; Myrseth and Omre, 2010; Roth et al., 2015). In PFs, a set of particles is sampled from the state distribution and propagated through the nonlinear system dynamics. When a measurement becomes available, the particles are resampled in accordance with their likelihood

[1] Corresponding author: J. B. Jørgensen (E-mail: `jbjo@dtu.dk`).

of being observed. The likelihoods are computed using the innovation posterior distribution. Similarly to the EnKF, the state estimates are determined statistically from the particles (Arulampalam et al., 2002; Rawlings and Bakshi, 2006; Tulsyan et al., 2016). The EKF and UKF provide efficient state estimation, but suffers from loss of accuracy for highly nonlinear systems. The EnKF and PFs provide a set of sampled particles from the true nonlinear distribution, but their computational efficiency depends on the number of particles required for the estimates to reach the desired accuracy. As a result of this, the EnKF and PF can be computationally inefficient for high dimensional systems.

The aim of this paper is to provide a condensed overview of available methods for state estimation in *continuous-discrete time* nonlinear stochastic systems that are described using stochastic differential equations (SDEs). The intention is to guide efficient implementation by providing an overview for the continuous-discrete nonlinear state estimation methods. The nonlinear state estimation methods can generally be separated into two steps; prediction and filtering. In the prediction step (time update), the system is propagated through time based on past information from a physical system. In the filtering step (measurement update), the state estimates are updated with the latest measurement information.

The paper is structured as follows. In Section 2, we present the nonlinear continuous-discrete stochastic differential equation models used in simulation and state estimation. In Section 3, we present the EKF, the UKF, the EnKF, and a PF, and finally present a discussion of the methods. In Section 4, we present a numerical example of state estimation for a modified four-tank system. Finally, we present conclusions in Section 5.

Nonlinear Continuous-Discrete Stochastic Systems

We consider continuous-discrete systems, in which the system state is described by nonlinear continuous stochastic differential equations and measurements are taken at discrete points in time. The nonlinear continuous-discrete stochastic differential equation models are defined as

$$\begin{aligned} d\boldsymbol{x}(t) &= f(t,\boldsymbol{x}(t),u(t),d(t),\theta)dt \\ &\quad + \sigma(t,\boldsymbol{x}(t),u(t),d(t),\theta)d\boldsymbol{\omega}(t), \end{aligned} \tag{1a}$$

$$\boldsymbol{y}(t_k) = h(t_k,\boldsymbol{x}(t_k),\theta) + \boldsymbol{v}(t_k), \tag{1b}$$

where $f(\cdot)$ is the drift function, $\sigma(\cdot)$ is the diffusion function, and $h(\cdot)$ is the measurement function. The states are $\boldsymbol{x}(t) \in \mathbb{R}^{n_x}$, the inputs are $u(t) \in \mathbb{R}^{n_u}$, the disturbances are $d(t) \in \mathbb{R}^{n_d}$, the parameters are $\theta \in \mathbb{R}^{n_\theta}$, and the measurements are $\boldsymbol{y}(t_k) \in \mathbb{R}^{n_y}$. The process noise $\boldsymbol{\omega}(t) \in \mathbb{R}^{n_\omega}$ is a standard Wiener process, such that $d\boldsymbol{\omega}(t) \sim \mathcal{N}(0, Idt)$, and $\boldsymbol{v}(t_k) \sim \mathcal{N}(0,R)$ is the measurement noise. The initial state is assumed to be distributed as $\boldsymbol{x}_0 \sim \mathcal{N}(\bar{\boldsymbol{x}}_0, P_0)$.

State Estimation in Nonlinear Systems

In this section, we present methods for state estimation in continuous-discrete nonlinear systems (1). These estimators are called continuous-discrete estimators.

Continuous-discrete extended Kalman filter

The EKF is initialised with the mean and covariance of the initial state of the system described in Section 2

$$\hat{x}_{0|0} = \bar{x}_0, \qquad P_{0|0} = P_0. \tag{2}$$

Time update: In the time update, the mean and covariance are computed as the solution to the ordinary differential equations (ODEs) for $t \in [t_k, t_{k+1}]$

$$\frac{d\hat{x}_k(t)}{dt} = f(t,\hat{x}_k(t),u(t),d(t),\theta), \tag{3a}$$

$$\frac{dP_k(t)}{dt} = A_k(t)P_k(t) + P_k(t)A_k^T(t) + \sigma_k(t)\sigma_k^T(t), \tag{3b}$$

where $\hat{x}_k(t_k) = \hat{x}_{k|k}$, and $P_k(t_k) = P_{k|k}$. $A_k(t) = \frac{\partial f}{\partial x}(t,\hat{x}_k(t),u(t),d(t),\theta)$ and $\sigma_k(t) = \sigma(t,\hat{x}_k(t),u(t),d(t),\theta)$. Alternatively, the covariance update can be represented and solved on integral form as presented by Jørgensen et al. (2007). The mean and covariance estimates are

$$\hat{x}_{k+1|k} = \hat{x}_k(t_{k+1}), \qquad P_{k+1|k} = P_k(t_{k+1}). \tag{4}$$

Measurement update: In the measurement update, we compute the innovation and its covariance as

$$e_k = y_k - \hat{y}_{k|k-1}, \qquad R_{e,k} = C_k P_{k|k-1} C_k^T + R, \tag{5}$$

where

$$\hat{y}_{k|k-1} = h(t_k,\hat{x}_{k|k-1},\theta), \qquad C_k = \frac{\partial h}{\partial x}(t_k,\hat{x}_{k|k-1},\theta). \tag{6}$$

The Kalman gain is computed as

$$K_{f_x,k} = P_{k|k-1} C_k^T R_{e,k}^{-1}. \tag{7}$$

The mean and covariance estimates are computed as

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_{f_x,k} e_k, \tag{8a}$$

$$P_{k|k} = P_{k|k-1} - K_{f_x,k} R_{e,k} K_{f_x,k}^T \tag{8b}$$

$$= \left(I - K_{f_x,k} C_k\right) P_{k|k-1} \left(I - K_{f_x,k} C_k\right)^T + K_{f_x,k} R K_{f_x,k}^T, \tag{8c}$$

where (8c), Joseph's stabilising form, is numerically stable.

Continuous-discrete unscented Kalman filter

The unscented Kalman filter is initialised with the mean and covariance of the initial state of the system described in Section 2

$$\hat{x}_{0|0} = \bar{x}_0, \qquad P_{0|0} = P_0. \tag{9}$$

Time update: In the time update, we compute the parameters

$$\bar{c} = \alpha^2 (\bar{n} + \kappa), \tag{10}$$

$$\bar{\lambda} = \alpha^2 (\bar{n} + \kappa) - \bar{n}, \tag{11}$$

where $\alpha \in]0,1]$, $\kappa \in [0,\infty[$, and $\bar{n} = n_x + n_\omega$. We compute the sigma-point weights

$$\bar{W}_m^{(0)} = \frac{\bar{\lambda}}{\bar{n}+\bar{\lambda}}, \tag{12a}$$

$$\bar{W}_c^{(0)} = \frac{\bar{\lambda}}{\bar{n}+\bar{\lambda}} + 1 - \alpha^2 + \beta, \tag{12b}$$

$$\bar{W}_m^{(i)} = \bar{W}_c^{(i)} = \frac{1}{2\left(\bar{n}+\bar{\lambda}\right)}, \tag{12c}$$

for $i \in \{1,2,\dots,2\bar{n}\}$ and where $\beta \in [0,\infty[$ ($\beta = 2$ optimal for Gaussian distributions). We sample deterministically a set of $2\bar{n}+1$ sigma-points. For propagation through the deterministic dynamics (ODE), we compute $2n_x+1$ sigma-points

$$\hat{x}_{k|k}^{(0)} = \hat{x}_{k|k}, \tag{13a}$$

$$\hat{x}_{k|k}^{(i)} = \hat{x}_{k|k} + \sqrt{c}\left(\sqrt{P_{k|k}}\right)_i, \tag{13b}$$

$$\hat{x}_{k|k}^{(n_x+i)} = \hat{x}_{k|k} - \sqrt{c}\left(\sqrt{P_{k|k}}\right)_i, \tag{13c}$$

for $i \in \{1,2,\dots,n_x\}$. $\left(\sqrt{P_{k|k}}\right)_i$ denotes the i'th column of the Cholesky decomposition of the covariance. For propagation through the stochastic dynamics, we compute $2n_\omega$ sigma-points

$$\hat{x}_{k|k}^{(2n_x+i)} = \hat{x}_{k|k}, \tag{14}$$

for $i \in \{1,2,\dots,2n_\omega\}$. Additionally, we compute the process noise

$$d\omega_k^{(2n_x+i)}(t) = \sqrt{c\,dt}\,(I)_i, \tag{15a}$$

$$d\omega_k^{(2n_x+n_\omega+i)}(t) = -\sqrt{c\,dt}\,(I)_i, \tag{15b}$$

where $i \in \{1,2,\dots,n_\omega\}$. We propagate the first sigma-points through the deterministic dynamics for $t \in [t_k, t_{k+1}]$ and compute the predictions as the solution to

$$d\hat{x}_k^{(i)}(t) = f(t,\hat{x}_k^{(i)}(t),u(t),d(t),\theta)dt, \tag{16}$$

for $\hat{x}_k^{(i)}(t_k) = \hat{x}_{k|k}^{(i)}$ and $i \in \{0,1,\dots,2n_x\}$. We similarly propagate the remaining sigma-points through the stochastic dynamics for $t \in [t_k, t_{k+1}]$ and compute the predictions as the solution to

$$\begin{aligned} d\hat{x}_k^{(i)}(t) &= f(t,\hat{x}_k^{(i)}(t),u(t),d(t),\theta)dt \\ &\quad + \sigma(t,\hat{x}_k^{(i)}(t),u(t),d(t),\theta)d\omega_k^{(i)}(t), \end{aligned} \tag{17}$$

where $\hat{x}_k^{(i)}(t_k) = \hat{x}_{k|k}^{(i)}$ and $i \in \{2n_x+1, 2n_x+2, \dots, 2n_x+2n_\omega\}$. The predictions are computed as the solution to (16) and (17), as $\hat{x}_{k+1|k}^{(i)} = \hat{x}_k^{(i)}(t_{k+1})$. The mean and covariance estimates are computed as

$$\hat{x}_{k+1|k} = \sum_{i=0}^{2\bar{n}} \bar{W}_m^{(i)} \hat{x}_{k+1|k}^{(i)}, \tag{18a}$$

$$P_{k+1|k} = \sum_{i=0}^{2\bar{n}} \bar{W}_c^{(i)} \left(\hat{x}_{k+1|k}^{(i)} - \hat{x}_{k+1|k}\right)\left(\hat{x}_{k+1|k}^{(i)} - \hat{x}_{k+1|k}\right)^T. \tag{18b}$$

Measurement update: In the measurement update, we compute the parameters

$$c = \alpha^2\left(n_x+\kappa\right), \qquad \lambda = \alpha^2\left(n_x+\kappa\right) - n_x. \tag{19}$$

We compute the sigma-point weights

$$W_m^{(0)} = \frac{\lambda}{n_x+\lambda}, \tag{20a}$$

$$W_c^{(0)} = \frac{\lambda}{n_x+\lambda} + 1 - \alpha^2 + \beta, \tag{20b}$$

$$W_m^{(i)} = W_c^{(i)} = \frac{1}{2\left(n_x+\lambda\right)}, \tag{20c}$$

for $i \in \{1,2,\dots,2n_x\}$. We compute a set of $2n_x+1$ deterministically sampled sigma-points

$$\hat{x}_{k|k-1}^{(0)} = \hat{x}_{k|k-1}, \tag{21a}$$

$$\hat{x}_{k|k-1}^{(i)} = \hat{x}_{k|k-1} + \sqrt{c}\left(\sqrt{P_{k|k-1}}\right)_i, \tag{21b}$$

$$\hat{x}_{k|k-1}^{(n_x+i)} = \hat{x}_{k|k-1} - \sqrt{c}\left(\sqrt{P_{k|k-1}}\right)_i, \tag{21c}$$

for $i \in \{1,2,\dots,n_x\}$. We compute the innovation as

$$e_k = y_k - \hat{y}_{k|k-1}, \tag{22}$$

where the prediction of the measurement prediction is computed as

$$\hat{y}_{k|k-1} = \hat{z}_{k|k-1} = \sum_{i=0}^{2n_x} W_m^{(i)} \hat{z}_{k|k-1}^{(i)}, \tag{23}$$

for $\hat{z}_{k|k-1}^{(i)} = h(t_k, \hat{x}_{k|k-1}^{(i)}, \theta)$. We compute the covariance and cross-covariance information from the sigma-points

$$R_{zz,k|k-1} = \sum_{i=0}^{2n_x} W_c^{(i)} \left(\hat{z}_{k|k-1}^{(i)} - \hat{z}_{k|k-1}\right)\left(\hat{z}_{k|k-1}^{(i)} - \hat{z}_{k|k-1}\right)^T, \tag{24a}$$

$$R_{e,k} = R_{yy,k|k-1} = R_{zz,k|k-1} + R, \tag{24b}$$

$$R_{xy,k|k-1} = \sum_{i=0}^{2n_x} W_c^{(i)} \left(\hat{x}_{k|k-1}^{(i)} - \hat{x}_{k|k-1}\right)\left(\hat{z}_{k|k-1}^{(i)} - \hat{z}_{k|k-1}\right)^T. \tag{24c}$$

The Kalman gain is computed as

$$K_{f_x,k} = R_{xy,k|k-1} R_{e,k}^{-1}. \tag{25}$$

The mean and covariance estimates are computed as

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_{f_x,k} e_k, \tag{26a}$$

$$P_{k|k} = P_{k|k-1} - K_{f_x,k} R_{e,k} K_{f_x,k}^T. \tag{26b}$$

Continuous-discrete ensemble Kalman filter

The ensemble Kalman filter is initialised with a set of particles, the ensemble, sampled from the initial state distribution from (2). The initial state ensemble is denoted $\{\hat{x}_{0|0}^{(i)}\}_{i=1}^{N_p}$.

Time update: In the time update, each particle in the ensemble is propagated through the system dynamics. The prediction ensemble is computed as the solution to

$$\begin{aligned} d\boldsymbol{x}_k^{(i)}(t) &= f(t,\boldsymbol{x}_k^{(i)}(t),u(t),d(t),\theta)dt \\ &\quad + \sigma(t,\boldsymbol{x}_k^{(i)}(t),u(t),d(t),\theta)d\boldsymbol{\omega}_k(t), \end{aligned} \tag{27}$$

for $i \in \{1,2,\dots,N_p\}$ and $t \in [t_k,t_{k+1}]$. The initial value is $x_k^{(i)} = \hat{x}_{k|k}^{(i)}$. The set of solutions, $\hat{x}_{k+1|k}^{(i)} = x_k^{(i)}(t_{k+1})$, gives rise to the prediction ensemble $\{\hat{x}_{k+1|k}^{(i)}\}_{i=1}^{N_p}$. The mean and covariance estimates are computed as

$$\hat{x}_{k+1|k} = \frac{1}{N_p}\sum_{i=1}^{N_p}\hat{x}_{k+1|k}^{(i)}, \tag{28a}$$

$$P_{k+1|k} = \frac{1}{N_p-1}\sum_{i=1}^{N_p}\left(\hat{x}_{k+1|k}^{(i)} - \hat{x}_{k+1|k}\right)\left(\hat{x}_{k+1|k}^{(i)} - \hat{x}_{k+1|k}\right)^T. \tag{28b}$$

Measurement update: In the measurement update, we compute the ensemble of predictions, $\{\hat{z}_{k|k-1}^{(i)}\}_{i=1}^{N_p}$, where $z_{k|k-1}^{(i)} = h(t_k,\hat{x}_{k|k-1}^{(i)},\theta)$, for $i \in \{1,2,\dots,N_p\}$. Furthermore, we compute the mean and covariance of the measurement distribution and cross-covariance of states and measurements, as

$$\hat{y}_{k|k-1} = \hat{z}_{k|k-1} = \frac{1}{N_p}\sum_{i=1}^{N_p}\hat{z}_{k|k-1}^{(i)}, \tag{29a}$$

$$R_{zz,k|k-1} = \frac{1}{N_p-1}\sum_{i=1}^{N_p}\left(\hat{z}_{k|k-1}^{(i)} - \hat{z}_{k|k-1}\right)\left(\hat{z}_{k|k-1}^{(i)} - \hat{z}_{k|k-1}\right)^T, \tag{29b}$$

$$R_{yy,k|k-1} = R_{zz,k|k-1} + R, \tag{29c}$$

$$R_{xy,k|k-1} = \frac{1}{N_p-1}\sum_{i=1}^{N_p}\left(\hat{x}_{k|k-1}^{(i)} - \hat{x}_{k|k-1}\right)\left(\hat{y}_{k|k-1}^{(i)} - \hat{y}_{k|k-1}\right)^T, \tag{29d}$$

and we compute samples from measurement distribution, as

$$y_k^{(i)} = y_k + v_k^{(i)}, \tag{30}$$

where $v_k^{(i)}$ are realisations of the measurement noise, $\boldsymbol{v}_k \sim \mathcal{N}(0,R)$. The innovations are computed for each particle in the measurement ensemble, as

$$e_k^{(i)} = y_k^{(i)} - \hat{z}_{k|k-1}^{(i)}. \tag{31}$$

The Kalman gain is computed as

$$K_{fx,k} = R_{xy,k|k-1}R_{yy,k|k-1}^{-1}. \tag{32}$$

The filtered state ensemble, $\{\hat{x}_{k|k}^{(i)}\}_{i=1}^{N_p}$, is computed as

$$\hat{x}_{k|k}^{(i)} = \hat{x}_{k|k-1}^{(i)} + K_{fx,k}e_k^{(i)}. \tag{33}$$

The mean and covariance estimates are computed as

$$\hat{x}_{k|k} = \frac{1}{N_p}\sum_{i=1}^{N_p}\hat{x}_{k|k}^{(i)}, \tag{34a}$$

$$P_{k|k} = \frac{1}{N_p-1}\sum_{i=1}^{N_p}\left(\hat{x}_{k|k}^{(i)} - \hat{x}_{k|k}\right)\left(\hat{x}_{k|k}^{(i)} - \hat{x}_{k|k}\right)^T. \tag{34b}$$

Continous-discrete particle filter

The particle filter is initialised with a set of particles sampled from the initial state distribution from (2). The initial set of particles is denoted $\{\hat{x}_{0|0}^{(i)}\}_{i=1}^{N_p}$.

Time update: In the time update, each particle is propagated through the nonlinear system dynamics. The set of predicted particles is computed as the solution to

$$\begin{aligned} d\boldsymbol{x}_k^{(i)}(t) &= f(t,\boldsymbol{x}_k^{(i)}(t),u(t),d(t),\theta)dt \\ &\quad + \sigma(t,\boldsymbol{x}_k^{(i)}(t),u(t),d(t),\theta)d\boldsymbol{\omega}_k(t), \end{aligned} \tag{35}$$

for $i \in \{1,2,\dots,N_p\}$ and $t \in [t_k,t_{k+1}]$. The initial value $x_k^{(i)} = \hat{x}_{k|k}^{(i)}$. The set of solutions, $\hat{x}_{k+1|k}^{(i)} = x_k^{(i)}(t_{k+1})$, gives rise to the prediction set $\{\hat{x}_{k+1|k}^{(i)}\}_{i=1}^{N_p}$. The mean and covariance estimates are computed as

$$\hat{x}_{k+1|k} = \frac{1}{N_p}\sum_{i=1}^{N_p}\hat{x}_{k+1|k}^{(i)}, \tag{36a}$$

$$P_{k+1|k} = \frac{1}{N_p-1}\sum_{i=1}^{N_p}\left(\hat{x}_{k+1|k}^{(i)} - \hat{x}_{k+1|k}\right)\left(\hat{x}_{k+1|k}^{(i)} - \hat{x}_{k+1|k}\right)^T. \tag{36b}$$

Measurement update: In the measurement update, we compute the set of measurement predictions, $\{\hat{z}_{k|k-1}^{(i)}\}_{i=1}^{N_p}$, where $\hat{z}_{k|k-1}^{(i)} = h(t_k,\hat{x}_{k|k-1}^{(i)},\theta)$, for $i \in \{1,2,\dots,N_p\}$. The innovations are computed for each particle, as

$$e_k^{(i)} = y_k - \hat{z}_{k|k-1}^{(i)}, \tag{37}$$

for $i \in \{1,2,\dots,N_p\}$. We compute a set of likelihood weights for each particle, arising from the posterior distribution of the innovations

$$\tilde{w}_k^{(i)} = \frac{1}{\sqrt{2\pi^{n_y}|R|}}\exp\left(-\frac{1}{2}\left(e_k^{(i)}\right)^T R^{-1}e_k^{(i)}\right), \tag{38}$$

where $|R|$ denotes the determinant of R, and normalise

$$w_k^{(i)} = \frac{\tilde{w}_k^{(i)}}{\sum_{j=1}^{N_p}\tilde{w}_k^{(j)}}, \tag{39}$$

for $i \in \{1,2,\dots,N_p\}$. The set of particles are then resampled in accordance with their likelihood respective weights. For a single realisation of a uniform distribution, $q_1 \sim \mathcal{U}[0,1]$, we compute a set of ordered resampling points

$$q_k^{(i)} = \frac{(i-1)+q_1}{N_p}, \tag{40}$$

for $i \in \{1,2,\dots,N_p\}$. We resample the particles by storing $m^{(i)}$ copies of each particle, $\hat{x}_{k|k-1}^{(i)}$, in the set. The indicies for the resampled particles, l, are chosen such that $q_k^{(l)} \in \left]s^{(i-1)},s^{(i)}\right]$, where $s^{(i)} = \sum_{j=1}^{i} w_k^{(j)}$. Particles with relatively high likelihood may appear several times in the resampled set and particles with relatively low likelihood may not appear at all. The resampled set is denoted as $\{\hat{x}_{k|k}^{(i)}\}_{i=1}^{N_p}$. The mean and covariance estimates are computed as

$$\hat{x}_{k|k} = \frac{1}{N_p}\sum_{i=1}^{N_p} \hat{x}_{k|k}^{(i)}, \tag{41a}$$

$$P_{k|k} = \frac{1}{N_p - 1}\sum_{i=1}^{N_p} \left(\hat{x}_{k|k}^{(i)} - \hat{x}_{k|k}\right)\left(\hat{x}_{k|k}^{(i)} - \hat{x}_{k|k}\right)^T. \tag{41b}$$

Discussion of methods

The EKF is a computationally efficient method for systems with a moderate number of states, while it is infeasible for systems with a very large number of states. The complexity in implementing the method is largely determined by computational aspects of solving the initial value problem (3) and issues related to computation of the covariance matrix. The accuracy of the EKF depends on how well the assumption of local linearity holds. This means that for highly nonlinear systems with relatively long sampling intervals, the EKF may perform poorly, as the assumptions pertaining to the propagation of the expectation and covariance will not hold. The UKF is comparable to the EKF in terms of its computational requirements. The computational efficiency partly arises by utilising the unscented transformation, where the number of deterministically sampled particles scales linearly with the state dimension, instead of randomly sampling particles, as is the case for other particle filters. The time update of the UKF is simple to implement, as it simply involves propagating a set of particles forward in time. For linear Gaussian systems, the UKF and EKF provide equivalent solutions. However, for nonlinear systems, the UKF propagates the particles through the true nonlinear system dynamics and therefore may capture more information than the EKF. The particle filters, i.e. the EnKF and PF, have computational efficiency which depends on the tuning, i.e. the size of the sample set. They suffer from the curse of dimensionality, as the sampling size required increases with the state dimension. However, the predictions more closely resemble the true nonlinear distribution as the sampling size increases, at the cost of computational efficiency. This means, that for highly nonlinear systems the EnKF and PF may capture more information than the EKF and UKF, but at the cost of computational efficiency. Nevertheless, the EnKF is often used for large-scale systems, but with few samples.

Example – Modified Four-Tank System (MFTS)

The modified four tank system is modelled by a set of ODEs describing mass balances as presented by Azam and Jørgensen (2015). The model is further modified by modelling the stochastic disturbances explicitly as states. The disturbances are governed by the stochastic processes

$$d\boldsymbol{F}_3(t) = \lambda_1 (\bar{F}_3(t) - \boldsymbol{F}_3(t))\,dt + \sigma_1 d\boldsymbol{\omega}_1(t), \tag{42a}$$

$$d\boldsymbol{F}_4(t) = \lambda_2 (\bar{F}_4(t) - \boldsymbol{F}_4(t))\,dt + \sigma_2 d\boldsymbol{\omega}_2(t). \tag{42b}$$

The resulting system is described by a continuous-discrete nonlinear system as described in (1).The performance of each state estimation method is evaluated in terms of the mean absolute percentage error (MAPE) , such that

$$MAPE = \frac{1}{nN}\sum_{k=1}^{N}\sum_{i=1}^{n}\left|\frac{x_{i,k} - \hat{x}_{i,k}}{x_{i,k}}\right|, \tag{43}$$

where N is the number of observations and n is the dimension of the state. The MAPE is computed separately for the states representing the liquid mass and the state representing the disturbances, as MAPE$_x$ and MAPE$_d$ respectively.

Table 1: run-times for time update (TU) and measurement update (MU), and MAPE for states (MAPE$_x$) and disturbances (MAPE$_d$).

name	**EKF**	**UKF**	**EnKF**	**PF**
time TU [s]	3.09e-01	2.90e+00	3.38e+01	1.36e+02
time MU [s]	1.22e-02	4.14e-02	2.30e-01	1.05e+00
MAPE$_x$ [%]	2.55e+00	2.97e+00	2.35e+00	2.40e+00
MAPE$_d$ [%]	1.57e+01	1.75e+01	1.47e+01	1.37e+01

Simulation example

Fig. 1 illustrates the simulation of the modified four tank system and Table 1 describes the results of the example. We simulate for 30 minutes with 120 equidistant samples. The simulation and estimation are computed with 1000 and 100 equidistant steps between samples, respectively. The UKF has the parameter set $[\beta, \alpha, \kappa] = [2.0, 0.001, 0.0]$. The EnKF and PF has particle set sizes of 250 and 1000, respectively. The disturbances are modelled with $\lambda_1 = \lambda_2 = 0.1$ and $\sigma_1 = \sigma_2 = 5.0$ for the simulation. For the EKF, EnKF, and PF, $\sigma_1 = \sigma_2 = 5.0$ and for the UKF $\sigma_1 = \sigma_2 = 1.0$. $\lambda_1 = \lambda_2 = 0.0$ for the EKF and UKF and $\lambda_1 = \lambda_2 = 2.0\text{e-}3$ for the EnKF and PF. From the results presented in Fig. 1 and Table 1, we see many of the properties described in the discussion of Section 3. The EKF and UKF are demonstrated the be the most computationally efficient methods, where EKF seem to be outperforming UKF in this particular numerical experiment. Furthermore, the EnKF and PF show better accuracy both in estimating state and disturbance variables, but at the cost of lower computational efficiency.

Conclusion

We present four methods for state estimation in continuous-discrete nonlinear systems involving stochastic differential equations: the EKF, the UKF, the EnKF, and a PF. The state estimation methods are implemented for non-stiff systems in Matlab, and a numerical experiment is performed for a simulated MFTS. The performance of each state estimation method is evaluated in terms of 1) the computational times for the time- and measurement-updates and 2) the accuracy measured by the MAPE for the state and disturbance estimates.

References

Arulampalam, M. S., S. Maskell, N. Gordon, and T. Clapp (2002). A tutorial on particle filters for online

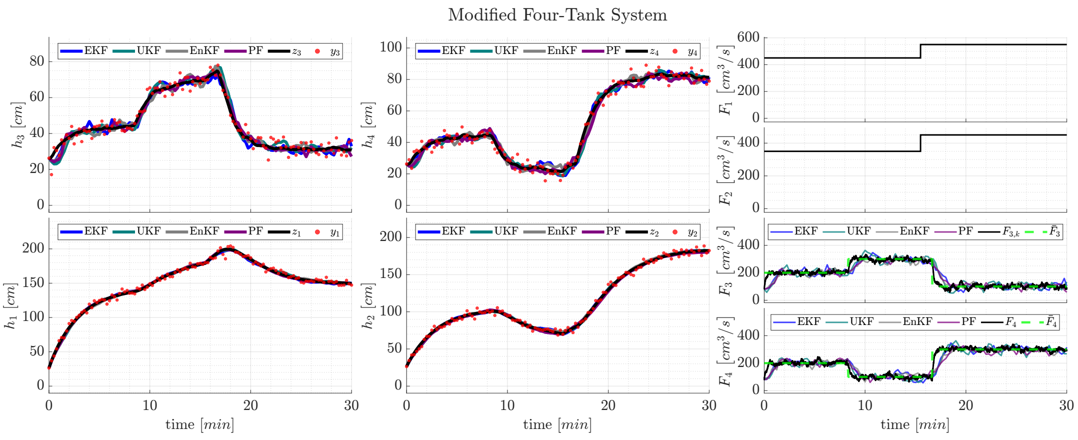


Figure 1: State estimation for simulation of the modified four-tank system. For the simulation model $\bar{F}_i \in \{100, 200, 300\}$ for $i \in \{3, 4\}$. The nominal values are kept constant at 150 for the state estimators.

nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing 50*(2), 174–188.

Azam, S. N. M. and J. B. Jørgensen (2015). Modeling and simulation of a modified quadruple tank system. In *proceedings of the 2015 IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*, Penang, Malaysia. November 27-29, pp. 365–370.

Bucy, R. S. and P. D. Joseph (2005). *Filtering for stochastic processes with applications to guidance*. American Mathematical Society.

De Vivo, F., A. Brandl, M. Battipede, and P. Gili (2017). Joseph covariance formula adaptation to square-root sigma-point Kalman filters. *Nonlinear Dynamics 88*(3), 1969–1986.

Frogerais, P., J.-J. Bellanger, and L. Senhadji (2011). Various ways to compute the continuous-discrete extended Kalman filter. *IEEE Transactions on Automatic Control 57*(4), 1000–1004.

Gillijns, S., O. B. Mendoza, J. Chandrasekar, B. L. R. De Moor, D. S. Bernstein, and A. Ridley (2006). What is the ensemble Kalman filter and how well does it work? In *proceedings of the 2006 American Control Conference (ACC)*, Minneapolis, M.N., USA. June 14-16, pp. 4448–4453.

Jazwinski, A. H. (2007). *Stochastic processes and filtering theory*. Dover Publication, Inc.

Jørgensen, J. B., M. R. Kristensen, P. G. Thomsen, and H. Madsen (2007). *New extended Kalman filter algorithms for stochastic differential algebraic equations*, pp. 359–366. Springer Berlin Heidelberg.

Jørgensen, J. B., H. Madsen, P. G. Thomsen, and M. R. Kristensen (2007). A computationally efficient and robust implementation of the continuous-discrete extended Kalman filter. In *proceedings of the 2007 Americal Control Conference (ACC)*, New York, NY, USA. July 11-13, pp. 3706–3712.

Julier, S. J. and J. K. Uhlmann (2004). Unscented filtering and nonlinear estimation. *Proceedings of the IEEE 92*(3), 401–422.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering 82*(1), 35–45.

Kandepu, R., B. Foss, and L. Imsland (2008). Applying the unscented Kalman filter for nonlinear state estimation. *Journal of Process Control 18*(7-8), 753–768.

Myrseth, I. and H. Omre (2010). Hierarchical ensemble Kalman filter. *SPE Journal 15*(2), 569–580.

Rawlings, J. B. and B. R. Bakshi (2006). Particle filtering and moving horizon estimation. *Computers & Chemical Engineering 30*(10-12), 1529–1541.

Rawlings, J. B., D. Q. Mayne, and M. Diehl (2017). *Model predictive control: theory, computation, and design* (2 ed.). Nob Hill Publishing Madison, WI.

Roth, M., C. Fritsche, G. Hendeby, and F. Gustafison (2015). The ensemble Kalman filter and its relations to other nonlinear filters. In *proceedings of the 23rd European Signal Processing Conference (EUSIPCO)*, Nice, France. August 31-September 4, pp. 1236–1240.

Tulsyan, A., R. B. Gopaluni, and S. R. Khare (2016). Particle filtering without tears: a primer for beginners. *Computers & Chemical Engineering 95*, 130–145.

Wan, E. A. and R. van der Merwe (2000). The unscented Kalman filter for nonlinear estimation. In *proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (AS-SPCC)*, Lake Louise, A.B., Canada. October 4, pp. 153–158.

APPENDIX C

Paper III - CDC 2021

A high-performance monte carlo simulation toolbox for uncertainty quantification of closed-loop systems

Authors:
Morten Ryberg Wahlgreen, Absjørn Thode Reenberg, Marcus Krogh Nielsen, Anton Rydahl, Tobias K. S. Ritschel, Bernd Dammann, John Bagterp Jørgensen

Published in:
proceedings of the 2021 60th IEEE conference on decision and control (CDC). December 13-15, 2021. Austin, Texas, USA.
DOI: `https://doi.org/10.1109/CDC45484.2021.9682781`

2021 60th IEEE Conference on Decision and Control (CDC)
December 13-15, 2021. Austin, Texas

A High-Performance Monte Carlo Simulation Toolbox for Uncertainty Quantification of Closed-loop Systems

Morten Ryberg Wahlgreen, Asbjørn Thode Reenberg, Marcus Krogh Nielsen, Anton Rydahl, Tobias K. S. Ritschel, Bernd Dammann, John Bagterp Jørgensen

2021 60th IEEE Conference on Decision and Control (CDC) | 978-1-6654-3659-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/CDC45484.2021.9682781

Abstract— **We apply Monte Carlo simulation for performance quantification and tuning of controllers in nonlinear closed-loop systems. Computational feasibility of large-scale Monte Carlo simulation is achieved by implementation of a parallelized high-performance Monte Carlo simulation toolbox for closed-loop systems in C for shared memory architectures. The toolbox shows almost linear scale-up on 16 CPU cores on a single NUMA node, and a scale-up of 27.3 on two NUMA nodes with a total of 32 CPU cores. We demonstrate performance quantification and tuning of a PID controller for a bioreactor in fed-batch operation. We perform 30,000 closed-loop simulations of the fed-batch reactor within 1 second. This is approximately a 2300 times computational performance increase compared to a serial reference implementation in Matlab. Additionally, we apply Monte Carlo simulation to perform automatic tuning of the PID controller based on maximizing average produced biomass within 8 seconds.**

I. Introduction

In closed-loop systems, we encounter unknown quantities that need to be estimated, e.g., model parameters. Additionally, it can be beneficial to quantify controller performance. Currently, there exist well-defined methods for parameters estimation [1], [2] and tuning of controllers in linear systems [3]–[5]. However, for nonlinear systems, quantification of controller performance and tuning is not as well developed. We propose a Monte Carlo simulation brute-force technique for automatic performance quantification and tuning of controllers in linear and nonlinear systems. With the Monte Carlo approach, we can tune controllers with any performance measure, e.g., maximizing economic yield or minimizing the risk of low production, such as in modern control applications [6], [7]. The Monte Carlo simulation technique is made computationally feasible by implementation of a high-performance Monte Carlo simulation toolbox parallelized for shared memory architectures in C.

Monte Carlo simulation is a widely used technique for quantification of uncertainties. It is applied in various areas, e.g., portfolio management and epidemiology [8]–[10]. Monte Carlo simulation uses random sampling to obtain numerical results about deterministic quantities. However, the method requires many samples to be effective, and thus computational efficiency becomes a bottleneck. The development in central processing unit (CPU) technology increases the number of possible applications for Monte Carlo simulation. However, trends in CPU development show that the clock frequency of new CPUs is no longer increasing due to power consumption and heat issues [11]. Instead, new CPUs have increased performance by increasing the number of cores. Consequently, the full potential of modern CPUs is only achieved with parallelized software executed on multi-core processors. Not all problems are parallelizable, but Monte Carlo simulation is a prime example of a parallelizable problem, as each simulation is independent of all other simulations. To achieve the full potential of Monte Carlo simulation on modern CPUs, we require state-of-the-art parallelized software in high-performance languages.

M. R Wahlgreen, A. T. Reenberg, M. K. Nielsen, A. Rydahl, T. K. S. Ritschel, B. Dammann, and J. B. Jørgensen are with the Department of Applied Mathematics and Computer Science, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark.
Corresponding author: J. B. Jørgensen (E-mail: jbjo@dtu.dk).

In this paper, we present closed-loop systems based on a stochastic continuous-discrete model, a stochastic differential equation (SDE) solver, and a controller. We introduce our implementation of a high-performance Monte Carlo simulation toolbox for closed-loop systems. Additionally, we introduce the SDE solvers and controllers contained in the toolbox. Furthermore, we demonstrate applications of the toolbox on a bioreactor in fed-batch operation [12]. In particular, we demonstrate that Monte Carlo simulation can be used for performance quantification and tuning of a proportional–integral–derivative (PID) controller.

The remaining part of the paper is organized as follows. Section II introduces the stochastic continuous-discrete system, two SDE solvers, and four controllers of increasing complexity. Section III presents the Monte Carlo simulation scheme for closed-loop systems and introduces our toolbox for Monte Carlo simulation. Section IV presents an example application of the toolbox. Section V presents our conclusion.

II. Closed-loop simulations

This section presents our representation of closed-loop systems for simulation. Our simulations consist of 1) an SDE model with discrete measurements, represented as a stochastic continuous-discrete model, 2) an SDE solver, and 3) a controller.

A. *Stochastic continuous-discrete system*

We consider stochastic continuous-discrete systems in the form

$$x(t_0) = x_0, \tag{1a}$$

$$\begin{aligned} dx(t) = {} & f(t, x(t), u(t), d(t), p_f)dt \\ & + \sigma(t, x(t), u(t), d(t), p_\sigma)d\omega(t), \end{aligned} \tag{1b}$$

$$y(t_k) = g(t_k, x(t_k), p_g) + v(t_k, p_v), \tag{1c}$$

$$z(t) = h(t, x(t), p_h), \tag{1d}$$

where $x(t)$ are the states, $u(t)$ are inputs, $d(t)$ are disturbances, and p_f, p_σ, p_g, p_v, and p_h are parameters. Additionally, x_0 is a normally distributed initial condition, $\omega(t)$ is a standard Wiener process, and $v(t_k, p_v)$ is normally distributed measurement noise at discrete time, i.e.,

$$x_0 \sim N(\bar{x}_0, P_0), \tag{2a}$$
$$d\omega(t) \sim N_{iid}(0, Idt), \tag{2b}$$
$$v(t_k, p_v) \sim N_{iid}(0, R(t_k, p_v)). \tag{2c}$$

Measurements are sampled with sampling time Δt, such that, $t_{k+1} = t_k + \Delta t$. We use zero-order-hold parameterization of the inputs and disturbances:

$$u(t) = u_k, \qquad t_k \le t < t_{k+1}, \tag{3a}$$
$$d(t) = d_k, \qquad t_k \le t < t_{k+1}. \tag{3b}$$

B. Stochastic differential equation solvers

SDE solvers are required to simulate stochastic continuous-discrete systems in the form (1). We consider an explicit-explicit (Euler-Maruyama) solver and an implicit-explicit solver [13], [14].

1) Explicit-explicit (Euler-Maruyama):

$$t_{k,n+1} = t_{k,n} + \Delta t, \tag{4a}$$
$$\begin{aligned} x_{k,n+1} = x_{k,n} &+ f(t_{k,n}, x_{k,n}, u_k, d_k, p_f)\Delta t \\ &+ \sigma(t_{k,n}, x_{k,n}, u_k, d_k, p_\sigma)\Delta\omega_{k,n}, \end{aligned} \tag{4b}$$

2) Implicit-explicit:

$$t_{k,n+1} = t_{k,n} + \Delta t, \tag{5a}$$
$$\begin{aligned} x_{k,n+1} = x_{k,n} &+ f(t_{k,n+1}, x_{k,n+1}, u_k, d_k, p_f)\Delta t \\ &+ \sigma(t_{k,n}, x_{k,n}, u_k, d_k, p_\sigma)\Delta\omega_{k,n}, \end{aligned} \tag{5b}$$

where $t_{k,0} = t_k$, $x_{k,0} = x_k$, and $\Delta w_{k,n} \sim N_{iid}(0, I\Delta t)$.

Let N_k denote the number of steps of size Δt in the interval $[t_k, t_{k+1}]$. Then

$$t_{k+1} = t_{k,N_k}, \tag{6a}$$
$$x_{k+1} = x_{k,N_k}. \tag{6b}$$

The explicit-explicit solver is suitable for non-stiff systems, whereas the implicit-explicit solver is suitable for stiff systems.

We let Φ represent the discretization of the state equation, (1b), with either the explicit-explicit solver or the implicit-explicit solver. To compactly describe simulation of closed-loop systems, we introduce the notation for a discretized version of (1),

$$x_{k+1} = \Phi(t_k, x_k, u_k, d_k, w_k, p_f, p_\sigma), \tag{7a}$$
$$y_k = g(t_k, x_k, p_g) + v_k, \tag{7b}$$
$$z_k = h(t_k, x_k, p_h), \tag{7c}$$

where $v_k = v(t_k, p_v)$.

C. Controller

The digital discrete-time controller in typical model-based control applications is represented as the dynamic system

$$x_k^c = \kappa(t_{k-1}, x_{k-1}^c, y_k, u_{k-1}, p_\kappa), \tag{8a}$$
$$u_k = \lambda(t_k, x_k^c, p_\lambda), \tag{8b}$$
$$z_k^c = \mu(t_k, x_k^c, p_\mu), \tag{8c}$$

where x_k^c are estimated states, z_k^c are predictions of the outputs and manipulated inputs, $\kappa(\cdot)$ is a state estimator, $\lambda(\cdot)$ is a regulator, and $\mu(\cdot)$ is a predictor.

We consider four controllers of increasing complexity.

1) Open-loop controller (no feedback): The open-loop controller does not include feedback and outputs a target value for the inputs

$$u_k = \lambda(\cdot) = \bar{u}_k. \tag{9}$$

The functions $\kappa(\cdot)$ and $\mu(\cdot)$ are not necessary for the open-loop controller.

2) Proportional–integral–derivative controller: The continuous PID controller is given by

$$u(t) = \bar{u}(t) + K_p e(t) + K_i \int_{t_0}^{t} e(\tau)d\tau + K_d \frac{de(t)}{dt}, \tag{10}$$

where K_p, K_i, and K_d are gain constants. The error, $e(t)$, is the difference between the set point, $\bar{y}(t)$, and the measurement, $y(t)$, i.e.,

$$e(t) = \bar{y}(t) - y(t). \tag{11a}$$

Notice, for the PID controller the output is assumed to be measured, i.e., $\bar{y}(t) = \bar{z}(t)$. It can be advantageous to let the derivative term act on the measurements, $y(t)$, rather than the error, $e(t)$ [15], [16]. We get,

$$u(t) = \bar{u}(t) + K_p e(t) + K_i \int_{t_0}^{t} e(\tau)d\tau - K_d \frac{dy(t)}{dt}. \tag{12}$$

We discretize the PID controller (12) as

$$e_k = \bar{y}_k - y_k^F, \tag{13a}$$
$$P_k = K_p e_k, \tag{13b}$$
$$I_k = I_{k-1} + T_s K_i e_k, \tag{13c}$$
$$D_k = -\frac{K_d}{T_s}(y_k^F - y_{k-1}^F), \tag{13d}$$
$$u_k = \bar{u}_k + P_k + I_k + D_k, \tag{13e}$$

where T_s is the sampling time and filtered measurements, y_k^F, are computed from the discrete-time low-pass filter

$$y_k^F = (1-\alpha)y_{k-1}^F + \alpha y_k, \tag{14}$$

with $\alpha \in [0, 1]$.

For the PID controller, $\lambda(\cdot)$ is given by (13e), $\kappa(\cdot)$ is given by (14), and $\mu(\cdot)$ is not necessary.

3) PID controller with clipping: We incorporate input bounds with clipping. The PID controller with clipping is,

$$\tilde{u}_k = \bar{u}_k + P_k + I_k + D_k, \tag{15a}$$
$$u_k = \max(u_{\min}, \min(u_{\max}, \tilde{u}_k)). \tag{15b}$$

4) Nonlinear model predictive control: The nonlinear model predictive controller (NMPC) includes a continuous-discrete extended Kalman filter (CD-EKF) based on (1) for state estimation and prediction [17], [18]. Given the filtered state-covariance pair, $\hat{x}_{k-1|k-1}$ and $P_{k-1|k-1}$, the CD-EKF obtains a one-step prediction

$$\hat{x}_{k|k-1} = \hat{x}_{k-1}(t_k), \tag{16a}$$
$$P_{k|k-1} = P_{k-1}(t_k), \tag{16b}$$

as the solution to

$$\frac{d}{dt}\hat{x}_{k-1}(t) = f(t, \hat{x}_{k-1}(t), u_{k-1}, d_{k-1}, p_f), \tag{17a}$$
$$\begin{aligned}\frac{d}{dt}P_{k-1}(t) &= A_{k-1}(t)P_{k-1}(t) + P_{k-1}(t)A_{k-1}(t)' \\ &\quad + \sigma_{k-1}(t)\sigma_{k-1}(t)',\end{aligned} \tag{17b}$$

for $t_{k-1} \leq t \leq t_k$ with initial condition

$$\hat{x}_{k-1}(t_{k-1}) = \hat{x}_{k-1|k-1}, \tag{18a}$$
$$P_{k-1}(t_{k-1}) = P_{k-1|k-1}, \tag{18b}$$

and

$$A_{k-1}(t) = \frac{\partial}{\partial x} f(t, \hat{x}_{k-1}(t), u_{k-1}, d_{k-1}, p_f), \tag{19a}$$
$$\sigma_{k-1}(t) = \sigma(t, \hat{x}_{k-1}(t), u_{k-1}, d_{k-1}, p_\sigma). \tag{19b}$$

The CD-EKF obtains a filtered state estimate, $\hat{x}_{k|k}$, and its covariance, $P_{k|k}$, from the one-step prediction, $\hat{x}_{k|k-1}$ and $P_{k|k-1}$, and the measurement, y_k. The CD-EKF computes the predicted measurement and derivative,

$$\hat{y}_{k|k-1} = g(t_k, \hat{x}_{k|k-1}, p_g), \tag{20a}$$
$$C_k = \frac{\partial}{\partial x} g(t_k, \hat{x}_{k|k-1}, p_g), \tag{20b}$$

the innovation and its covariance,

$$e_k = y_k - \hat{y}_{k|k-1}, \tag{21a}$$
$$R_{e,k} = C_k P_{k|k-1} C_k' + R_k, \tag{21b}$$

and the Kalman gain,

$$K_{fx,k} = P_{k|k-1} C_k' R_{e,k}^{-1}. \tag{22}$$

We obtain the estimated state-covariance pair from (20)-(22) as

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_{fx,k} e_k, \tag{23a}$$
$$P_{k|k} = P_{k|k-1} - K_{fx,k} R_{e,k} K_{fx,k}'. \tag{23b}$$

The CD-EKF is the state estimator, $\kappa(\cdot)$, that computes filtered state estimates, $x_k^c = \hat{x}_{k|k}$, from measurements y_k, and one-step state prediction, $\hat{x}_{k|k-1}$.

The NMPC uses a regulator based on a weighted least-squares objective and regularization of the input rate-of-movement. This regulator can be expressed in terms of the optimal control problem (OCP)

$$\begin{aligned}
\min_{x,u} \quad & \varphi_k = \varphi_{z,k} + \varphi_{\Delta u,k}, && \text{(24a)}\\
s.t. \quad & x(t_k) = \hat{x}_{k|k}, && \text{(24b)}\\
& \dot{x}(t) = f(t, x, u, d, p_f), \; t_k \leq t \leq t_k + T, && \text{(24c)}\\
& z(t) = h(t, x, p_h), && \text{(24d)}\\
& u(t) = u_{k+j}, \quad j \in \mathcal{N}, \; t_{k+j} \leq t \leq t_{k+j+1}, && \text{(24e)}\\
& d(t) = d_{k+j}, \quad j \in \mathcal{N}, \; t_{k+j} \leq t \leq t_{k+j+1}, && \text{(24f)}\\
& u_l \leq u_{k+j} \leq u_u, \qquad j \in \mathcal{N}, && \text{(24g)}\\
& \Delta u_l \leq \Delta u_{k+j} \leq \Delta u_u, \; j \in \mathcal{N}, && \text{(24h)}
\end{aligned}$$

with $f(t, x, u, d, p_f) = f(t, x(t), u(t), d(t), p_f)$ and the objective terms

$$\varphi_{z,k} = \frac{1}{2}\int_{t_k}^{t_k+T} \left\| W_z \left(z(t) - \bar{z}(t) \right) \right\|_2^2 dt, \tag{25a}$$
$$\varphi_{\Delta u,k} = \frac{1}{2}\sum_{j=0}^{N-1} \left\| \bar{W}_{\Delta u} \Delta u_{k+j} \right\|_2^2, \tag{25b}$$

where $\bar{W}_{\Delta u} = W_{\Delta u}/T_s$. The term $\varphi_{z,k}$ is output target tracking and $\varphi_{\Delta u,k}$ is input rate of movement penalty. We use the prediction and control horizon, T, defined as $T = NT_s$, where T_s is the sampling time and N is the discrete prediction and control horizon. Additionally, we define $\mathcal{N} = \{0, 1, ..., N-1\}$ such that $t_{k+j} = t_k + jT_s$ for $j \in \mathcal{N}$. We solve the OCP with a simultaneous approach, where we discretize each control interval with M time steps using Euler's implicit method.

We denote the optimal solution $\left\{\hat{x}_{k+j+1|k}, \hat{u}_{k+j|k}\right\}_{j\in\mathcal{N}}$. The input corresponding to the first control interval, $u_k = \hat{u}_{k|k} = \lambda(\cdot)$, is part of the solution of this optimal control problem. Only u_k is implemented in the system. Furthermore, $\{\hat{z}_{k+j+1|k}, \hat{u}_{k+j|k}\}_{j=0}^{N-1} = z_k^c = \mu(\cdot)$ is the predicted output and the predicted manipulated inputs from the controller that can be used for visualization.

D. Simulation of closed-loop systems

We compactly write a closed-loop simulation as

$$y_k = g(t_k, x_k, p_g) + v_k, \tag{26a}$$
$$z_k = h(t_k, x_k, p_h), \tag{26b}$$
$$x_k^c = \kappa(t_{k-1}, x_{k-1}^c, y_k, u_{k-1}, p_\kappa), \tag{26c}$$
$$u_k = \lambda(t_k, x_k^c, p_\lambda), \tag{26d}$$
$$z_k^c = \mu(t_k, x_k^c, p_\mu), \tag{26e}$$
$$x_{k+1} = \Phi(t_k, x_k, u_k, d_k, w_k, p_f, p_\sigma), \tag{26f}$$

for $k = 0, 1, ..., N_s - 1$.

III. Monte Carlo simulation

Our Monte Carlo simulations are based on the closed-loop simulation (26). We perform N_{mc} distinct closed-loop simulations for different, e.g., process noise realizations. Algorithm 1 presents an overview of the Monte Carlo simulation scheme. For sufficiently large N_{mc}, the computed Monte

Algorithm 1: Monte Carlo simulation

```
Result: Statistics and output data
// Monte Carlo loop
for $i = 1, 2, \dots, N_{mc}$ do
    // Closed loop simulation
    for $k = 0, 1, \dots, N_s - 1$ do
        // Measurement
        $y_k^{(i)} = g(t_k, x_k^{(i)}, p_g^{(i)}) + v_k^{(i)}$
        // Output
        $z_k^{(i)} = h(t_k, x_k^{(i)}, p_h^{(i)})$
        // State estimation
        $x_k^{c,(i)} = \kappa(t_{k-1}, x_{k-1}^{c,(i)}, y_k^{(i)}, u_{k-1}^{(i)}, p_\kappa^{(i)})$
        // Regulator
        $u_k^{(i)} = \lambda(t_k, x_k^{c,(i)}, p_\lambda^{(i)})$
        // Output prediction
        $z_k^{c,(i)} = \mu(t_k, x_k^{c,(i)}, p_\mu^{(i)})$
        // Simulator
        $x_{k+1}^{(i)} = \Phi(t_k, x_k^{(i)}, u_k^{(i)}, d_k^{(i)}, w_k^{(i)}, p_f^{(i)}, p_\sigma^{(i)})$
    end
    // Final measurement and output
    $y_{N_s}^{(i)} = g(t_{N_s}, x_{N_s}^{(i)}, p_g^{(i)}) + v_{N_s}^{(i)}$
    $z_{N_s}^{(i)} = h(t_{N_s}, x_{N_s}^{(i)}, p_h^{(i)})$
end
```

Carlo data can quantify uncertainties in the closed-loop system. Possible applications are; estimation of unknown model parameters, tuning controllers, and testing performance of controllers on different noise realizations.

A. Toolbox

We implement a Monte Carlo simulation toolbox for closed-loop systems in C. The toolbox provides an interface for closed-loop Monte Carlo simulations that currently includes implementations of

- an explicit-explicit Euler-Maruyama SDE solver,
- an implicit-explicit SDE solver,
- an open-loop controller,
- a single-input single-output (SISO) PID controller with clipping, and
- an NMPC based on the CD-EKF and a simultaneous approach combined with IPOPT [19].

The toolbox includes three test examples, and the user can provide a set of model functions for a system and perform Monte Carlo simulations with the toolbox. Additionally, the toolbox allows for user-provided controllers and SDE solvers with specific interfaces. This allows the user to test and benchmark controllers and SDE solvers using Monte Carlo simulations. The toolbox supports perturbations of model parameters, controller parameters, noise realizations, initial conditions, and disturbances.

We include a parallelized version with OpenMP for shared memory architectures. Each worker is assigned distinct closed-loop simulations. Such parallelization requires that each worker has access to a local workspace for the SDE solver and the controller to avoid data races. Additionally, some controllers utilize information from previous steps, e.g., the integral term of a PID controller or the CD-EKF for an NMPC. Each worker also requires a local version of such information. The Monte Carlo simulation toolbox distributes memory blocks to each local worker, such that workers do not have overlapping cache lines. This consideration is essential for achieving optimal parallel performance.

We demonstrate some applications of the toolbox in section IV.

TABLE I
PARAMETERS FOR FED-BATCH REACTOR.

Variable	Value	Unit
$\mu_{\max}$	0.37	1/h
K_S	0.021	kg/m^3
K_I	0.38	kg/m^3
γ_s	1.777	kg substrate/kg biomass
$c_{S,in}$	10.0	kg/m^3

TABLE II
INITIAL CONDITION AND OPERATIONAL BOUNDS.

Variable	Value	Unit
V_0	1.00	m^3
$c_{X,0}$	2.00	kg/m^3
$c_{S,0}$	0.0893	kg/m^3
$V_{\max}$	12.39	m^3
$c_{X,\max}$	2.00	kg/m^3
$c_{S,\max}$	3.00	kg/m^3
$F_{S,\max}$	10.00	m^3
$F_{W,\max}$	10.00	m^3

IV. BIOREACTOR IN FED-BATCH OPERATION

A. Model

We consider the SDE model for a bioreactor in fed-batch operation [12],

$$dV = (F_S + F_W)dt + \sigma_1 d\omega_1(t), \tag{27a}$$
$$dm_X = (R_X V)dt + \sigma_2 d\omega_2(t), \tag{27b}$$
$$dm_S = (F_S c_{S,in} + R_S V)dt + \sigma_3 d\omega_3(t), \tag{27c}$$

where $m_X = c_X V$, $m_S = c_S V$, and

$$R_X = r, \qquad r = \mu(c_S)c_X, \tag{28a}$$
$$R_S = -\gamma r, \quad \mu(c_S) = \mu_{\max}\frac{c_S}{K_S + c_S + c_S^2/K_I}. \tag{28b}$$

We represent the system as a stochastic continuous-discrete model in the form (1), where $x(t) = [V(t); m_X(t); m_S(t)]$, $y(t_k) = c_S(t_k) + v(t_k)$, and $z(t) = c_S(t)$.

Table I presents the parameters of the system and Table II presents the initial conditions and operational bounds of the system.

B. Control strategy

We operate the bioreactor with an open-loop input trajectory, $\bar{u} = [\bar{F}_W; \bar{F}_S]$. Additionally, we use a SISO PID

TABLE III

SYSTEM INFORMATION.

Architecture:	x86_64
CPU op-mode(s):	32-bit, 64-bit
CPU(s):	32
Thread(s) per core:	1
Core(s) per socket:	16
Socket(s):	2
NUMA node(s):	2
Model name:	Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz
CPU MHz:	2900.000
L1d cache:	32 kB
L1i cache:	32 kB
L2 cache:	1024 kB
L3 cache:	22528 kB
RAM:	384 GB

controller with clipping, (15), that manipulate the substrate inlet, F_S, to achieve optimal substrate concentration, c_S^*, achieved at the maximum of $\mu(c_S)$,

$$c_S^* = \sqrt{K_I K_S}. \tag{29}$$

Thus, $\bar{z}(t) = c_S^*$. We use the bang-bang open-loop trajectory [12]. In the deterministic case, the bang-bang trajectory was one among infinitely many optimal solutions. However, it was the least sensitive to uncertainties. The inputs are computed as,

$$\bar{F}_W = \begin{cases} F_{W,\max}, & 0 \le t \le t_{switch}, \\ 0, & t_{switch} \le t \le t_f, \end{cases} \tag{30a}$$

$$\bar{F}_S = \frac{F_W c_S^* + \gamma_s \beta^*(t)}{c_{S,in} - c_S^*}, \tag{30b}$$

where

$$\beta^*(t) = \mu(c_S^*) c_{X,\max} V_0 \exp(\mu(c_S^*) t). \tag{31}$$

C. Simulation of the true system

We simulate the fed-batch reactor in closed-loop with an Euler-Maruyama solver. The reactor runs for 10 hours from time $t_0 = 0$ to $t_f = 10$ h. The sampling time is $T_s = 36$ seconds resulting in $N_s = 1000$ steps. At each step, we solve the SDE with $N_k = 10$ Euler-Maruyama steps.

D. Monte Carlo simulations of fed-batch reactor

Here, we demonstrate an application of the Monte Carlo simulation toolbox. The simulations are conducted on a dual-socket Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz system (see Table III for CPU details).

1) Scaling: Fig. 1 shows the wall time and the scale-up for 10,000 Monte Carlo simulations. We observe close to linear scaling within one non-uniform memory access (NUMA) node, and slightly decreasing scale-up when exceeding one socket. We point out that the toolbox is not optimized to utilize multiple NUMA nodes, so a decrease in performance on more than one socket is expected.

2) Open-loop controller: We perform Monte Carlo simulations for the fed-batch reactor in open-loop. Fig. 3(a) shows a probability density function (PDF) plot for 30,000 realizations of process noise. The mean produced biomass is $\bar{m}_X(t_f) = 20.69$ kg.

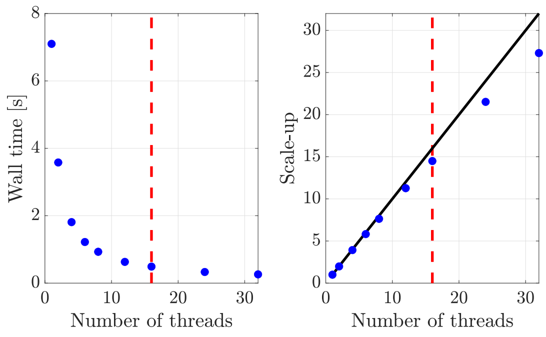


Fig. 1. Wall time and scale-up plots for 10,000 Monte Carlo simulations. The red dashed line is the number of cores on a single NUMA node. We get a scale-up of 27.3 on 32 cores.

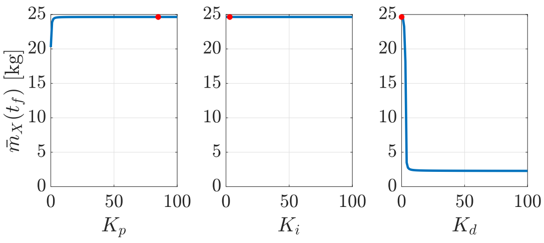
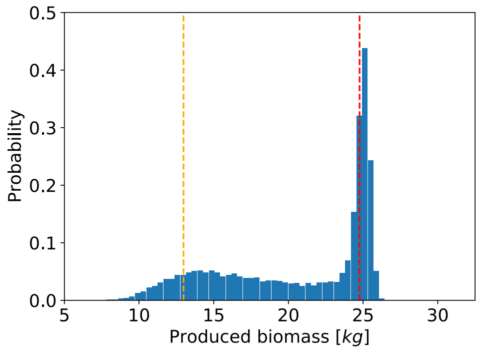


Fig. 2. Tuning of PID gains. Left: locate $K_p = 85$ as the optimum. Middle: locate $K_i = 3$ as the optimum. Right: locate $K_d = 0$ as the optimum. Total Monte Carlo simulations: $3 \cdot 101,000 = 303,000$. Computation time: ~ 7.50 seconds.

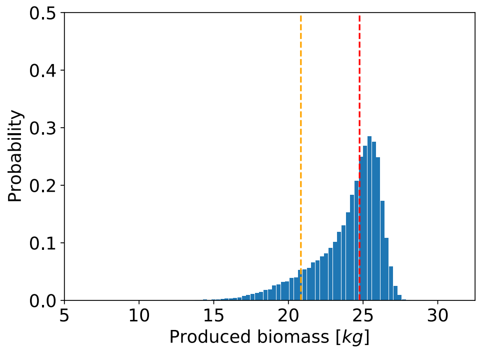
3) Quantification of PID controller performance: Consider a PID controller with $K_p = 1.0$, $K_i = 0.0$, and $K_d = 0.0$. We perform a Monte Carlo simulation with 30,000 process noise realizations. Fig. 3(b) presents a PDF plot of the produced biomass. The PDF follows a long-tailed distribution towards the lower values of produced biomass with mean produced biomass $\bar{m}_X(t_f) = 24.04$ kg, i.e., a 16.19% increase in biomass production compared to the open-loop controller. However, low produced biomass, for some realizations of process noise, indicates poor controller performance. The computation time of the Monte Carlo simulation is 0.77s. That is approximately a 2300 times speed-up compared to a reference serial implementation in Matlab.

4) Tuning: We apply Monte Carlo simulation to tune the value of K_p, K_i, and K_d in the PID controller. The tuning is based on maximizing the average produced biomass, $\bar{m}_X(t_f)$, for 1000 realizations of process noise. We point out that the tuning could have been based on other factors, e.g., maximizing the 10% quantile. We investigate 101 equidistant values in $[0, 100]$ of K_p, K_i, and K_d. Thus, the tuning of each gain requires 101,000 closed loop simulations. Fig. 2 shows the tuning results of the PID controller. The optimal parameters for the PID gains are $K_p = 85$, $K_i = 3$, and $K_d = 0$.

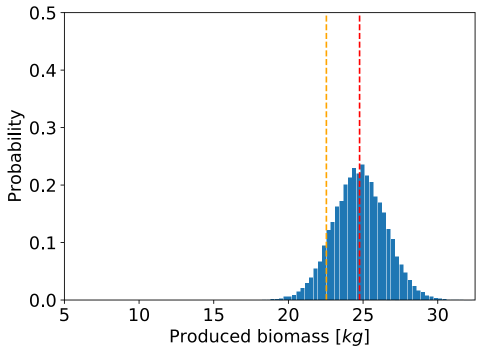
Fig. 3(c) shows a PDF plot for 30,000 noise realizations



(a) Open-loop controller. Computation time: 0.78s.



(b) Sub-optimal PID controller. Computation time: 0.77s.



(c) Optimal PID controller. Computation time: 0.76s.

Fig. 3. Probability density function of biomass production computed from $30,000$ closed-loop simulations with different process noise realizations. The closed-loop consists of the fed-batch model, the Euler-Maruyama SDE solver, and a controller specified in the subplot. The red dashed line is the produced biomass in a simulation without process noise and the orange dashed line is the 10% quantile.

with the tuned PID controller. The PDF is almost normally distributed with mean produced biomass, $\bar{m}_X(t_f) = 24.76$. Compared to the non-optimal PID controller, the tuned controller results in an 2.98% increased average biomass production and reduced risk of low biomass production. It is evident that the tuning improved the performance of the PID controller.

5) NMPC: Initial investigation of an NMPC based on the open source optimization software, IPOPT, does not show the same scaling as the PID controller. We observe a significant stall time in the memory allocation with $malloc()$, when increasing the number of threads. These calls to $malloc()$ are located in IPOPT and give reason to believe that IPOPT has internal memory allocation. Each memory allocation has a lock that interrupts all activity, i.e., stalling all threads. In future work, we will expand the toolbox to include an NMPC based on optimization software that does not have internal memory allocation. We believe that scaling similar to the PID case can be achieved with such an NMPC. Fig. 4 presents a PDF plot for 1000 process noise realizations. The NMPC and the tuned PID controller show similar performance. The experiment is conducted on 6 cores as performance decreases above 6 cores due to the problem mentioned above. The computation time is ~ 30 min.

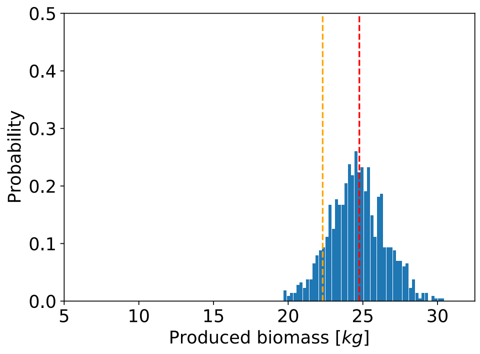


Fig. 4. Probability density function of biomass production computed from 1000 NMPC closed-loop simulations with different process noise realizations. The red dashed line is the produced biomass in a simulation without process noise and the orange dashed line is the 10% quantile. The computation time is ~ 30 min.

V. Conclusion

The paper presents a Monte Carlo simulation approach for performance quantification and tuning of controllers in linear and nonlinear systems. The approach is computationally feasible due to the implementation of a parallel high-performance Monte Carlo simulation toolbox in C for closed-loop systems. In particular, we demonstrate performance quantification and tuning of a PID controller for a bioreactor in fed-batch operation. Our results show that large-scale Monte Carlo simulations can be performed within seconds. The computational performance of the toolbox show approximately a 2300 times speed-up compared to a serial reference implementation in Matlab.

High-performance closed-loop Monte-Carlo simulations, as illustrated in this paper, has countless applications in systems and control. Drug dosing, as in treatment of diabetes, is a very prominent example of this where several dosing strategies must be compared by their probability density functions [20], [21].

References

[1] D. Boiroux, T. K. S. Ritschel, N. K. Poulsen, H. Madsen, and J. B. Jørgensen, "Efficient Computation of the Continuous-Discrete Extended Kalman Filter Sensitivities Applied to Maximum Likelihood stimation," *58th Conference on Decision and Control*, pp. 6983–6988, 2019.

[2] H. G. Bock, E. Kostina, and J. P. Schlöder, "Numerical Methods for Parameter Estimation in Nonlinear Differential Algebraic Equations," *GAMM-Mitteilungen*, vol. 30, no. 2, pp. 376–408, 2007.

[3] S. Skogestad, "Simple analytic rules for model reduction and PID controller tuning," *Journal of Process Control*, p. 291–309, 2003.

[4] D. Olesen, J. K. Huusom, and J. B. Jørgensen, "A Tuning Procedure for ARX-based MPC," *IEEE Multi-conference on Systems and Control*, pp. 188–193, 2013.

[5] D. H. Olesen, J. K. Huusom, and J. B. Jørgensen, "A tuning procedure for ARX-based MPC of multivariate processes," *American Control Conference*, pp. 1721–1726, 2013.

[6] T. K. S. Ritschel, D. Boiroux, M. K. Nielsen, J. K. Huusom, S. B. Jørgensen, and J. B. Jørgensen, "Economic Optimal Control of a U-loop Bioreactor using Simultaneous Collocation-based Approaches," in *2019 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 2019, pp. 933–938.

[7] ——, "Economic Nonlinear Model Predictive Control of a U-loop Bioreactor," in *2020 European Control Conference (ECC)*. IEEE, 2020, pp. 208–213.

[8] L. J. Hong, S. Juneja, and J. Luo, "Estimating sensitivities of portfolio credit risk using Monte Carlo," *Informs Journal on Computing*, vol. 26, no. 4, pp. 848–865, 2014.

[9] A. Shlyakhter, L. Mirny, A. Vlasov, and R. Wilson, "Monte Carlo Modeling of Epidemiologic Studies," *Human and Ecological Risk Assessment*, vol. 2, no. 4, pp. 920–936, 1996.

[20] J. B. Jørgensen, D. Boiroux, and Z. Mahmoudi, "An artificial pancreas based on simple control algorithms and physiological insight," *12th Symposium IFAC on Dynamics and Control of Process Systems, including Biosystems (DYCOPS 2019), Florianópolis - SC, Brazil*, April 23-26 2019.

[10] S. J. Ratick and G. Schwarz, "Monte Carlo Simulation," in *Encyclopedia of Human Geography (Second Edition)*. Elsevier, 2009, ch. 14, pp. 175–185.

[11] P. Ross, "Why CPU Frequency Stalled," *IEEE Spectrum*, vol. 45, no. 4, p. 72, 2008.

[12] T. E. Ryde, M. R. Wahlgreen, M. K. Nielsen, S. Hørsholt, S. B. Jørgensen, and J. B. Jørgensen, "Optimal Feed Trajectories for Fed-batch Fermentation with Substrate Inhibition Kinetics," *International Symposium on Advanced Control of Chemical Processes, Accepted*, 2021.

[13] D. J. Higham, "An Algorithmic Introduction to Numerical Simulation of Stochastic Differential Equations," *SIAM Review*, vol. 43, no. 3, pp. 525–546, 2001.

[14] T. Tian and K. Burrage, "Implicit Taylor methods for stiff stochastic differential equations," *Applied Numerical Mathematics*, vol. 38, pp. 167–185, 2001.

[15] B. Wittenmark, K. Johan Åström, and K.-E. Årzén, in *Computer Control: An Overview*. IFAC professional brief, 2009.

[16] K. J. Åström and R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2008.

[17] M. R. Wahlgreen, E. Schroll-Fleischer, D. Boiroux, T. K. S. Ritschel, H. Wu, J. K. Huusom, and J. B. Jørgensen, "Nonlinear Model Predictive Control for an Exothermic Reaction in an adiabatic CSTR," *6th Conference on Advances in Control and Optimization of Dynamical Systems ACODS, Chennai, India*, February 16-19 2020.

[18] J. B. Jørgensen, T. K. S. Ritschel, D. Boiroux, E. Schroll-Fleischer, M. R. Wahlgreen, M. K. Nielsen, H. Wu, and J. K. Huusom, "Simulation of NMPC for a Laboratory Adiabatic CSTR with an Exothermic Reaction," *Proceedings of 2020 European Control Conference*, pp. 202–207, 2020.

[19] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.

[21] D. Boiroux and J. Jørgensen, "Nonlinear Model Predictive Control and Artificial Pancreas Technologies," *IEEE Conference on Decision and Control (CDC), Miami Beach, Florida, USA*, December 17-19 2018.

APPENDIX D
Paper IV - CCTA 2019 A

Economic optimal control of a U-loop bioreactor using simultaneous collocation-based approaches

Authors:
Tobias K. S. Ritschel, Dimitri Boiroux, Marcus Krogh Nielsen, Jakob Kjøbsted Huusom, Sten Bay Jørgensen, John Bagterp Jørgensen

Published in:
proceedings of the 2019 IEEE conference on control technology and applications (CCTA). August 19-21, 2019. Hong Kong, China.
DOI: `https://doi.org/10.1016/10.1109/CCTA.2019.8920479`

Economic Optimal Control of a U-loop Bioreactor using Simultaneous Collocation-based Approaches

Tobias K. S. Ritschel, Dimitri Boiroux, Marcus Krogh Nielsen, Jakob Kjøbsted Huusom, Sten Bay Jørgensen, John Bagterp Jørgensen

Abstract— **In this paper, we consider economic optimal control of single-cell protein (SCP) production in a U-loop reactor. The model of the U-loop reactor contains both ordinary and partial differential equations. Consequently, the optimal control problems are large-scale. The optimal operating profile for the SCP production is an unstable attractor. Therefore, we consider two simultaneous collocation-based approaches for solving the optimal control problems. We implement these two approaches in C, and we use IPOPT to solve the involved nonlinear program (NLP). Finally, we present a performance study that demonstrates the feasibility of solving economic optimal control problems that involve the U-loop reactor in real-time.**

I. Introduction

The objective of optimal control is to compute an open-loop control strategy that optimizes a performance measure which either represents 1) the economics of the dynamic process or 2) the deviation of the process outputs from pre-defined setpoints. The solution of optimal control problems is relevant to model predictive control algorithms which use the moving horizon optimization principle to compute a closed-loop feedback control strategy, i.e. they solve a sequence of open-loop optimal control problems [1]. Such applications involve strict computational requirements. It is particularly challenging to meet such requirements if the process involves a large number of state variables. This is often the case when the model arises from the discretization of partial differential equations, e.g. oil reservoir fluid flow in porous media [2], [3], or from processes that involve several interconnected units, e.g. distillation columns that consist of several trays [4]. Furthermore, the solution of optimal control problems can also be used for offline analysis of process control strategies for transient operating conditions, e.g. plant startup. The computational requirements are less strict in such applications.

A. Numerical solution of optimal control problems

There exist several approaches for numerical solution of optimal control problems. Direct methods, i.e. single-shooting, multiple-shooting [5], [6], and collocation-based approaches [4], are commonly used for real-life applications [1]. These methods transcribe the infinite-dimensional optimal control problem to a finite-dimensional nonlinear program (NLP) that can be solved using numerical optimization algorithms [7]. In single-shooting, the solution of the dynamical constraints is nested into the solution of the NLP which leads to a small-scale NLP. In collocation-based approaches, the discretized dynamical constraints are incorporated directly into the NLP leading to a large-scale problem. Multiple-shooting is a hybrid approach that attempts to combine the advantages of both single-shooting and collocation-based approaches. Two key advantages of multiple-shooting and collocation-based approaches over single-shooting are 1) that they are applicable to unstable systems and 2) that it is more straightforward to implement path constraints for these approaches. However, both of these approaches lead to large-scale NLPs.

B. Production of single-cell protein in a U-loop reactor

Methanotrophs are bacteria that grow on carbon sources such as methane or methanol which are cheap. The protein content of methanotrophs is high, and they can be used to produce single-cell protein (SCP) which can be used for animal feed. Consequently, SCP produced from methanotrophs can be used to sustain the growing human population. The U-loop reactor is a novel technology for producing SCP based on methanotrophs. However, it is nontrivial to operate the U-loop reactor, and in particular, the startup is challenging.

The dynamics of SCP production in a U-loop reactor have previously been modeled as a set of partial and ordinary differential equations [8]–[12]. Using this model, Olsen et al. [13] computed optimal operating points for steady-state operation. However, they did not consider the startup. In this work, we extend previous work on economic optimal control of the U-loop reactor [14] that involved the computation of economically optimal startup profiles. The key contributions of this work are that 1) we describe the numerical details of two collocation-based optimal control algorithms, 2) we provide details on computationally efficient C implementations of these algorithms which are based on the open-source software IPOPT 3.12.12, and 3) we present a performance study that demonstrates the feasibility of using the algorithms in closed-loop nonlinear model predictive control. A key aspect of SCP production in the U-loop reactor is that the optimal operating profile turns out to be an unstable attractor [14] (the system will diverge from the optimal trajectory if it is not controlled). This is the reason that we use collocation-

*This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 723661, Spire project: "Coordinating Optimisation of Complex Industrial Processes" (www.cocop-spire.eu). This work also received funding in the project EUDP 64013-0558 in the IEA annex for energy efficient process control. Tobias K. S. Ritschel, Dimitri Boiroux, Marcus Krogh Nielsen, and John Bagterp Jørgensen are with the Department of Applied Mathematics and Computer Science, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark. Jakob Kjøbsted Huusom and Sten Bay Jørgensen are with the Department of Chemical and Biochemical Engineering, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark. Tobias K. S. Ritschel is also with 2-control ApS, DK-2900 Hellerup, Denmark. `jbjo@dtu.dk`

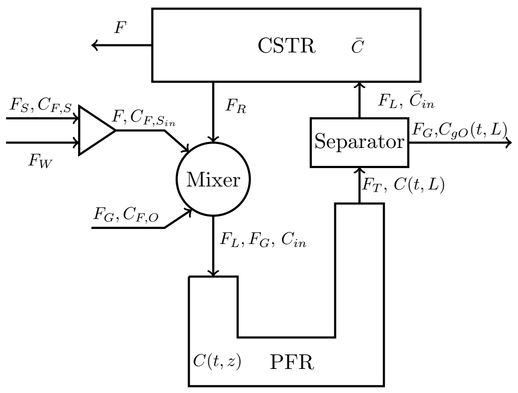


Fig. 1. A schematic of SCP production in the U-loop reactor.

based approaches. It would also be possible to use multiple-shooting.

C. *Paper organization*

This paper is organized as follows. In Section II, we present the model of SCP production in the U-loop reactor (using methanol as the feed). In Section III, we describe the optimal control problem that we consider as well as the two collocation-based approaches. In Section IV, we discuss the implementation of these two approaches, and in Section V, we present two numerical examples together with the performance study. Finally, we conclude this work in Section VI.

II. MATHEMATICAL MODEL

Fig. 1 shows a schematic of SCP production in the U-loop reactor [8], [9], [13], [14]. The feed substrate (methanol), feed water, feed gas (oxygen), and the recirculated mixture from the top tank are mixed together in a mixer and supplied to the inlet of the U-loop pipe. The U-loop pipe is modeled as a liquid-gas phase plug flow reactor (PFR), and the top tank is modeled as a liquid continuous stirred tank reactor (CSTR). At the end of the U-loop pipe, the gas phase is separated (ideally) from the liquid phase which enters into the top tank. Liquid (consisting of a mixture of water, biomass, substrate, and dissolved gas) is harvested from the top tank.

A. *Mixing section*

At the inlet of the PFR, the gas flow rate is equal to the feed gas flow rate, F_G, and the liquid flow rate, F_L, is obtained from a total static mass balance:

$$F_L = F_R + F_S + F_W. \tag{1}$$

F_R is the flow rate of recirculated liquid from the top tank through the mixer to the U-loop pipe, and F_S and F_W are the feed flow rates of substrate and water, respectively. The concentrations of biomass, X, substrate, S, dissolved oxygen in the liquid phase, O, and oxygen in the gas phase, gO, at the inlet of the PFR are obtained using component mass balances:

$$C_{in,X} = \frac{F_R \bar{C}_X}{F_L}, \tag{2a}$$

$$C_{in,S} = \frac{F_S C_{F,S} + F_R \bar{C}_S}{F_L}, \tag{2b}$$

$$C_{in,O} = \frac{F_R \bar{C}_O}{F_L}, \tag{2c}$$

$$C_{in,gO} = C_{F,O}. \tag{2d}$$

$\bar{C}_X$, $\bar{C}_S$, and $\bar{C}_O$ are the concentrations of biomass (X), substrate (S), and oxygen (O) in the top tank, respectively.

B. *The U-loop modeled as a plug-flow reactor*

The phase fluxes at the inlet to the PFR, N_i, are

$$N_i(t,0) = v C_{in,i}(t), \quad i \in \{X, S, O, gO\}. \tag{3}$$

The linear velocity, v, is given by

$$v = \frac{F_L + F_G}{A}, \tag{4}$$

where A is the cross-sectional area of the U-loop pipe. The concentrations, $C_i = C_i(t,z)$, of each of the components $i \in \{X, S, O, gO\}$ along the U-loop pipe are given by the mass balances

$$\frac{\partial C_X}{\partial t} = -\frac{\partial N_X}{\partial z} + R_X, \tag{5a}$$

$$\frac{\partial C_S}{\partial t} = -\frac{\partial N_S}{\partial z} + R_S, \tag{5b}$$

$$\frac{\partial C_O}{\partial t} = -\frac{\partial N_O}{\partial z} + R_O + \frac{1}{1-\varepsilon} J_{gl,O}, \tag{5c}$$

$$\frac{\partial C_{gO}}{\partial t} = -\frac{\partial N_{gO}}{\partial z} - \frac{1}{\varepsilon} J_{gl,O}, \tag{5d}$$

for the time interval $t_0 \leq t \leq t_f$ and for $0 \leq z \leq L$. t_0 and t_f are the initial and final time, and L denotes the length of the U-loop pipe. $N_i = N_i(C_i(t,z)) = N_i(t,z)$ for $i \in \{X, S, O, gO\}$ are the fluxes of each component along the U-loop pipe, and $R_i = R_i(t,z)$ for $i \in \{X, S, O\}$ are the production rates of the components in the liquid phase. $J_{gl,O} = J_{gl,O}(t,z)$ denotes the rate at which oxygen is transferred from the gas phase to the liquid phase, and $\varepsilon = F_G/(F_G + F_L)$ is the fraction of gas in each cross section which is assumed to be constant along the U-loop pipe.

The outlet boundary conditions of the PFR are described using Danckwerts' conditions:

$$\frac{\partial C_i}{\partial z}(t,L) = 0, \quad i \in \{X, S, O, gO\}, \quad t_0 \leq t \leq t_f. \tag{6}$$

C. *Gas-liquid separator*

The gas and liquid are assumed to be instantaneously and perfectly separated at the outlet of the U-loop pipe, i.e. the gas phase is completely removed and the liquid phase is supplied directly to the top tank. Consequently, the concentrations at the inlet to the top tank are

$$\bar{C}_{in,i}(t) = C_i(t,L), \quad i \in \{X, S, O\}, \quad t_0 \leq t \leq t_f. \tag{7}$$

D. Model of the top tank

The liquid volume of the top tank, V, is constant, and the component mass balances for the components in the liquid phase are

$$\frac{d\bar{C}_i}{dt} = \bar{D}\left(\bar{C}_{in,i} - \bar{C}_i\right) + R_i(\bar{C}_X, \bar{C}_S, \bar{C}_O),\ i \in \{X, S, O\}, \tag{8}$$

for $t_0 \le t \le t_f$. $\bar{D} = \bar{F}/V$ is the dilution rate where $\bar{F} = F_L$. The flow rate of the product stream out of the top tank is

$$F = \bar{F} - F_R = F_L - F_R = F_S + F_W. \tag{9}$$

E. Convective and diffusive transport

The flux, N_i, contains a term that describes convective transport, vC_i, and one that describes diffusive transport, J_i:

$$N_i = v_i C_i + J_i, \quad i \in \{X, S, O, gO\}. \tag{10}$$

Fick's law is used to model the diffusive transport:

$$J_i = -D_i \frac{\partial C_i}{\partial z}, \quad i \in \{X, S, O, gO\}. \tag{11}$$

F. Gas-liquid transport

The rate at which oxygen is transferred from the gas phase to the liquid phase is

$$J_{gl,O} = (k_L a)_O (C_{sat,O} - C_O), \tag{12}$$

where the saturation concentration of oxygen is obtained using Henry's law and the ideal gas law:

$$C_{sat,O} = \frac{P_{gO}}{H_O} = \frac{RT}{M_{wO} H_O} C_{gO}. \tag{13}$$

Here, P_{gO} denotes the partial pressure of oxygen in the gas phase, and H_O is the Henry constant for oxygen. R is the gas constant, T denotes the temperature, and M_{wO} is the molar weight of oxygen. C_{gO} is the concentration of oxygen in the gas phase.

G. Stoichiometry and kinetics

The stoichiometric relation for the conversion of methanol (S) to biomass (X) is

$$\begin{aligned} \mathrm{CH_3OH} + Y_{SN}\mathrm{HNO_3} + Y_{SO}\mathrm{O_2} \to \\ Y_{SX}X + Y_{SC}\mathrm{CO_2} + Y_{SW}\mathrm{H_2O}. \end{aligned} \tag{14}$$

Alternatively, this relation can be formulated as

$$\begin{aligned} Y_{XS}\mathrm{CH_3OH} + Y_{XN}\mathrm{HNO_3} + Y_{XO}\mathrm{O_2} \to \\ X + Y_{XC}\mathrm{CO_2} + Y_{XW}\mathrm{H_2O}. \end{aligned} \tag{15}$$

The yield coefficients (for both formulations of the stoichiometric relation) are shown in Table I. The production rates of substrate (methanol), R_S, and oxygen dissolved in the liquid phase, R_O, are given in terms of the production rate of biomass, R_X:

$$R_S = -\gamma_S R_X, \qquad \gamma_S = \frac{M_{wS}}{M_{wX} Y_{SX}}, \tag{16a}$$

$$R_O = -\gamma_O R_X, \qquad \gamma_O = \frac{M_{wO} Y_{SO}}{M_{wX} Y_{SX}}. \tag{16b}$$

TABLE I
YIELD COEFFICIENTS

	i	Y_{Si} [mol/mol]	Y_{Xi} [mol/mol]	M_{wi} [g/mol]	W_{Xi} [g/g]
$\mathrm{CH_3OH}$	S	1.000	1.366	32.042	1.778
$\mathrm{HNO_3}$	N	0.146	0.199	63.013	0.510
$\mathrm{O_2}$	O	0.439	0.600	31.999	0.779
$\mathrm{CH_{1.8}O_{0.5}N_{0.2}}$	X	0.732	1.000	24.626	1.000
$\mathrm{CO_2}$	C	0.268	0.366	44.010	0.654
$\mathrm{H_2O}$	W	1.415	1.933	18.015	1.414

The biomass production rate is

$$R_X = \mu(C_S, C_O) C_X, \tag{17}$$

where $\mu = \mu(C_S, C_O)$ is the specific growth rate which, for Methylococcus Capsulatus, is given by the Monod-Haldane expression:

$$\mu = \mu(C_S, C_O) = \mu_{\max} \mu_S(C_S) \mu_O(C_O). \tag{18a}$$

The growth factors are

$$\mu_S(C_S) = \frac{C_S}{K_S + C_S + C_S^2/K_I}, \tag{18b}$$

$$\mu_O(C_O) = \frac{C_O}{K_O + C_O}. \tag{18c}$$

H. Profit

In this work, we use the expression for the profit of the SCP production in the U-loop reactor described by Drejer et al. [14]. The profit is the difference between the value of the produced SCP (biomass) and the cost of the raw materials, and we assume that the two most expensive raw materials are the feed methanol (substrate) and oxygen, i.e. the cost of other raw materials is not included in the expression for the profit. Consequently, the profit is

$$\begin{aligned} \phi = &\int_{t_0}^{t_f} \left(p_X F(t) \bar{C}_X(t) - p_S F_S(t) - p_O F_G(t) \right) dt \\ &+ p_X \left(\bar{C}_X(t_f) V + \int_0^L C_X(t_f, z) A \, dz \right) \\ &- p_X \left(\bar{C}_X(t_0) V + \int_0^L C_X(t_0, z) A \, dz \right), \end{aligned} \tag{19}$$

where p_X is the unit value of SCP, p_S is the unit cost of methanol (substrate), and p_O is the unit cost of oxygen.

III. OPTIMAL CONTROL

We consider optimal control problems in the form

$$\min_{[x(t)]_{t_0}^{t_f}, \{u_k\}_{k=0}^{N-1}} \quad \phi = \phi([x(t); u(t); d(t)]_{t_0}^{t_f}), \tag{20a}$$

subject to

$$x(t_0) = \hat{x}_0, \tag{20b}$$

$$\dot{x}(t) = f(x(t), u(t), d(t)), \quad t \in [t_0, t_f], \tag{20c}$$

$$u(t) = u_k, \quad t \in [t_k, t_{k+1}[, \quad k = 0, \ldots, N-1, \tag{20d}$$

$$d(t) = \hat{d}_k, \quad t \in [t_k, t_{k+1}[, \quad k = 0, \ldots, N-1, \tag{20e}$$

where the objective function is in Bolza form (as is the expression for the profit (19)):

$$\begin{aligned}\phi &= \phi([x(t); u(t); d(t)]_{t_0}^{t_f}) \\ &= \int_{t_0}^{t_f} \Phi(x(t), u(t), d(t))dt + \hat{\Phi}(x(t_f), u(t_f), d(t_f)).\end{aligned} \tag{21}$$

Φ is the stage-cost, and $\hat{\Phi}$ is the cost-to-go. The optimal control problem (20) contains dependent decision variables (the state variables), $[x(t)]_{t_0}^{t_f}$, and independent decision variables (the manipulated inputs), $\{u_k\}_{k=0}^{N-1}$. (20b) is an initial condition for the differential equations (20c) which is obtained from the model of the U-loop reactor by discretizing the involved partial differential equations using a finite-volume discretization. (20d)-(20e) are zero-order-hold (ZOH) discretizations of the manipulated inputs, $u(t)$, and the disturbance variables, $d(t)$.

A. Temporal discretization

The dynamical system in (20c) is stiff. Therefore, we use an implicit method to temporally discretize the system. In this work, we use Euler's implicit method. In the k'th control interval, we discretize the dynamical system using N_k time steps, i.e.

$$x_{k,n+1} - x_{k,n} = f(x_{k,n+1}, u_k, \hat{d}_k)\Delta t_{k,n}, \tag{22}$$

for $n = 0, \dots, N_k - 1$. We formulate the discretized differential equations in residual form:

$$\begin{aligned}R_{k,n} &= R_{k,n}(x_{k,n+1}, x_{k,n}, u_k, \hat{d}_k) \\ &= x_{k,n+1} - x_{k,n} - f(x_{k,n+1}, u_k, \hat{d}_k)\Delta t_{k,n} \\ &= 0.\end{aligned} \tag{23}$$

At the boundaries of the control intervals, we enforce continuity through

$$x_{0,0} = \hat{x}_0, \tag{24a}$$

$$x_{k,0} = x_{k-1,N_{k-1}}, \qquad k = 1, \dots, N-1. \tag{24b}$$

B. The simultaneous approach

We transcribe the infinite-dimensional optimal control problem (20) to a finite-dimensional NLP using the temporal discretization based on Euler's implicit method (23)-(24):

$$\min_{\{\{x_{k,n}\}_{n=0}^{N_k}\}_{k=0}^{N-1}, \{u_k\}_{k=0}^{N-1}} \quad \Psi, \tag{25a}$$

subject to

$$x_{0,0} = \hat{x}_0, \tag{25b}$$

$$x_{k,0} = x_{k-1,N_{k-1}}, \quad k = 1, \dots, N-1, \tag{25c}$$

$$\begin{aligned}&R_{k,n}(x_{k,n+1}, x_{k,n}, u_k, \hat{d}_k) = 0, \\ &\qquad n = 0, \dots, N_k - 1, \quad k = 0, \dots, N-1.\end{aligned} \tag{25d}$$

The objective function in (25a) is based on a discretization of the objective function (21) using the right rectangle rule:

$$\begin{aligned}\Psi &= \Psi(\{\{x_{k,n}\}_{n=0}^{N_k}\}_{k=0}^{N-1}, \{u_k\}_{k=0}^{N-1}, \{\hat{d}_k\}_{k=0}^{N-1}) \\ &= \sum_{k=0}^{N-1} \sum_{n=0}^{N_k-1} \Phi(x_{k,n+1}, u_k, \hat{d}_k)\Delta t_{k,n} \\ &+ \hat{\Phi}(x_{N-1,N_{N-1}}, u_{N-1}, \hat{d}_{N-1}).\end{aligned} \tag{26}$$

We use the right rectangle rule to be consistent with the discretization of the differential equations using Euler's implicit method.

We consider two approaches for solving the NLP (25). In the first approach, we directly incorporate the continuity constraints (25b)-(25c) into the NLP as actual constraints. We refer to this as the standard approach. In the second approach, we use the continuity constraints to eliminate the states at the beginning of every control interval, $\{x_{k,0}\}_{k=0}^{N-1}$. This leads to a reduced number of decision variables and constraints in the NLP. We refer to this as the reduced approach. A possible advantage of the standard approach is that the constraints are more loosely coupled which may lead to faster convergence (i.e. fewer iterations) when solving the NLP, whereas the advantage of the reduced approach is that the computational cost per iteration in the numerical solution of the NLP is lower (due to the lower number of decision variables and constraints).

In both approaches, we solve an NLP in the form

$$\min_{s} \qquad \Psi(s, \hat{d}), \tag{27a}$$

$$\text{subject to} \quad R(s, \hat{x}_0, \hat{d}) = 0, \tag{27b}$$

where $\hat{d} = [\hat{d}_0; \cdots; \hat{d}_{N-1}]$. In the standard approach, the decision variables are

$$s = [s_0; \cdots; s_{N-1}], \tag{28a}$$

$$s_k = [x_{k,0}; \cdots; x_{k,N_k}; u_k], \quad k = 0, \dots, N-1, \tag{28b}$$

and the residual equations (27b) include the continuity constraints (25b)-(25c). In the reduced approach, the decision variables are

$$s = [s_0; \cdots; s_{N-1}], \tag{29a}$$

$$s_k = [x_{k,1}; \cdots; x_{k,N_k}; u_k], \tag{29b}$$

and the residual equations (27b) do not include the continuity constraints.

The total number of decision variables in the standard approach is $(N_t+N)n_x+Nn_u$ where $N_t = \sum_{k=0}^{N-1} N_k$ is the total number of time steps, n_x is the dimension of the state vector, and n_u is the dimension of the manipulated inputs. The total number of residual equations is $(N_t + N)n_x$. The total number of decision variables in the reduced approach is $N_t n_x + N n_u$, and the total number of residual equations is $N_t n_x$. Consequently, in the reduced approach, the NLP involves $N n_x$ fewer decision variables and constraints than in the standard approach. This difference can be considerable if N_t and N are of comparable size and if n_x is significantly larger than n_u.

IV. Implementation details

We implement the two collocation-based approaches described in Section III in C, and we use IPOPT 3.12.12 [15] to solve the NLP (27). Apart from providing the objective function, $\Psi(s,\hat{d})$, and the residual function, $R(s,\hat{x}_0,\hat{d})$, we provide IPOPT with 1) the gradient of the objective function, $\frac{\partial \Psi}{\partial s}$, 2) the Jacobian of the residual function, $\frac{\partial R}{\partial s}$, and 3) the Hessian of the Lagrangian, $\nabla^2_{ss}L(s,\hat{d})$, where the Lagrangian is $L(s,\hat{d}) = \sigma\Psi(s,\hat{d}) + \lambda^T R(s,\hat{x}_0,\hat{d})$ [16, Sec. 9]. IPOPT provides both the scaling factor σ and the vector of Lagrange multipliers λ. The Jacobian matrix of the residual function and the Hessian of the Lagrangian are large and sparse, and we provide the exact sparsity patterns of these two matrices (i.e. we do not represent any zero entries unnecessarily).

IPOPT requires third party software related to the solution of linear systems of equations. We use the MA57 routine from HSL [17] as well as METIS 4.0.3 [18]. Furthermore, we compare the computational efficiency of using the OpenBLAS 0.2.20 linear algebra software and Netlib's implementation of BLAS (downloaded using IPOPT). In both cases, we use LAPACK 3.4.2 (also downloaded using IPOPT).

We compile the C code (including IPOPT) using gcc 5.4.0, and we carry out the computations presented in this paper on a 64-bit workstation which uses the operating system Ubuntu 16.04 LTS. The workstation has 15.6 GB RAM and four cores, each of which contains two Intel Core i7 3.60 GHz processors (eight processors in total). The four cores share a level 3 cache of 8192 KB, and the level 2 and level 1 cache of each core have 256 KB and 64 KB (32 KB instruction cache and 32 KB data cache), respectively.

V. Numerical examples

In this section, we present a performance study of the solution of two optimal control problems. The objective in both examples is to maximize the profit (19) over a control and prediction horizon of 30 h consisting of $N = 600$ control intervals with $N_k = 1$ time steps per control interval. The first optimal control problem is the optimal startup problem considered by Drejer et al. [14]. The second optimal control problem is the one that would be solved next in a closed-loop nonlinear model predictive control algorithm. The purpose of the second problem is to test the computational efficiency of warm starting.

In the first problem, at the initial time, the U-loop reactor contains 0.1 kg/m^3 biomass in the top tank and in the U-loop pipe and no substrate or oxygen. We consider two initial guesses, one that is constant in time, and one in which the states are obtained from a linear interpolation of the states in the first initial guess and the initial condition (the manipulated inputs are identical in the two initial guesses). Fig. 2 and Fig. 3 show the optimal states and the manipulated inputs (together with the corresponding initial guesses and bounds). The initial guesses and the bounds on the concentrations of biomass, substrate, and dissolved oxygen in the U-loop pipe are the same as for the concentrations in the top tank. The initial guess of the oxygen in the gas phase of the U-loop pipe is 2.0 kg/m^3, and the upper and lower bounds are 6.0 kg/m^3 and 0.0 kg/m^3, respectively.

TABLE II

Number of decision variables and residual equations in the standard and reduced collocation-based approaches. There are 83 state variables and 3 manipulated inputs.

Approach	Standard	Reduced
Number of decision variables	102600	52200
Number of residual equations	100800	50400

For the second problem, we construct an initial guess, $s^{\text{ws}} = [s_0^{\text{ws}}; \cdots; s_{N-1}^{\text{ws}}]$, by $s_k^{\text{ws}} = s_{k+1}^*$ for $k = 0, \ldots, N-2$ and $s_{N-1}^{\text{ws}} = s_{N-2}^{\text{ws}}$ where $s^* = [s_0^*; \cdots; s_{N-1}^*]$ is the solution to the first optimal control problem. We construct initial guesses of the Lagrange multipliers in a similar manner. The initial condition in the second problem is the optimal states at the end of the first control interval ($k = 0$) in the first optimal control problem, i.e. $\hat{x}_0 = x_{0,N_0}^*$. The bounds are identical in the two problems.

Table II shows the number of decision variables and residual equations in the two approaches, and Table III shows various key performance indicators for the solution of the two optimal control problems. For the standard approach, it is better to use the interpolated initial guess whereas for the reduced approach, the interpolated initial guess leads to slow convergence. The use of warm starting leads to a significant reduction in the number of iterations and the computation time for both approaches, and we are able to solve the optimal control problem in 28.5 s which is less than the length of the control intervals (3 min). This demonstrates that it is feasible to solve the optimal control problems in real-time. Finally, the use of OpenBLAS generally improves the computational performance, and in some cases, it leads to a significant reduction in the number of iterations in IPOPT.

VI. Conclusion

In this work, we consider economic optimal control of SCP production in a U-loop reactor. The dynamical model of the U-loop reactor consists of both partial and ordinary differential equations. We use a finite-volume approach to discretize the partial differential equations. This leads to a set of ordinary differential equations with 83 states. Consequently, the optimal control problems are large-scale. We describe two collocation-based approaches for solving these large-scale optimal control problems. We implement the two approaches in C, and we use IPOPT 3.12.12 to solve the involved NLP (which involves up to around 100,000 decision variables). Furthermore, we present a performance study which demonstrates that it is feasible to solve optimal control problems that involve a U-loop reactor in real-time. Future work involves 1) the use of fully implicit Runge-Kutta (FIRK) methods instead of Euler's implicit method in the collocation-based approaches and 2) implementation of economic nonlinear model predictive control for SCP production in a U-loop reactor.

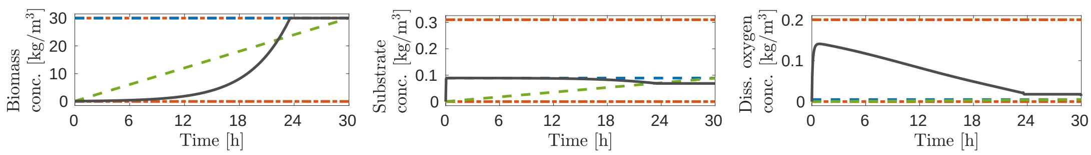


Fig. 2. Constant initial guesses (blue dashed), interpolated initial guesses (green dashed), and optimal values (black solid) of the concentrations of biomass, substrate, and dissolved oxygen in the top tank together with their bounds (red dash-dotted). The constant initial guess of the biomass concentration coincides with the corresponding optimal bound.

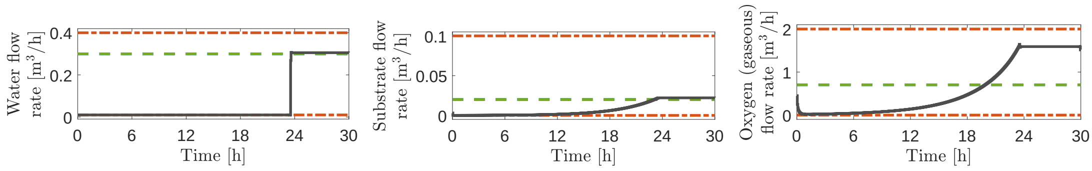


Fig. 3. Initial guesses (green dashed) and optimal values (black solid) of the manipulated inputs (water, substrate, and oxygen feed flow rates) together with their bounds (red dash-dotted). The manipulated inputs are identical in the constant and the interpolated initial guesses.

TABLE III

Key performance indicators of the standard (Std.) and reduced (Red.) collocation-based approaches using different starting guesses and linear algebra software.

Initial guess	Constant		Interpolated		Warm started	
Approach	Std.	Red.	Std.	Red.	Std.	Red.
Netlib BLAS						
Iterations	328	317	159	624	60	24
Func. eval.	329	318	232	647	71	25
Con. eval.	329	318	232	647	71	25
Grad. eval.	329	318	88	617	61	25
Jac. eval.	329	318	161	629	61	25
Hess. eval.	328	317	159	624	60	24
CPU time (s)	302.3	268.9	164.4	1113.9	71.9	31.4
OpenBLAS						
Iterations	330	299	145	426	29	24
Func. eval.	331	300	152	445	30	25
Con. eval.	331	300	152	445	30	25
Grad. eval.	331	300	126	419	30	25
Jac. eval.	331	300	147	430	30	25
Hess.eval.	330	299	145	426	29	24
CPU time (s)	284.7	238.6	199.9	703.3	36.9	28.5

References

[1] T. Binder, L. Blank, H. G. Bock, R. Bulirsch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J. P. Schlöder, and O. von Stryk, "Introduction to model based optimization of chemical processes on moving horizons," in *Online Optimization of Large Scale Systems*. Springer-Verlag Berlin Heidelberg, 2001, pp. 295–339.

[2] S. Kameswaran, L. T. Biegler, and G. H. Staus, "Dynamic optimization for the core-flooding problem in reservoir engineering," *Computers & chemical engineering*, vol. 29, no. 8, pp. 1787–1800, 2005.

[3] T. A. N. Heirung, M. R. Wartmann, J. D. Jansen, B. E. Ydstie, and B. A. Foss, "Optimization of the water-flooding process in a small 2D horizontal oil reservoir by direct transcription," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 10 863–10 868, 2011.

[4] L. T. Biegler, "An overview of simultaneous strategies for dynamic optimization," *Chemical Engineering and Processing: Process Intensification*, vol. 46, no. 11, pp. 1043–1053, 2007.

[5] H. G. Bock and K. J. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems," *IFAC Proceedings Volumes*, vol. 17, no. 2, pp. 1603–1608, 1984.

[6] A. Schäfer, P. Kühl, M. Diehl, J. Schlöder, and H. G. Bock, "Fast reduced multiple shooting methods for nonlinear model predictive control," *Chemical Engineering and Processing: Process Intensification*, vol. 46, no. 11, pp. 1200–1214, 2007.

[7] J. Nocedal and S. J. Wright, *Numerical optimization*, 2nd ed. Springer Science & Business Media, 2006.

[8] D. F. Olsen, J. B. Jørgensen, J. Villadsen, and S. B. Jørgensen, "Modeling and simulation of single cell protein production," *IFAC Proceedings Volumes*, vol. 43, no. 6, pp. 502–507, 2010.

[9] O. A. Prado-Rubio, J. B. Jørgensen, and S. B. Jørgensen, "Systematic model analysis for single cell protein (SCP) production in a U-loop reactor," *Computer Aided Chemical Engineering*, vol. 28, pp. 319–324, 2010.

[10] A. M. Al Taweel, Q. Shah, and B. Aufderheide, "Effect of mixing on microorganism growth in loop bioreactors," *International Journal of Chemical Engineering*, no. Article ID 984827, 2012.

[11] M. Wu, J. K. Huusom, K. V. Gernaey, and U. Krühne, "Modelling and simulation of a U-loop reactor for single cell protein production," *Computer Aided Chemical Engineering*, vol. 38, pp. 1287–1292, 2016.

[12] L. A. H. Petersen, J. Villadsen, S. B. Jørgensen, and K. V. Gernaey, "Mixing and mass transfer in a pilot scale U-loop bioreactor," *Biotechnology and Bioengineering*, vol. 114, no. 2, pp. 344–354, 2017.

[13] D. F. Olsen, J. B. Jørgensen, J. Villadsen, and S. B. Jørgensen, "Optimal operating points for SCP production in the U-loop reactor," *IFAC Proceedings Volumes*, vol. 43, no. 5, pp. 499–504, 2010.

[14] A. Drejer, T. Ritschel, S. B. Jørgensen, and J. B. Jørgensen, "Economic optimizing control for single-cell protein production in a U-loop reactor," *Computer Aided Chemical Engineering*, vol. 40, pp. 1759–1764, 2017.

[15] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.

[16] A. Wächter, "Short tutorial: getting started with Ipopt in 90 minutes," in *Combinatorial Scientific Computing*, ser. Dagstuhl Seminar Proceedings, U. Naumann, O. Schenk, H. D. Simon, and S. Toledo, Eds., no. 09061. Dagstuhl, Germany: Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2009. [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2009/2089

[17] HSL, "A collection of Fortran codes for large scale scientific computation," www.hsl.rl.ac.uk, accessed February 2nd 2019.

[18] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 359–392, 1998.

APPENDIX E

Paper V - CCTA 2019 B

The extended Kalman filter for nonlinear state estimation in a u-loop bioreactor

Authors:
Tobias K. S. Ritschel, Dimitri Boiroux, Marcus Krogh Nielsen, Jakob Kjøbsted Huusom, Sten Bay Jørgensen, John Bagterp Jørgensen

Published in:
proceedings of the 2019 IEEE conference on control technology and applications (CCTA). August 19-21, 2019. Hong Kong, China.
DOI: `https://doi.org/10.1016/10.1109/CCTA.2019.8920643`

The Extended Kalman Filter for Nonlinear State Estimation in a U-loop Bioreactor

Tobias K. S. Ritschel, Dimitri Boiroux, Marcus Krogh Nielsen, Jakob Kjøbsted Huusom, Sten Bay Jørgensen, John Bagterp Jørgensen

Abstract— **In this paper, we consider nonlinear state estimation in the U-loop reactor for single-cell protein (SCP) production. The model of the U-loop reactor is a mixture of stochastic partial differential equations and stochastic differential equations which are stiff. By a typical finite-volume spatial discretization, the resulting system of stochastic differential equations for numerical simulation and state estimation has 83 states. We investigate and discuss the continuous-discrete EKF for state estimation in this high-dimensional and stiff continuous-discrete-time system.**

I. Introduction

State estimation as a filter algorithm is a central component in monitoring, fault detection, and model predictive control. When the process is nonlinear, a nonlinear filtering algorithm is needed. The nonlinear filtering problem is a computationally hard problem. It is well known that the solution of the Fokker-Planck equation (Kolmogorov's forward equation), i.e. the exact probability density function of the states, is the optimal solution to the state estimation problem [1]. However, due to the curse of dimensionality and computational tractability, the Fokker-Planck equation is restricted to low dimensional problems and cannot be applied to most practical problems. It can certainly not be applied to estimation in systems where the model is a number of partial and ordinary differential equations. Therefore, a number of alternative approximate nonlinear filtering algorithms are used. These algorithms are also challenged by the high state dimensionality of models arising from finite-volume discretizaton of partial differential equation systems. Single-cell protein (SCP) production in the U-loop reactor studied in this paper gives rise to a coupled system of partial and ordinary differential equations. Consequently, a finite-volume spatial discretization gives rise to a high-dimensional system. In this paper, we investigate the extended Kalman filter (EKF) for nonlinear state estimation in this system.

*This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 723661, Spire project: "Coordinating Optimisation of Complex Industrial Processes" (www.cocop-spire.eu). This article reflects only the author's views and the Commission is not responsible for any use that may be made of the information contained therein. This work also received funding in the project EUDP 64013-0558 in the IEA annex for energy efficient process control.
Tobias K. S. Ritschel, Dimitri Boiroux, Marcus Krogh Nielsen, and John Bagterp Jørgensen are with the Department of Applied Mathematics and Computer Science, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark. Jakob Kjøbsted Huusom and Sten Bay Jørgensen are with the Department of Chemical and Biochemical Engineering, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark. Tobias K. S. Ritschel is also with 2-control ApS, DK-2900 Hellerup, Denmark. jbjo@dtu.dk

A. Nonlinear filtering algorithms

The Kalman filter is an optimal filter for linear systems with normally distributed process and measurement noise [2]. When the system is nonlinear, high-dimensional, and evolving in continuous-time, a number of different approximations such as the EKF and the unscented Kalman filter (UKF) have been suggested, while approaches based on the Fokker-Planck equation, hidden Markov models, particle filtering, or the ensemble Kalman filter (EnKF) suffer from the curse of dimensionality [1]–[7]. Most literature on nonlinear filtering algorithms for process applications considers low dimensional and discrete-discrete systems with only few details related to the numerical methods [8]–[13]. In this paper, we formulate the EKF for a *high-dimensional continuous-discrete* system stemming from a mixed system of stochastic partial and ordinary differential equations that are measured at discrete times. Such systems are ubiquitous in process systems engineering. We use the EKF because it is computationally efficient and involves few tuning parameters, e.g. compared to the UKF. Furthermore, we demonstrate the relevance of the formulated EKF by application to a novel process from industrial biotechnology.

B. Single cell protein production in a U-loop reactor

Methanotrophs can grow on cheap carbon sources such as methane or methanol. They have a high protein content and can be used to produce SCP. SCP can be used for animal feed and thereby sustain a growing human population. SCP may be produced using a U-loop reactor. However, the operation of such a reactor and in particular the startup is non-trivial. Using a mathematical model that describes the dynamics of SCP production in a U-loop reactor [14]–[18], we use the EKF to estimate the state of the reactor during startup based on the economic optimizing control strategy described by Drejer et al. [19]. The information provided by this state estimation is central to economic nonlinear model predictive control [20] for economically optimal operation and startup of the U-loop reactor. Previously, Olsen et al. [21] computed the optimal steady-state operating points, but did not consider the startup. Furthermore, state estimation has not previously been considered for the U-loop reactor. The computation and implementation of the optimal startup profile is challenging as the optimal profile turns out to be an unstable attractor, i.e. an uncontrolled system diverges from this optimal trajectory. Thus, using an open-loop control strategy would cause the startup of the U-loop reactor to fail [19].

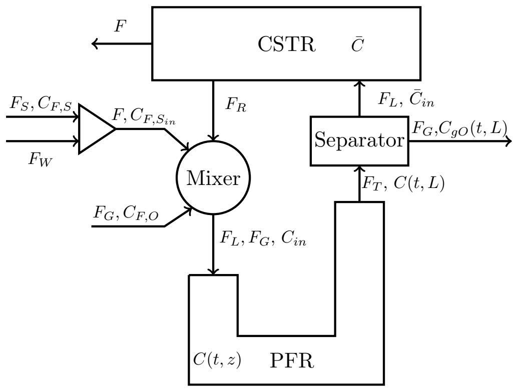


Fig. 1. U-loop reactor diagram for mathematical modeling.

C. Paper organization

The remaining part of the paper is organized as follows. Section II presents the mathematical model for SCP production in a U-loop reactor with methanol as the feed. Section III discusses the EKF for continuous-discrete systems, and in Section IV, we present the numerical example of state estimation during startup of the U-loop reactor. Section V summarizes the conclusions.

II. Mathematical model

Fig. 1 is a conceptual diagram used to derive a mathematical model of SCP production in the U-loop reactor [14], [15], [19], [21]. At its inlet, the U-loop section consists of a mixer, in which the recirculated mixture from the top tank is mixed with feed substrate (methanol), feed water, and feed gas (oxygen). The remaining part of the U-loop is modeled as a liquid-gas phase plug flow reactor (PFR). The end of the U-loop is modeled as an ideal separator that completely separates the gas phase from the liquid phase. The top tank is modeled as a liquid phase continuous stirred tank reactor (CSTR). A liquid mixture of water, biomass, substrate, and dissolved gas is harvested from the top tank. The top tank also recirculates part of the mixture.

A. Mixing section

The gas flow rate at the inlet of the PFR is equal to the feed gas flow rate, F_G. The liquid flow rate, F_L, to the inlet of the PFR is obtained by a total static mass balance:

$$F_L = F_R + F_S + F_W. \tag{1}$$

F_R is the flow rate of liquid from the top tank to the U-loop, F_S is the flow rate of feed substrate, and F_W is the flow rate of feed water. Component mass balances for the biomass, X, the substrate, S, the oxygen dissolved in the liquid phase, O, and the oxygen in the gas phase, gO, give the following inlet concentrations to the PFR

$$C_{in,X} = \frac{F_R \bar{C}_X}{F_L}, \tag{2a}$$
$$C_{in,S} = \frac{F_S C_{F,S} + F_R \bar{C}_S}{F_L}, \tag{2b}$$
$$C_{in,O} = \frac{F_R \bar{C}_O}{F_L}, \tag{2c}$$
$$C_{in,gO} = C_{F,O}. \tag{2d}$$

$\bar{C}_X$, $\bar{C}_S$, and $\bar{C}_O$ denote the concentrations of biomass (X), substrate (S), and oxygen (O) in the top tank.

B. The U-loop modeled as a plug-flow reactor

The corresponding inlet fluxes, N_i, to the PFR are

$$N_i(t,0) = vC_{in,i}(t), \quad i \in \{X,S,O,gO\}, \tag{3}$$

where the linear velocity is computed as

$$v = \frac{F_L + F_G}{A}. \tag{4}$$

A is the cross-sectional area of the U-loop pipe. The concentrations, $C_i = C_i(t,z)$, of all components, $i \in \{X,S,O,gO\}$, are computed as functions of time, t, and position, z, by the following mass balances

$$\frac{\partial C_X}{\partial t} = -\frac{\partial N_X}{\partial z} + R_X, \tag{5a}$$
$$\frac{\partial C_S}{\partial t} = -\frac{\partial N_S}{\partial z} + R_S, \tag{5b}$$
$$\frac{\partial C_O}{\partial t} = -\frac{\partial N_O}{\partial z} + R_O + \frac{1}{1-\varepsilon} J_{gl,O}, \tag{5c}$$
$$\frac{\partial C_{gO}}{\partial t} = -\frac{\partial N_{gO}}{\partial z} - \frac{1}{\varepsilon} J_{gl,O}, \tag{5d}$$

for $t_a \leq t \leq t_b$ and $0 \leq z \leq L$. t_a is the initial time, t_b is the final time, and L is the length of the U-loop pipe. $N_i = N_i(C_i(t,z)) = N_i(t,z)$ denotes the flux of each component, $i \in \{X,S,O,gO\}$, $R_i = R_i(t,z)$ denotes the production rate for $i \in \{X,S,O\}$, and $J_{gl,O} = J_{gl,O}(t,z)$ is the transfer rate of oxygen from the gas phase to the liquid phase. $\varepsilon = F_G/(F_G + F_L)$ is the fraction of gas in each cross section. It is assumed to be constant throughout the U-loop pipe.

Danckwerts' conditions describe the outlet boundary conditions of the PFR, i.e.

$$\frac{\partial C_i}{\partial z}(t,L) = 0, \quad i \in \{X,S,O,gO\}, \quad t_a \leq t \leq t_b. \tag{6}$$

C. Gas-liquid separating section

We assume that the gas and liquid are perfectly and instantaneously separated when the fast flowing gas-liquid mixture from the U-loop pipe enters the top tank. This is modeled as an ideal static gas-liquid separator, where the gas is completely removed, and the liquid phase enters the liquid phase in the top tank. This implies that the inlet concentrations of the liquid phase to the top tank are

$$\bar{C}_{in,i}(t) = C_i(t,L), \quad i \in \{X,S,O\}, \quad t_a \leq t \leq t_b. \tag{7}$$

D. Top-tank model

The top tank is modeled as a CSTR with a constant liquid volume. The mass balances for the components in the liquid phase are

$$\frac{d\bar{C}_i}{dt} = \bar{D}\left(\bar{C}_{in,i} - \bar{C}_i\right) + R_i(\bar{C}_X, \bar{C}_S, \bar{C}_O),\ i \in \{X, S, O\}, \tag{8}$$

for $t_a \leq t \leq t_b$. The dilution rate is $\bar{D} = \bar{F}/V$, where $\bar{F} = F_L$ and V is the liquid volume in the top tank. As the liquid volume in the top tank is constant, the flow rate of the product stream from the top tank is

$$F = \bar{F} - F_R = F_L - F_R = F_S + F_W. \tag{9}$$

E. Convective and diffusive transport

The flux, N_i, consists of convective transport, vC_i, and diffusive transport, J_i:

$$N_i = v_i C_i + J_i, \quad i \in \{X, S, O, gO\}. \tag{10}$$

The diffusive transport is modeled using Fick's law:

$$J_i = -D_i \frac{\partial C_i}{\partial z}, \quad i \in \{X, S, O, gO\}. \tag{11}$$

F. Gas-liquid transport

The transport of oxygen from the gas phase to the liquid phase is governed by the relation

$$J_{gl,O} = (k_L a)_O (C_{sat,O} - C_O), \tag{12}$$

where Henry's law in combination with the ideal gas law provides the saturation concentration of oxygen, i.e.

$$C_{sat,O} = \frac{P_{gO}}{H_O} = \frac{RT}{M_{wO} H_O} C_{gO}. \tag{13}$$

H_O is the Henry constant for oxygen, P_{gO} is the partial pressure of oxygen in the gas phase, R is the gas constant, T is the temperature, M_{wO} is the molar weight of oxygen, and C_{gO} is the concentration of oxygen in the gas phase.

G. Stoichiometry and kinetics

The overall conversion of methanol (S) to biomass (X) is governed by the stoichiometric relation

$$\begin{aligned} &CH_3OH + Y_{SN} HNO_3 + Y_{SO} O_2 \rightarrow \\ &\quad Y_{SX} X + Y_{SC} CO_2 + Y_{SW} H_2O \end{aligned} \tag{14}$$

which can also be formulated as

$$\begin{aligned} &Y_{XS} CH_3OH + Y_{XN} HNO_3 + Y_{XO} O_2 \rightarrow \\ &\quad X + Y_{XC} CO_2 + Y_{XW} H_2O. \end{aligned} \tag{15}$$

Table I reports the yield coefficients for this overall reaction. The stoichiometry implies that the production rate of substrate (methanol), R_S, and the production rate of oxygen in the liquid phase, R_O, are related to the production rate of biomass, R_X, by

$$R_S = -\gamma_S R_X, \qquad \gamma_S = \frac{M_{wS}}{M_{wX} Y_{SX}}, \tag{16a}$$

$$R_O = -\gamma_O R_X, \qquad \gamma_O = \frac{M_{wO} Y_{SO}}{M_{wX} Y_{SX}}. \tag{16b}$$

TABLE I
YIELD COEFFICIENTS

	i	Y_{Si} [mol/mol]	Y_{Xi} [mol/mol]	M_{wi} [g/mol]	W_{Xi} [g/g]
CH_3OH	S	1.000	1.366	32.042	1.778
HNO_3	N	0.146	0.199	63.013	0.510
O_2	O	0.439	0.600	31.999	0.779
$CH_{1.8}O_{0.5}N_{0.2}$	X	0.732	1.000	24.626	1.000
CO_2	C	0.268	0.366	44.010	0.654
H_2O	W	1.415	1.933	18.015	1.414

The production rate of biomass, R_X, is given by

$$R_X = \mu(C_S, C_O) C_X, \tag{17}$$

where $\mu = \mu(C_S, C_O)$ is the specific growth rate. The specific growth rate of Methylococcus Capsulatus in a methanol medium is given by the Monod-Haldane expression

$$\mu = \mu(C_S, C_O) = \mu_{\max} \mu_S(C_S) \mu_O(C_O), \tag{18a}$$

where the growth factors, $\mu_S(C_S)$ and $\mu_O(C_O)$, are

$$\mu_S(C_S) = \frac{C_S}{K_S + C_S + C_S^2/K_I}, \tag{18b}$$

$$\mu_O(C_O) = \frac{C_O}{K_O + C_O}. \tag{18c}$$

H. Stochastic effects and parameters

In this paper, we model the variation of some key parameters as stochastic processes,

$$d\mu_{\max}(t) = \kappa_\mu(\bar{\mu}_{\max} - \mu_{\max}(t))dt + \sigma_\mu dw_\mu(t), \tag{19a}$$

$$d\gamma_S(t) = \kappa_{\gamma_S}(\bar{\gamma}_S - \gamma_S(t))dt + \sigma_{\gamma_S} dw_{\gamma_S}(t), \tag{19b}$$

$$d\gamma_O(t) = \kappa_{\gamma_O}(\bar{\gamma}_O - \gamma_O(t))dt + \sigma_{\gamma_O} dw_{\gamma_O}(t), \tag{19c}$$

around the mean (i.e. the nominal) values of the parameters. Similarly, we consider stochastic variations of the feed substrate concentration according to

$$dC_{F,S}(t) = \kappa_{C_{F,S}}(\bar{C}_{F,S}(t) - C_{F,S}(t))dt + \sigma_{C_{F,S}} dw_{C_{F,S}}(t). \tag{20}$$

III. Nonlinear state estimation

Using spatial discretization on N_z equidistantly spaced finite volumes, the mathematical model for the U-loop reactor presented in Section II can be compactly represented as

$$d\boldsymbol{x}_d(t) = f(\boldsymbol{x}_d(t), \boldsymbol{x}_s(t), u(t), \theta)dt, \tag{21a}$$

and the corresponding compact model for the stochastic effects described in Section II-H is

$$d\boldsymbol{x}_s(t) = f_s(\boldsymbol{x}_s(t), u(t), \theta)dt + \sigma_s d\boldsymbol{w}_s(t). \tag{21b}$$

In this section, we describe the EKF for continuous-discrete-time stochastic systems in the form

$$d\boldsymbol{x}(t) = f(\boldsymbol{x}(t), u(t), \theta)dt + \sigma(\boldsymbol{x}(t), u(t), \theta)d\boldsymbol{w}(t), \tag{22a}$$

$$\boldsymbol{z}(t) = g(\boldsymbol{x}(t), \theta), \tag{22b}$$

$$\boldsymbol{y}(t_k) = \boldsymbol{z}(t_k) + \boldsymbol{v}(t_k; \theta). \tag{22c}$$

The combined model (21) is a special case of the stochastic differential equation (22a), (22b) is a model of the process outputs, and (22c) is a model of the measurements. $\boldsymbol{x}$ is the state vector, u is the vector of manipulated inputs, θ is the parameter vector, $\boldsymbol{z}$ is the vector of outputs, and $\boldsymbol{y}$ are measurements of the outputs (at discrete times, t_k) which are corrupted by measurement noise, $\boldsymbol{v}(t_k;\theta) \sim N_{iid}(0, R_v(\theta))$. $\boldsymbol{w}(t)$ is a standard Wiener process, i.e. $d\boldsymbol{w}(t) \sim N_{iid}(0, I dt)$. The initial states are normally distributed: $\boldsymbol{x}(t_0) \sim N(\bar{x}_0, P_0)$.

A. Simulation and temporal discretization

We use a semi-implicit method [22] to discretize the stochastic differential equation (22a):

$$\begin{aligned} x_{k,n+1} - x_{k,n} &= f(x_{k,n+1}, u_k, \theta)\Delta t_{k,n} \\ &\quad + \sigma(x_{k,n}, u_k, \theta)\Delta w_{k,n}. \end{aligned} \tag{23}$$

$\Delta w_{k,n}$ is a realization of $\Delta \boldsymbol{w}_{k,n} \sim N_{iid}(0, I\Delta t_{k,n})$. The states, $x_{k,n+1}$, are obtained by solution of

$$\begin{aligned} R_{k,n} &= R_{k,n}(x_{k,n+1}) \\ &= x_{k,n+1} - f(x_{k,n+1}, u_k, \theta)\Delta t_{k,n} \\ &\qquad - x_{k,n} - \sigma(x_{k,n}, u_k, \theta)\Delta w_{k,n} \\ &= 0, \end{aligned} \tag{24}$$

using a variant of Newton-Raphson's method.

B. Extended Kalman filter

The EKF for the continuous-discrete-time stochastic system (22) consists of a filter and a one-step predictor.

1) Initialization: The initialization of the EKF is based on the prior information represented by the distribution of the initial states. Consequently, $\hat{x}_{0|-1} = \bar{x}_0$ and $P_{0|-1} = P_0$.

2) Filter step: At each discrete time k, i.e. time t_k, the filter assumes that $\hat{x}_{k|k-1}$ and $P_{k|k-1}$ are available. It computes the one-step prediction of the output, $\hat{z}_{k|k-1}$, and the measurement, $\hat{y}_{k|k-1}$, by

$$\hat{y}_{k|k-1} = \hat{z}_{k|k-1} = g(\hat{x}_{k|k-1}, \theta). \tag{25}$$

Linearization of the output equation provides

$$C_k = \frac{\partial g}{\partial x}(\hat{x}_{k|k-1}, \theta), \tag{26}$$

such that the innovation, e_k, and its covariance, $R_{e,k} = \langle \boldsymbol{y}_{k|k-1}, \boldsymbol{y}_{k|k-1} \rangle$, may be computed by

$$e_k = y_k - \hat{y}_{k|k-1}, \tag{27a}$$
$$R_{e,k} = C_k P_{k|k-1} C_k' + R_v(\theta), \tag{27b}$$

where y_k is the measurement. The Kalman filter gain, $K_{fx,k} = \langle \boldsymbol{x}_{k|k-1}, \boldsymbol{y}_{k|k-1} \rangle \langle \boldsymbol{y}_{k|k-1}, \boldsymbol{y}_{k|k-1} \rangle^{-1}$, is

$$K_{fx,k} = P_{k|k-1} C_k' R_{e,k}^{-1}. \tag{28}$$

The Kalman filter gain, $K_{fx,k}$, is used in the computation of the filtered state, $\hat{x}_{k|k}$, and its covariance, $P_{k|k} = \langle \boldsymbol{x}_{k|k-1}, \boldsymbol{x}_{k|k-1} \rangle - \langle \boldsymbol{x}_{k|k-1}, \boldsymbol{y}_{k|k-1} \rangle \langle \boldsymbol{y}_{k|k-1}, \boldsymbol{y}_{k|k-1} \rangle^{-1} \langle \boldsymbol{y}_{k|k-1}, \boldsymbol{x}_{k|k-1} \rangle$. They are computed according to

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_{fx,k} e_k, \tag{29a}$$
$$P_{k|k} = P_{k|k-1} - K_{fx,k} R_{e,k} K_{fx,k}'. \tag{29b}$$

3) One-step predictor: The one-step prediction of the states are computed by solution of the initial value problem

$$\frac{d}{dt}\hat{x}_k(t) = f(\hat{x}_k(t), u_k, \theta), \qquad \hat{x}_k(t_k) = \hat{x}_{k|k}. \tag{30}$$

In order to compute the one-step prediction of the covariance, we compute the sensitivities of the one-step prediction of the states at time t with respect to the states at time s, $\Phi_k(t,s)$, by solving the initial value problem

$$\frac{d}{dt}\Phi_k(t,s) = A_k(t)\Phi_k(t,s), \qquad \Phi_k(s,s) = I, \tag{31}$$

where

$$A_k(t) = \frac{\partial f}{\partial x}(\hat{x}_k(t), u_k, \theta). \tag{32}$$

Using these sensitivities, we compute the covariance at time t by [23]

$$\begin{aligned} P_k(t) &= \Phi_k(t, t_k) P_{k|k} \Phi_k(t, t_k)' \\ &\quad + \int_{t_k}^{t} \Phi_k(t,s)\sigma_k(s)\sigma_k(s)'\Phi_k(t,s)' ds, \end{aligned} \tag{33}$$

where

$$\sigma_k(t) = \sigma(\hat{x}_k(t), u_k, \theta). \tag{34}$$

We use Euler's implicit method (with multiple time steps in between measurements) together with a variant of Newton-Raphson's method to solve the initial value problems (30) and (31), and we use a left rectangle rule to approximate the integral in (33).

IV. Numerical example

In this section, we present a numerical example of state estimation in the U-loop reactor during startup based on the economic optimizing control strategy presented by Drejer et al. [19] which is stabilized by a P-controller (i.e. using feedback). The P-controller changes the substrate feed flow rate every 15 s based on the substrate concentration in the top tank. The feedback dynamics are not represented in the dynamical model used by the EKF. Furthermore, the EKF uses average values of the substrate feed flow rate computed by the P-controller (averaged over each sampling interval). We consider a time interval of 30 h, and the EKF uses measurements of the concentration of the dissolved oxygen at positions $z = 4$ m, $z = 7$ m, and $z = 10$ m in the U-loop pipe. These measurements are obtained every third minute. We discretize the U-loop pipe using 20 equidistant finite volumes. Consequently, we measure the concentrations of the dissolved oxygen in the 7'th, 12'th, and 17'th finite volumes in the discretized U-loop pipe. The resulting dynamical system contains 83 state variables as well as the 3 uncertain parameters, $\mu_{\max}$, γ_S, and γ_O, and the uncertain substrate feed concentration, $C_{F,S}$, as described in Section

II-H. Consequently, the EKF estimates the 87 state variables in the combined dynamical system.

We use a stochastic simulation to represent the true U-loop reactor and to compute the measurements of the dissolved oxygen concentrations. In this stochastic simulation, we use 12 time steps in between the measurements corresponding to the update frequency of the P-controller. We use 6 time steps between the measurements in the EKF. We provide more detail on this numerical example in the Appendix.

Fig. 2 shows the manipulated inputs, and Fig. 3 shows the filtered estimates of the concentrations of biomass, substrate, and dissolved oxygen in the top tank computed using the EKF (in blue) together with the true concentrations (in green). Furthermore, it shows the absolute deviations of the filtered estimates from the true concentrations. The EKF is able to accurately estimate the concentrations of biomass and dissolved oxygen in the top tank. However, the estimates of the substrate concentration in the top tank deviate significantly from the true values. This is possibly because the EKF does not estimate the substrate concentration in the feed, $C_{F,S}$, accurately during the first 15 h of the startup. The filtered estimate of $C_{F,S}$ is shown in Fig. 4 together with the filtered estimates of the uncertain model parameters $\mu_{\max}$, γ_S, and γ_O (in blue) as well as the true values (in green) and the nominal values (in black) of these parameters.

In the implementation of the EKF, the evaluation of the right-hand side function $f(x(t), u(t), \theta)$ in the combined dynamical system (22) is implemented in C, and the rest is implemented in Matlab R2016b. The average computation times of the filtering step and the one-step predictor are 0.1 ms and 9.4 ms which are negligible compared to the sampling time.

V. Conclusion

In this work, we describe the EKF for nonlinear state estimation in a U-loop reactor for SCP production. The EKF requires a dynamical model of the U-loop reactor. We present such a dynamical model which consists of both stochastic partial differential equations and stochastic differential equations, and we use a finite-volume approach to spatially discretize the stochastic partial differential equations. Furthermore, we demonstrate, using a numerical example, that the EKF can be used to estimate the state of the U-loop reactor during startup. In this example, we use a previously developed economic optimizing control strategy together with a P-controller for stabilization. The EKF is implemented using Matlab and C, and the computation time is negligible compared to the sampling time indicating that real-time implementation is feasible. Future work could involve using the UKF for improving the accuracy of the state estimation in the U-loop reactor.

References

[1] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*. San Diego, CA, USA: Academic Press, 1970.

[2] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear Estimation*. Prentice Hall, 2000.

[3] A. Gelb, *Applied Optimal Estimation*. MIT Press, 1974.

[4] D. Simon, *Optimal State Estimation. Kalman, H_∞, and Nonlinear Approaches*. Wiley, 2006.

[5] A. M. Fraser, *Hidden Markov Models and Dynamical Systems*. SIAM, 2008.

[6] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model Predictive Control: Theory, Computation, and Design*, 2nd ed. Madison, WI, USA: Nob Hill Publishing, 2017.

[7] S. Gillijns, O. B. Mendoza, J. Chandrasekar, B. L. R. De Moor, D. S. Bernstein, and A. Ridley, "What is the ensemble Kalman filter and how well does it work?" in *Proceedings of the 2006 American Control Conference*, Minneapolis, Minnesota, USA, 2006, pp. 4448–4453.

[8] M. Nørgaard, N. K. Poulsen, and O. Ravn, "New developments in state estimation for nonlinear systems," *Automatica*, vol. 36, pp. 1627–1638, 2000.

[9] T. S. Schei, "A finite-difference method for linearization in nonlinear estimation algorithms," *Automatica*, vol. 33, no. 11, pp. 2053–2058, 1997.

[10] D. Dochain, "State and parameter estimation in chemical and biochemical processes: a tutorial," *Journal of Process Control*, vol. 13, pp. 801–818, 2003.

[11] J. B. Rawlings and B. R. Bakshi, "Particle filtering and moving horizon estimation," *Computers and Chemical Engineering*, vol. 30, pp. 1529–1541, 2006.

[12] T. S. Schei, "On-line estimation for process control and optimization applications," *Journal of Process Control*, vol. 18, pp. 821–828, 2008.

[13] S. Kolås, B. A. Foss, and T. S. Schei, "Noise modeling concepts in nonlinear state estimation," *Journal of Process Control*, vol. 19, pp. 1111–1125, 2009.

[14] D. F. Olsen, J. B. Jørgensen, J. Villadsen, and S. B. Jørgensen, "Modeling and simulation of single cell protein production," *IFAC Proceedings Volumes*, vol. 43, no. 6, pp. 502–507, 2010.

[15] O. A. Prado-Rubio, J. B. Jørgensen, and S. B. Jørgensen, "Systematic model analysis for single cell protein (SCP) production in a U-loop reactor," *Computer Aided Chemical Engineering*, vol. 28, pp. 319–324, 2010.

[16] A. M. Al Taweel, Q. Shah, and B. Aufderheide, "Effect of mixing on microorganism growth in loop bioreactors," *International Journal of Chemical Engineering*, no. Article ID 984827, 2012.

[17] M. Wu, J. K. Huusom, K. V. Gernaey, and U. Krühne, "Modelling and simulation of a U-loop reactor for single cell protein production," *Computer Aided Chemical Engineering*, vol. 38, pp. 1287–1292, 2016.

[18] L. A. H. Petersen, J. Villadsen, S. B. Jørgensen, and K. V. Gernaey, "Mixing and mass transfer in a pilot scale U-loop bioreactor," *Biotechnology and Bioengineering*, vol. 114, no. 2, pp. 344–354, 2017.

[19] A. Drejer, T. Ritschel, S. B. Jørgensen, and J. B. Jørgensen, "Economic optimizing control for single-cell protein production in a U-loop reactor," *Computer Aided Chemical Engineering*, vol. 40, pp. 1759–1764, 2017.

[20] L. N. Petersen and J. B. Jørgensen, "Real-time economic optimization for a fermentation process using model predictive control," in *European Control Conference (ECC) 2014*. IEEE, 2014, pp. 1831–1836.

[21] D. F. Olsen, J. B. Jørgensen, J. Villadsen, and S. B. Jørgensen, "Optimal operating points for SCP production in the U-loop reactor," *IFAC Proceedings Volumes*, vol. 43, no. 5, pp. 499–504, 2010.

[22] K. Burrage and T. Tian, "The composite Euler method for stiff stochastic differential equations," *Journal of Computational and Applied Mathematics*, vol. 131, pp. 407–426, 2001.

[23] J. B. Jørgensen, M. R. Kristensen, P. G. Thomsen, and H. Madsen, "New extended Kalman filter algorithms for stochastic differential algebraic equations," in *Assessment and Future Directions of Nonlinear Model Predictive Control*, ser. Lecture Notes in Control and Information Sciences. Springer-Verlag Berlin Heidelberg, 2007, vol. 358, pp. 359–366.

Appendix

Table II shows parameters used in the numerical example. The units of the diffusion coefficients, σ_μ, σ_{γ_S}, σ_{γ_O}, and $\sigma_{C_{F,S}}$, contain a factor of 1/h$^{1/2}$ because the unit of the increment of the Wiener process, $d\boldsymbol{w}(t)$, in the stochastic continuous-discrete-time system (22) is h$^{1/2}$, i.e. the unit of the covariance of this increment is h.

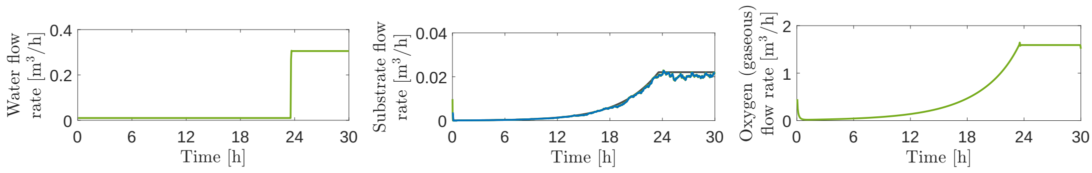


Fig. 2. Manipulated inputs: feed flow rates of water, substrate, and gaseous oxygen. For the substrate flow rate, the green curve represents the values used in the stochastic simulation of the U-loop reactor, the blue curve represents the values used in the EKF, and the black curve represents the open-loop economically optimal strategy computed by Drejer et al. [19] (the green and blue curves almost completely coincide). For the water and oxygen flow rates, the green curve is used both in the stochastic simulation and in the EKF.

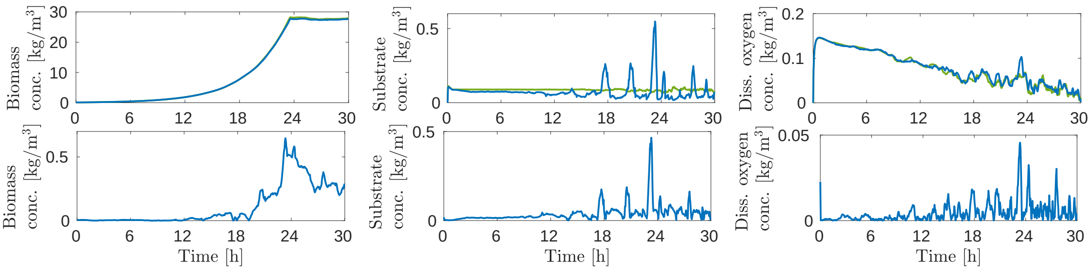


Fig. 3. Top row: filtered estimates of the concentrations in top tank of the U-loop reactor obtained with the EKF (blue) together with the true concentrations (green). Bottom row: absolute differences between the filtered estimates and the true values of these concentrations.

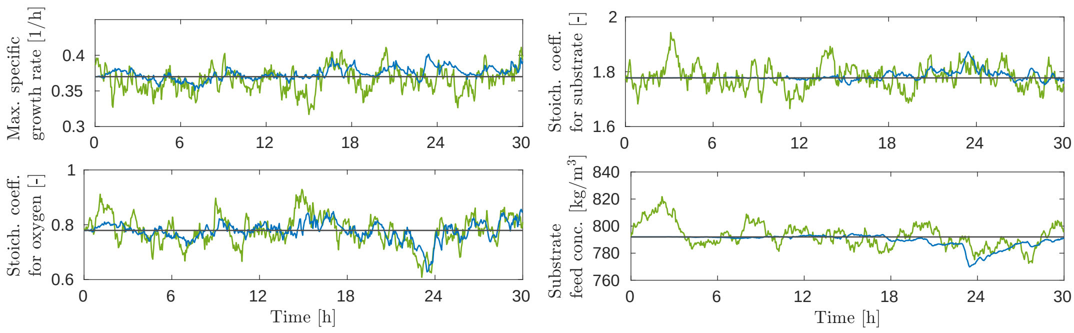


Fig. 4. Filtered estimates of the maximum specific growth rate, $\mu_{\max}$, the stoichiometric coefficients for substrate and oxygen, γ_S and γ_O, and the substrate concentration in the inlet, $C_{F,S}$ computed using the EKF (blue) together with the true (green) and nominal (black) values of these parameters.

The mean of the initial states, $\bar{x}_0$, is constructed such that the biomass concentration in the U-loop pipe and the top tank are 0.1 kg/m^3, and the reactor contains no substrate or oxygen (neither dissolved or in gaseous form). The covariance of the initial states, P_0, used in the EKF and to sample the true initial states is a diagonal matrix whose entries are 10^{-4} (the units vary).

The P-controller used to stabilize the open-loop economic optimizing control strategy presented by Drejer et al. [19] is

$$\tilde{F}_S = F_S^* + K_{cS}(\bar{C}_S^* - \bar{C}_S), \tag{35a}$$

where F_S^* and $\bar{C}_S^*$ are the optimized open-loop substrate feed flow rate and the substrate concentration in the top tank. $\bar{C}_S$ is the actual substrate concentration in the top tank, and in this example, we use $K_{cS} = 0.1$.

TABLE II
PARAMETER VALUES USED IN THE NUMERICAL EXAMPLE.

Symbol	Value	Unit
$\bar{\mu}_{\max}$	0.37	1/h
$\bar{\gamma}_S$	1.78	-
$\bar{\gamma}_O$	0.78	-
$\bar{C}_{F,S}$	792	kg/m^3
κ_μ	2	1/h
κ_{γ_S}	2	1/h
κ_{γ_O}	2	1/h
$\kappa_{C_{F,S}}$	0.5	1/h
σ_μ	0.04	1/h$^{3/2}$
σ_{γ_S}	0.1	1/h$^{1/2}$
σ_{γ_O}	0.1	1/h$^{1/2}$
$\sigma_{C_{F,S}}$	10.0	kg/(m^{3}h$^{1/2}$)
R_v	$10^{-4} \cdot \mathrm{I}_{3\times 3}$	(kg/m^3)^2

APPENDIX F

Paper VI - ECC 2020

Economic nonlinear model predictive control of a U-loop bioreactor

Authors:
Tobias K. S. Ritschel, Dimitri Boiroux, Marcus Krogh Nielsen, Jakob Kjøbsted Huusom, Sten Bay Jørgensen, John Bagterp Jørgensen

Published in:
proceedings of the 2020 IEEE european control conference (ECC). May 12-15, 2020. Saint Petersburg, Russia.
DOI: `https://doi.org/10.23919/ECC51009.2020.9143629`

2020 European Control Conference (ECC)
May 12-15, 2020. Saint Petersburg, Russia

Economic Nonlinear Model Predictive Control of a U-loop Bioreactor

Tobias K. S. Ritschel, Dimitri Boiroux, Marcus Krogh Nielsen, Jakob Kjøbsted Huusom, Sten Bay Jørgensen, John Bagterp Jørgensen

Abstract— **In this paper, we present an algorithm for economic nonlinear model predictive control (NMPC) of single-cell protein production in a U-loop bioreactor. The model of the U-loop bioreactor consists of both stochastic ordinary and partial differential equations. Using a typical finite-volume discretization, the model contains 87 state variables. The NMPC algorithm is based on the continuous-discrete extended Kalman filter and a simultaneous collocation method. We present a closed-loop simulation which demonstrates the computational feasibility of real-time implementation of the NMPC algorithm for startup and steady state operation of the U-loop reactor.**

I. Introduction

Nonlinear model predictive control (NMPC) algorithms use the moving horizon optimization principle to compute a closed-loop feedback control strategy, i.e. they solve a sequence of open-loop optimal control problems (OCPs). The objective of NMPC algorithms is to 1) optimize a techno-economic performance measure or 2) track a set of predefined setpoints. NMPC involves strict computational requirements because it requires real-time solution of state estimation problems and OCPs. It is particularly challenging to meet such requirements when 1) the process involves both fast and slow time-scales (feedback must be fast while the control and prediction horizon must be long) and 2) the model of the process involves a large number of state variables, e.g. due to the discretization of partial differential equations (PDEs) in the model. Single-cell protein (SCP) production in the U-loop reactor studied in this paper is an example of such a process.

A. Nonlinear model predictive control

NMPC algorithms combine state estimation algorithms [1] with algorithms for solving OCPs [2]. It is common to use the extended Kalman filter (EKF) for state estimation of nonlinear processes [3]. Alternatives include the unscented Kalman filter (UKF), particle filters (PFs), the ensemble Kalman filter (EnKF) [4]–[6], and approaches based on the Fokker-Planck equation or hidden Markov models. The EKF and UKF are computationally tractable for models with many state variables. However, the remaining methods suffer from the curse of dimensionality [1], [6]–[11]. It is common to use direct methods to solve OCPs. In direct methods, the infinite-dimensional OCP is transcribed to a finite-dimensional nonlinear program (NLP) which can be solved using numerical optimization algorithms [12]. Direct methods include single-shooting, multiple-shooting [13], [14], and simultaneous collocation [15]. A key advantage of multiple-shooting and simultaneous collocation over single-shooting is their ability to handle unstable systems.

B. Single-cell protein production in the U-loop bioreactor

In the U-loop bioreactor, methanotrophs grow on cheap carbon sources, e.g. methane or methanol. The protein content in methanotrophs is high, and they can be used to produce SCP which is used for animal feed. Consequently, using the U-loop reactor to produce SCP can help sustain the growing human population. However, it is not trivial to operate the U-loop reactor because the optimal operating point is unstable (i.e. using an open-loop control strategy would lead to divergence from this operating point). Several authors have considered mathematical modeling of the dynamics of SCP production in the U-loop reactor [16]–[21], and Olsen et al. [22] computed optimal steady-state operating points based on a mathematical model. Furthermore, Ritschel et al. [23], [24] described the continuous-discrete EKF (CDEKF) and two simultaneous collocation-based approaches for state estimation and economic optimal control of the U-loop reactor, and Ritschel et al. [24] and Drejer et al. [25] presented economically optimal open-loop control strategies for the startup. However, NMPC of the U-loop reactor has not previously been considered.

C. This work

The key contribution of this work is an NMPC algorithm for economic optimizing control of the U-loop bioreactor. We use the model of the U-loop reactor described in [23]. It involves PDEs which are discretized in space using a finite-volume method. Consequently, the number of state variables in the model is large. Furthermore, the uncertainty of four of the variables and parameters in the model is represented using stochastic differential equations (SDEs). The NMPC algorithm uses the CDEKF and the simultaneous collocation method described in [23] and [24], respectively. Furthermore, we use a P-controller to ensure sufficiently fast feedback,

*This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 723661, Spire project: "Coordinating Optimisation of Complex Industrial Processes" (www.cocop-spire.eu). This work also received funding in the project EUDP 64013-0558 in the IEA annex for energy efficient process control. Tobias K. S. Ritschel, Dimitri Boiroux, Marcus Krogh Nielsen, and John Bagterp Jørgensen are with the Department of Applied Mathematics and Computer Science, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark. Jakob Kjøbsted Huusom and Sten Bay Jørgensen are with the Department of Chemical and Biochemical Engineering, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark. Tobias K. S. Ritschel is also with 2-control ApS, DK-2800 Kgs. Lyngby, Denmark. `jbjo@dtu.dk`

and the OCP is only solved once for every 20 control intervals (once every hour). Finally, we present a closed-loop simulation which demonstrates that real-time implementation of the NMPC algorithm for economic optimizing control of the startup and operation of the U-loop reactor is computationally feasible.

D. Paper organization

In Section II, we briefly describe the stochastic mathematical model of the U-loop reactor, and in Section III, we describe the numerical simulation of this stochastic model. In Section IV, we present the CDEKF, and we describe the OCP and the simultaneous collocation method in Section V. The P-controller is described in Section VI, and we present the closed-loop simulation in Section VII. Finally, conclusions are given in Section VIII.

II. Mathematical model

In this work, we consider a stochastic mathematical model of the U-loop reactor. The model is described in detail in [23]. In this section, we briefly discuss a few key aspects, as well as the general mathematical form, of the model. Fig. 1 shows a diagram of the U-loop reactor. It consists of 1) a pipe (referred to as the U-loop pipe) which is modeled as a plug-flow reactor (PFR) and 2) a top tank which is modeled as a continuous stirred-tank reactor (CSTR). In the U-loop reactor, methanol (CH_3OH), nitric acid (HNO_3), and dioxygen (O_2) is converted into biomass (X), carbon dioxide (CO_2), and water (H_2O):

$$\begin{aligned} &Y_{XS}\mathrm{CH_3OH} + Y_{XN}\mathrm{HNO_3} + Y_{XO}\mathrm{O_2} \rightarrow \\ &\qquad X + Y_{XC}\mathrm{CO_2} + Y_{XW}\mathrm{H_2O}. \end{aligned} \tag{1}$$

The feed substrate, water, and gaseous oxygen is mixed with recycled liquid (water, substrate, and dissolved oxygen) from the top tank before entering the U-loop pipe. The gaseous oxygen is partly dissolved in the liquid phase. At the end of the U-loop pipe, the gas phase is separated from the liquid phase which enters into the top tank. Finally, part of the liquid is harvested from the top tank, and the remaining part is recycled. The flow in the U-loop pipe is described by a set of PDEs, and the model of the top tank consists of a set of ordinary differential equations (ODEs). Both the PDEs and the ODEs are nonlinear, and they are derived using the principle of mass conservation. We use the method of lines (based on a standard finite volume discretization) to transform the PDEs to a set of ODEs. Furthermore, we use SDEs to represent the uncertainty in 1) three of the parameters in the production rates of the reaction (1) (the specific growth rate, $\mu_{\max}$, and the stoichiometric coefficients related to substrate and oxygen, γ_S and γ_O) and 2) the substrate feed concentration, $C_{F,S}$.

Consequently, the stochastic mathematical model of the U-loop reactor that we consider in this work is in the form

$$d\boldsymbol{x}(t) = f(\boldsymbol{x}(t), u(t))dt + \sigma(\boldsymbol{x}(t), u(t))d\boldsymbol{w}(t), \tag{2a}$$

$$\boldsymbol{z}(t) = g(\boldsymbol{x}(t)), \tag{2b}$$

$$\boldsymbol{y}(t_k) = \boldsymbol{z}(t_k) + \boldsymbol{v}(t_k). \tag{2c}$$

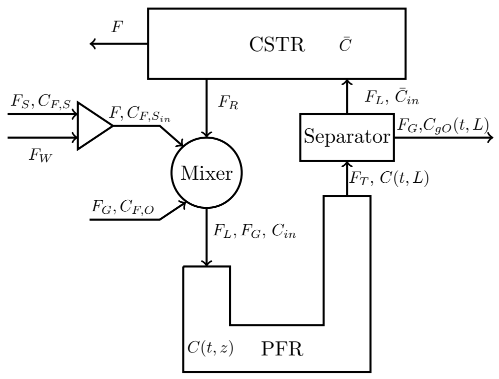


Fig. 1. Diagram of the U-loop reactor used in the mathematical modeling.

The state variables, $\boldsymbol{x}(t)$, are 1) the concentrations of biomass, substrate, dissolved oxygen, and gaseous oxygen in the U-loop pipe and 2) the concentrations of biomass, substrate, and dissolved oxygen in the top tank. The manipulated inputs, $u(t)$, are the feed flow rates of water, F_W, substrate, F_S, and gaseous oxygen, F_G, and $\boldsymbol{w}(t)$ is a standard Wiener process: $d\boldsymbol{w}(t) \sim N_{\mathrm{iid}}(0, Idt)$. The outputs, $\boldsymbol{z}(t)$, are the concentrations of dissolved oxygen at three positions along the U-loop pipe. Noisy measurements, $\boldsymbol{y}(t_k)$, of the outputs are obtained at discrete times, t_k. The measurement noise, $\boldsymbol{v}_k = \boldsymbol{v}(t_k)$, and the initial states, $\boldsymbol{x}(t_0)$ are normally distributed, i.e. $\boldsymbol{v}_k \sim N_{\mathrm{iid}}(0, R_v)$ and $\boldsymbol{x}(t_0) \sim N(\bar{x}_0, P_0)$.

The model (2) is used in the CDEKF, and (2a) is used in the OCP (where the stochastic term is neglected). The model does not include the dynamics related to the P-controller. Furthermore, the P-controller uses noisy measurements of the substrate concentration in the top tank.

III. Stochastic simulation

We use a semi-implicit method [26] to discretize the SDE (2a), i.e.

$$\begin{aligned} x_{k,n+1} - x_{k,n} &= f(x_{k,n+1}, u_{k,n})\Delta t_{k,n}^{\mathrm{pc}} \\ &\quad + \sigma(x_{k,n}, u_{k,n})\Delta w_{k,n}, \end{aligned} \tag{3}$$

for $n = 0, \ldots, M_k - 1$ where $\Delta w_{k,n}$ is a realization of $\Delta \boldsymbol{w}_{k,n} \sim N_{\mathrm{iid}}(0, I\Delta t_{k,n}^{\mathrm{pc}})$. For simplicity of notation, we consider one time step for each time the P-controller is used to compute the substrate feed flow rate (i.e. M_k time steps in the k'th control interval). $u_{k,n}$ denotes the manipulated inputs where the substrate feed flow rate is computed using the P-controller and the water and oxygen feed flow rates are given as the solution (in the k'th control interval) to the OCP (17) described in Section V. We use a variant of Newton-Raphson's method to solve (3) for the states, $x_{k,n+1}$.

IV. The continuous-discrete extended Kalman filter

The CDEKF for the continuous-discrete stochastic system (2) consists of 1) a measurement-update (i.e. a filter) and

2) a time-update (i.e. a one-step predictor). The CDEKF is initialized using the prior information which is represented by the distribution of the initial states: $\hat{x}_{0|-1} = \bar{x}_0$, and $P_{0|-1} = P_0$.

A. Measurement-update

At each discrete time, t_k, (i.e. whenever a set of measurements, y_k, become available) we assume that the one-step prediction of the states and the corresponding covariance, $\hat{x}_{k|k-1}$ and $P_{k|k-1}$, are available. First, we compute the one-step prediction of the measurements, $\hat{y}_{k|k-1} = \hat{z}_{k|k-1}$:

$$\hat{y}_{k|k-1} = g(\hat{x}_{k|k-1}). \tag{4}$$

Next, we compute the innovation and the corresponding covariance, i.e.

$$\begin{aligned} e_k &= y_k - \hat{y}_{k|k-1}, && (5a)\\ R_{e,k} &= C_k P_{k|k-1} C_k' + R_v, && (5b)\end{aligned}$$

where (5b) is obtained using linearization of the output equation (2b) and

$$C_k = \frac{\partial g}{\partial x}(\hat{x}_{k|k-1}). \tag{6}$$

The Kalman filter gain matrix is

$$K_{fx,k} = P_{k|k-1} C_k' R_{e,k}^{-1}. \tag{7}$$

Finally, we compute the filtered estimate of the states and the corresponding covariance matrix:

$$\begin{aligned} \hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_{fx,k} e_k, && (8a)\\ P_{k|k} &= P_{k|k-1} - K_{fx,k} R_{e,k} K_{fx,k}'. && (8b)\end{aligned}$$

B. Time-update

The one-step prediction of the states is obtained by solving the initial value problem

$$\frac{d}{dt}\hat{x}_k(t) = f(\hat{x}_k(t), u(t)), \qquad \hat{x}_k(t_k) = \hat{x}_{k|k}. \tag{9}$$

In the computation of the covariance of the one-step prediction, we use the sensitivities of the one-step prediction of the states at time t with the respect to the states at time s, $\Phi_k(t,s) = \frac{\partial \hat{x}_k(t)}{\partial \hat{x}_k(s)}$, which satisfies the initial value problem

$$\frac{d}{dt}\Phi_k(t,s) = A_k(t)\Phi_k(t,s), \qquad \Phi_k(s,s) = I, \tag{10}$$

where

$$A_k(t) = \frac{\partial f}{\partial x}(\hat{x}_k(t), u(t)). \tag{11}$$

The covariance at time t is [3]

$$\begin{aligned} P_k(t) = {}& \Phi_k(t,t_k) P_{k|k} \Phi_k(t,t_k)' \\ &+ \int_{t_k}^{t} \Phi_k(t,s)\sigma_k(s)\sigma_k(s)'\Phi_k(t,s)' ds, \end{aligned} \tag{12}$$

where

$$\sigma_k(s) = \sigma(\hat{x}_k(s), u(s)). \tag{13}$$

1) Numerical solution: We use Euler's implicit method to discretize the ODE in (9), i.e.

$$\hat{x}_{k,n+1} - \hat{x}_{k,n} = f(\hat{x}_{k,n+1}, u_{k,n})\Delta t_{k,n}^{\text{pc}}, \tag{14}$$

for $n = 0, \ldots, M_k - 1$ where $\hat{x}_{k,0} = \hat{x}_{k|k}$. As in Section III, we consider one time step for each time the P-controller is used to update the substrate feed flow rate, and we use a variant of Newton-Raphson's method to solve (14) for $\hat{x}_{k,n+1}$. We also use Euler's implicit method to discretize the ODE in (10), i.e.

$$\begin{aligned} &\Phi_k(t_{k,n+1}, t_{k,n}) - \Phi_k(t_{k,n}, t_{k,n}) \\ &\qquad = A_k(t_{k,n+1})\Phi_k(t_{k,n+1}, t_{k,n})\Delta t_{k,n}^{\text{pc}}, \end{aligned} \tag{15}$$

where $\Phi_k(t_{k,n}, t_{k,n}) = I$. We use a left rectangle rule to approximate the integral in (12) such that

$$\begin{aligned} P_{k,n+1} &= \Phi_k(t_{k,n+1}, t_{k,n})\Lambda_{k,n}\Phi_k(t_{k,n+1}, t_{k,n})', && (16a)\\ \Lambda_{k,n} &= P_{k,n} + \sigma(\hat{x}_{k,n}, u_{k,n})\sigma(\hat{x}_{k,n}, u_{k,n})'\Delta t_{k,n}^{\text{pc}}, && (16b)\end{aligned}$$

where $P_{k,0} = P_{k|k}$. The one-step prediction of the states and the covariance are $\hat{x}_{k+1|k} = \hat{x}_{k,M_k}$ and $P_{k+1|k} = P_{k,M_k}$.

V. Optimal control

At the beginning of every N_p'th control interval (i.e. at time t_k for $k = 0, N_p, 2N_p$, etc.), we solve the OCP

$$\min_{[x(t)]_{t_k}^{t_{k+N}}, \{u_{k+j|k}\}_{j=0}^{N-1}} \quad \phi = \phi\left([x(t); u(t)]_{t_k}^{t_{k+N}}\right), \tag{17a}$$

subject to

$$\begin{aligned} & x(t_k) = \hat{x}_{k|k}, && (17b)\\ & \dot{x}(t) = f(x(t), u(t)), \quad t \in [t_k, t_{k+N}], && (17c)\\ & u(t) = u_{k+j|k}, \quad t \in [t_{k+j}, t_{k+j+1}[, \quad j = 0, \ldots, N-1, && (17d)\\ & u^{\min} \leq u_{k+j|k} \leq u^{\max}, \quad j = 0, \ldots, N-1. && (17e)\end{aligned}$$

The prediction and control horizon is N control intervals, the states, $[x(t)]_{t_k}^{t_{k+N}}$, are dependent decision variables, and the manipulated inputs, $\{u_{k+j|k}\}_{j=0}^{N-1}$, are independent decision variables. The initial states in the initial condition (17b) are the filtered estimate of the states (8a) computed in the CDEKF. In the OCP, we assume that there is no process noise in the dynamic model, i.e. (17c) is deterministic. (17d) is a zero-order-hold (ZOH) parametrization of the manipulated inputs, and (17e) are bounds on the manipulated inputs. The objective function is in Bolza form:

$$\begin{aligned} \phi &= \phi\left([x(t); u(t)]_{t_k}^{t_{k+N}}\right) \\ &= \int_{t_k}^{t_{k+N}} l(x(t), u(t))dt + l_f(x(t_f), u(t_f)) \\ &\quad + \int_{t_k}^{t_{k+N}} l_s(x(t))dt. \end{aligned} \tag{18}$$

The first term is the integral of the stage-cost, the second term is the cost-to-go (where $u(t_f) = u_{k+N-1|k}$), and the third term represents soft bound constraints on the state variables, $x^{\min} \leq x(t) \leq x^{\max}$ for $t \in [t_k, t_{k+N}]$, where

$$l_s(x(t)) = \sum_{i=1}^{n_x} \Big(\alpha_i^{\max} \max\{0, x_i(t) - x_i^{\max}\}^2 + \alpha_i^{\min} \max\{0, x_i^{\min} - x_i(t)\}^2 \Big). \quad (19)$$

n_x is the number of state variables, and $\alpha_i^{\max}$ and $\alpha_i^{\min}$ are weights.

A. Simultaneous collocation

We use Euler's implicit method to discretize the dynamical constraints (17c), i.e.

$$x_{k+j,n+1} - x_{k+j,n} = f(x_{k+j,n+1}, u_{k+j|k})\Delta t_{k+j,n}, \quad (20)$$

where $n = 0, \dots, N_{k+j} - 1$, and $j = 0, \dots, N - 1$. We formulate (20) as $R_{k+j,n} = 0$ where

$$\begin{aligned} R_{k+j,n} &= R_{k+j,n}(x_{k+j,n+1}, x_{k+j,n}, u_{k+j|k}) \\ &= x_{k+j,n+1} - x_{k+j,n} \\ &\quad - f(x_{k+j,n+1}, u_{k+j|k})\Delta t_{k+j,n}. \end{aligned} \quad (21)$$

Furthermore, the states must be continuous at the boundaries of the control intervals:

$$x_{k,0} = \hat{x}_{k|k}, \quad (22a)$$

$$x_{k+j+1,0} = x_{k+j,N_{k+j}}, \quad j = 0, \dots, N-2. \quad (22b)$$

We substitute the discretized differential equations (21) and the continuity conditions (22) for the initial value problem (17b)-(17c) and the ZOH parametrization (17d) to obtain the transcribed NLP

$$\min_{\{\{x_{k+j,n}\}_{n=0}^{N_{k+j}}\}_{j=0}^{N-1}, \{u_{k+j|k}\}_{j=0}^{N-1}} \quad \psi, \quad (23a)$$

subject to

$$x_{k,0} = \hat{x}_{k|k}, \quad (23b)$$

$$x_{x+j+1,0} = x_{k+j,N_{k+j}}, \quad j = 0, \dots, N-2, \quad (23c)$$

$$R_{k+j,n}(x_{k+j,n+1}, x_{k+j,n}, u_{k+j|k}) = 0, \quad n = 0, \dots, N_{k+j} - 1, \quad j = 0, \dots, N-1, \quad (23d)$$

$$u^{\min} \leq u_{k+j|k} \leq u^{\max}, \quad j = 0, \dots, N-1. \quad (23e)$$

We use a right rectangle rule to discretize the integrals in the objective function (19), and we add a rate-of-movement penalization term (the last term):

$$\begin{aligned} \psi &= \psi\left(\{\{x_{k+j,n}\}_{n=0}^{N_{k+j}}\}_{j=0}^{N-1}, \{u_{k+j|k}\}_{j=0}^{N-1}\right) \\ &= \sum_{j=0}^{N-1} \sum_{n=0}^{N_{k+j}-1} l(x_{k+j,n+1}, u_{k+j|k})\Delta t_{k+j,n} \\ &\quad + l_f(x_{k+N-1,N_{k+N-1}}, u_{k+N-1|k}) \\ &\quad + \sum_{j=0}^{N-1} \sum_{n=0}^{N_{k+j}-1} l_s(x_{k+j,n+1})\Delta t_{k+j,n} \\ &\quad + \sum_{j=0}^{N-1} \frac{1}{\Delta t_{k+j}} \|\Delta u_{k+j|k}\|_{R_{\Delta u}}^2. \end{aligned} \quad (24)$$

The changes in the manipulated inputs are given by

$$\Delta u_{k|k} = u_{k|k} - u_{k-1|k-1}, \quad (25a)$$

$$\Delta u_{k+j|k} = u_{k+j|k} - u_{k+j-1|k}, \quad j = 1, \dots, N-1. \quad (25b)$$

VI. P-Controller

For $n = 0, \dots, M_k - 1$ (in the k'th control interval), we use a P-controller to compute the substrate feed flow rate, F_S, in the time interval $[t_{k,n}^{\mathrm{pc}}, t_{k,n+1}^{\mathrm{pc}}[$:

$$\tilde{F}_S = F_S^* + K_{cS}(\bar{C}_S^* - \bar{C}_S^m), \quad (26a)$$

$$F_S = \max\{0, \min\{F_S^{\max}, \tilde{F}_S\}\}. \quad (26b)$$

$\bar{C}_S^m = \bar{C}_S + v_S$ is a noisy measurement of the substrate concentration in the top tank (at time $t_{k,n}^{\mathrm{pc}}$), $\bar{C}_S$, where $v_S \sim N(0, R_S)$. F_S^* and $\bar{C}_S^*$ denote the optimal substrate feed flow rate (in the k'th control interval) and the optimal biomass concentration (at time t_{k,N_k}), i.e. they are part of the most recently obtained solution of the OCP (17) which is solved once for every N_p control intervals. K_{cS} is the proportional gain, and $F_s^{\max}$ is the maximum substrate feed flow rate.

VII. Numerical Example

In this section, we present a closed-loop simulation of economic NMPC of the U-loop reactor over a period of 72 h. We discretize the PDEs in the model using 20 finite volumes. Consequently, the model contains 87 state variables. The computations are carried out on an Ubuntu 16.04 LTS 64-bit workstation with eight Intel Core i7 3.60 GHz processors and 15.6 GB RAM.

A. Stochastic simulation of the U-loop reactor

We use a stochastic simulation to represent the true U-loop reactor in the closed-loop simulation. The initial biomass concentration is 0.1 kg/m^3 (both in the top tank and in each finite volume in the discretized U-loop pipe). The remaining initial concentrations are 0 kg/m^3. The volume of the U-loop pipe is 1.068 m^3, and the volume of the top tank is 0.25 m^3. The U-loop pipe is 12 m long, and its diameter is 0.089 m^2. The concentrations of the dissolved oxygen at 4 m, 7 m, and 10 m along the U-loop pipe (corresponding to the 7'th, 12'th, and 17'th finite volumes) are measured every third minute. The substrate concentration in the top tank is measured every minute. For all four measured concentrations, the variance of the measurement noise is 10^{-4} (kg/m^3)^2.

B. The NMPC algorithm

The prediction and control horizon in the NMPC algorithm is 30 h, and the control intervals are 3 min each. We use the economic objective function described in [24], [25] which represents the difference between the value of the produced biomass (40 \$/kg) in the entire U-loop reactor and the cost of raw materials (approximated by the cost of substrate, 0.34 \$/kg, and oxygen, 3 \$/kg). The model of the U-loop reactor used in this work is not realistic for very high biomass concentrations. Therefore, we impose a soft upper constraint of 30 kg/m^3 on all biomass concentrations. Furthermore, we impose a lower bound of 0.05 kg/m^3 on all concentrations. Finally, we solve the OCP (17) once every hour (i.e. $N_p = 20$). The P-controller updates the substrate feed flow rate every minute. The CDEKF uses 1) the initial condition described above as the initial state estimate and 2) a diagonal initial estimate of the covariance, P_0, where all diagonal elements are 10^{-2} (the units vary).

C. Closed-loop simulation

Fig. 2 shows 1) the concentrations of biomass, substrate, and dissolved oxygen in the top tank, and 2) the manipulated inputs. The NMPC algorithm successfully operates the U-loop reactor during startup such that steady-state operation is reached after 30 h. The CDEKF accurately estimates the concentrations of biomass and dissolved oxygen in the top tank. However, the difference between the estimated and the true substrate concentration is often relatively high. Furthermore, the P-controller makes significant changes to the optimized substrate feed flow rate in order to maintain the substrate concentration at the optimized value. Fig. 3 shows the estimated and the true maximum specific growth rate, $\mu_{\max}$, and stoichiometric coefficients for substrate and oxygen, γ_S and γ_O, which are estimated reasonably well. Furthermore, it shows the estimated substrate feed concentration which is almost equal to its nominal value, i.e. it is not estimated accurately. Fig. 4 shows the true production rate, harvest rate, and profit rate. During the startup, more biomass is produced than harvested, i.e. it is accumulated inside the reactor. This is desirable because the biomass production rate is proportional to the biomass concentration. During the subsequent steady-state operation, roughly all of the produced biomass is harvested, and the profit rate is around 380 \$/h. Fig. 5 shows a histogram of the computation times of solving the OCP (17) (excluding the first OCP which can be solved offline). The majority of the computation times are less than 45 s, and the worst-case computation time is 70 s. This demonstrates that real-time implementation of the NMPC algorithm is computationally feasible.

VIII. Conclusions

In this paper, we consider economic NMPC of SCP production in the U-loop reactor. We consider a dynamical model of the U-loop reactor which consists of ODEs and PDEs, and we use SDEs to represent four sources of uncertainty. We discretize the PDEs using a finite-volume method. The resulting stochastic model contains 87 state variables. We present an NMPC algorithm based on the CDEKF and a simultaneous collocation method (both developed in previous work). Furthermore, we combine the NMPC algorithm with a stabilizing P-controller. Finally, we present a closed-loop simulation which demonstrates that 1) the NMPC algorithm is able to successfully control the U-loop reactor during both startup and steady-state operation, and 2) real-time implementation of the NMPC algorithm is computationally feasible (the worst-case computation time per control interval is 70 s).

References

[1] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[2] T. Binder, L. Blank, H. G. Bock, R. Bulirsch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J. P. Schlöder, and O. von Stryk, "Introduction to model based optimization of chemical processes on moving horizons," in *Online Optimization of Large Scale Systems*. Springer-Verlag Berlin Heidelberg, 2001, pp. 295–339.

[3] J. B. Jørgensen, M. R. Kristensen, P. G. Thomsen, and H. Madsen, "New extended Kalman filter algorithms for stochastic differential algebraic equations," in *Assessment and Future Directions of Nonlinear Model Predictive Control*, ser. Lecture Notes in Control and Information Sciences. Springer-Verlag Berlin Heidelberg, 2007, vol. 358, pp. 359–366.

[4] G. Evensen, *Data assimilation: the ensemble Kalman filter*, 2nd ed. Springer-Verlag Berlin Heidelberg, 2009.

[5] ——, "The ensemble Kalman filter for combined state and parameter estimation," *IEEE Control Systems Magazine*, vol. 29, no. 3, pp. 83–104, 2009.

[6] S. Gillijns, O. B. Mendoza, J. Chandrasekar, B. L. R. De Moor, D. S. Bernstein, and A. Ridley, "What is the ensemble Kalman filter and how well does it work?" in *Proceedings of the 2006 American Control Conference*, Minneapolis, Minnesota, USA, June 2006, pp. 4448–4453.

[7] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*. San Diego, CA, USA: Academic Press, 1970.

[8] A. Gelb, *Applied Optimal Estimation*. MIT Press, 1974.

[9] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear Estimation*. Prentice Hall, 2000.

[10] A. M. Fraser, *Hidden Markov Models and Dynamical Systems*. SIAM, 2008.

[11] J. B. Rawlings, D. Q. Mayne, and M. M. Diehl, *Model predictive control: theory, computation, and design*, 2nd ed. Nob Hill Publishing, 2017.

[12] J. Nocedal and S. J. Wright, *Numerical optimization*, 2nd ed. Springer Science & Business Media, 2006.

[13] H. G. Bock and K. J. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems," *IFAC Proceedings Volumes*, vol. 17, no. 2, pp. 1603–1608, 1984.

[14] A. Schäfer, P. Kühl, M. Diehl, J. Schlöder, and H. G. Bock, "Fast reduced multiple shooting methods for nonlinear model predictive control," *Chemical Engineering and Processing: Process Intensification*, vol. 46, no. 11, pp. 1200–1214, 2007.

[15] L. T. Biegler, "An overview of simultaneous strategies for dynamic optimization," *Chemical Engineering and Processing: Process Intensification*, vol. 46, no. 11, pp. 1043–1053, 2007.

[16] D. F. Olsen, J. B. Jørgensen, J. Villadsen, and S. B. Jørgensen, "Modeling and simulation of single cell protein production," *IFAC Proceedings Volumes*, vol. 43, no. 6, pp. 502–507, 2010.

[17] O. A. Prado-Rubio, J. B. Jørgensen, and S. B. Jørgensen, "Systematic model analysis for single cell protein (SCP) production in a U-loop reactor," *Computer Aided Chemical Engineering*, vol. 28, pp. 319–324, 2010.

[18] A. M. Al Taweel, Q. Shah, and B. Aufderheide, "Effect of mixing on microorganism growth in loop bioreactors," *International Journal of Chemical Engineering*, no. Article ID 984827, 2012.

[19] M. Wu, J. K. Huusom, K. V. Gernaey, and U. Krühne, "Modelling and simulation of a U-loop reactor for single cell protein production," *Computer Aided Chemical Engineering*, vol. 38, pp. 1287–1292, 2016.

[20] L. A. H. Petersen, J. Villadsen, S. B. Jørgensen, and K. V. Gernaey, "Mixing and mass transfer in a pilot scale U-loop bioreactor," *Biotechnology and Bioengineering*, vol. 114, no. 2, pp. 344–354, 2017.

[21] L. A. H. Petersen, C. Lieven, S. K. Nandy, J. Villadsen, S. B. Jørgensen, I. Christensen, and K. V. Gernaey, "Dynamic investigation and modeling of the nitro cometabolism in Methylococcus capsulatus (Bath)," *Biotechnology and Bioengineering*, 2019.

[22] D. F. Olsen, J. B. Jørgensen, J. Villadsen, and S. B. Jørgensen, "Optimal operating points for SCP production in the U-loop reactor," *IFAC Proceedings Volumes*, vol. 43, no. 5, pp. 499–504, 2010.

[23] T. K. S. Ritschel, D. Boiroux, M. K. Nielsen, J. K. Huusom, S. B. Jørgensen, and J. B. Jørgensen, "The extended Kalman filter for nonlinear state estimation in a U-loop bioreactor," in *Proceedings of the 3rd IEEE Conference on Control Technology and Applications*, Hong Kong, China, 2019.

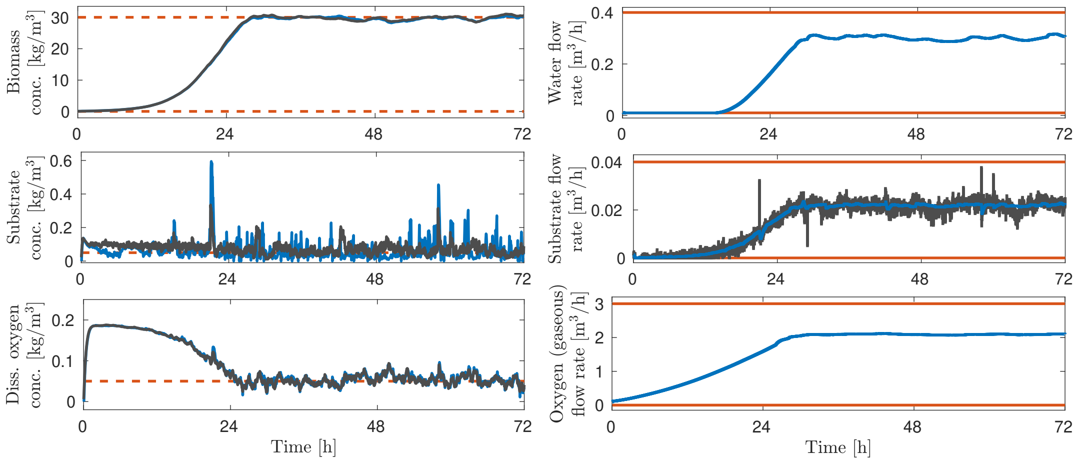


Fig. 2. Left column: Filtered estimates of the concentrations in the top tank in the U-loop reactor obtained with the CDEKF (blue solid) together with the true concentrations (black solid) and the soft bounds (red dashed). Right column: The water, substrate, and gaseous oxygen feed flow rates computed by solving the OCP (17) (blue solid), the substrate feed flow rate computed using the P-controller (black solid), and the hard bounds (red solid). The filtered estimates of the biomass concentration and the dissolved oxygen concentration are almost indistinguishable from their true values.

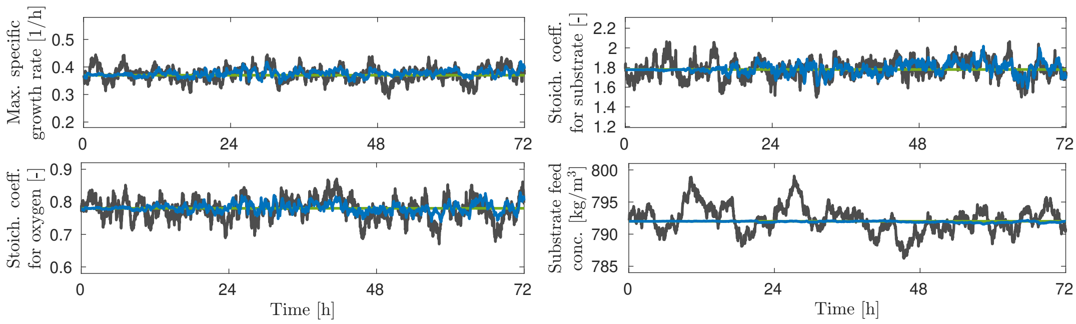


Fig. 3. Filtered estimates (blue solid) of the maximum specific growth rate, $\mu_{\max}$, the stoichiometric coefficients for substrate and oxygen, γ_S and γ_O, and the substrate feed concentration, $C_{F,S}$, together with the true (black solid) and nominal (green) values.

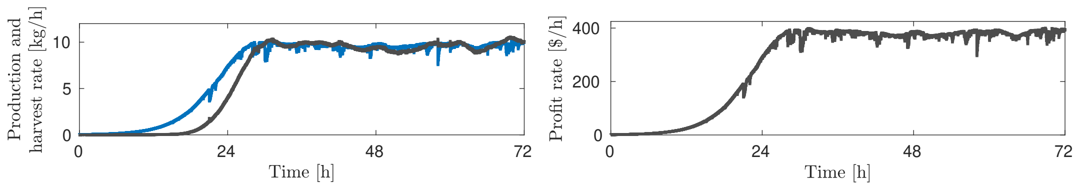


Fig. 4. Left: production rate (blue solid) and harvest rate (black solid). Right: profit rate.

[24] ——, "Economic optimal control of a U-loop bioreactor using simultaneous collocation-based approaches," in *Proceedings of the 3rd IEEE Conference on Control Technology and Applications*, Hong Kong, China, 2019.

[25] A. Drejer, T. Ritschel, S. B. Jørgensen, and J. B. Jørgensen, "Economic optimizing control for single-cell protein production in a U-loop reactor," *Computer Aided Chemical Engineering*, vol. 40, pp. 1759–1764, 2017.

[26] K. Burrage and T. Tian, "The composite Euler method for stiff stochastic differential equations," *Journal of Computational and Applied Mathematics*, vol. 131, pp. 407–426, 2001.

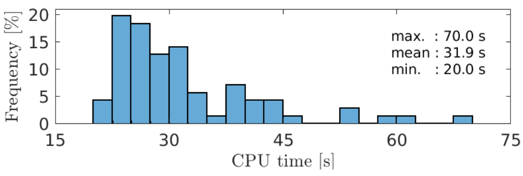


Fig. 5. Histogram of the computation times of solving the OCP (17).

APPENDIX G

Technical Report I

Modelling of Reactive Systems

Authors:
Marcus Krogh Nielsen and John Bagterp Jørgensen

Submitted to:
Arxiv, at https://arxiv.org/

Technical Report

Modelling of Reactive Systems

Stoichiometry, Kinetics, and Reactors

Marcus Krogh Nielsen
John Bagterp Jørgensen

Department of Applied Mathematics and Computer Science,
Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark
Unibio A/S, DK-4000 Roskilde, Denmark

DTU Compute
Department of Applied Mathematics and Computer Science

DTU Compute
Department of Applied Mathematics and Computer Science
Technical University of Denmark

Matematiktorvet
Building 303B
2800 Kongens Lyngby, Denmark
Phone +45 4525 3031
compute@compute.dtu.dk
www.compute.dtu.dk

Summary

In this technical report, we describe a framework for modelling of reactive systems. We introduce the stoichiometric matrix, describing the stoichiometry of a reactive system. We introduce the rate of reaction vector, describing the rate at which the reactions happen. We introduce the production vector for reactive systems, defined as a function of the stoichiometric matrix and the reaction rates. We describe and derive models for four reactor types; batch reactors, fedbatch reactors, continuous stirred tank reactor, and plug flow reactors. We apply the modelling framework, defining stoichiometric matrices and rates of reaction, to two example reactive systems; an exothermic chemical reaction in an adiabatic environment and a fermentation for single-cell protein production. Finally, we present numerical examples for the two example reactive systems in the four presented reactor types; batch, fedbatch, continuous stirred tank, and plug flow reactors.

The appendix introduces gas-liquid mass transfer dynamics for systems involving both liquid and gas phases. Additionally, the appendix describes Jacobian wrt. states and inputs as well as model implementations in Matlab and Python for the four presented reactor types.

Executive Summary

Objective

- Describe a general modelling framework for reactive systems.
- Present models describing batch, fedbatch, continuous stirred tank, and plug flow reactors.
- Apply the modelling framework in numerical experiments on example reactive systems.

Reactive Systems

- Introduce and formulate the stoichimetric matrix for reactive systems.
- Introduce and formulate the rate of reaction vector for reactive systems.
- Describe kinetic models describing chemical systems.
- Describe kinetic models describing biochemical systems, i.e. fermentation.

Batch Reactors

- Introduce batch reactors.
- Derive an ordinary differential equation model based on mass balances.

Fedbatch Reactors

- Introduce fedbatch reactors.
- Derive an ordinary differential equation model based on mass balances.

Continuous Stirred Tank Reactors

- Introduce continuous stirred tank reactors.
- Derive an ordinary differential equation model based on mass balances in the variable volume case.
- Derive an ordinary differential equation model based on mass balances in the constant volume case.

Plug Flow Reactors

- Introduce plug flow reactors.
- Derive a partial differential equation model based on mass balances.
- Derive an ordinary differential equation model based on a finite-volume discretisation of the model.

Numerical Examples

- Apply the modelling framework to an exothermic chemical reaction in an adiabatic environment.
- Perform numerical experiments for the chemical reaction example in batch, fed-batch, continuous stirred tank, and plug flow reactors.
- Apply the modelling framework to a fermentation for single-cell protein production.
- Perform numerical experiments for the fermentation example in batch, fedbatch, continuous stirred tank, and plug flow reactors.

Appendix

- Describe gas-liquid mass transfer for reactive systems involving both liquid and gas phases.
- Present Jacobians wrt. states and inputs for four reactor types.
- Present implementations in Matlab and Python for four reactor types.

Contents

CHAPTER 1

Introduction

In this technical report, we describe a modelling framework for reactive systems. The technical report is structured as follows: In Chapter 2, we describe a modelling framework for reactive systems defined by stoichiometry and reaction rates. We introduce the stoichiometric matrix, for defining general reactive systems. We introduce the rate of reaction vector, describing reaction rates for each of the stoihciometric reactions defined in the stoichiometric matrix. We define production rates for each components as a function of the stoichiometric matrix and the reaction rates. Finally, we present kinetic models defining the rate of reaction in both chemical and fermentation processes.

In Chapters 3-6, we present descriptions and models stirred tank reactors (STRs) and plug flow reactors (PFRs). STRs describe a class of ideally mixed (well-mixed) reactors, where all points in the reactors are assumed to have the same characteristics, e.g. temperature, pressure, and concentrations. In this technical report, we consider STRs in batch operation (BR), in fed-batch operation (FBR), and in non-steady-state continuous operation (CSTR). Finally, we consider the non-steady-state PFR. The reactors are illustrated in Figure 1.1.

An STR in batch operation is characterised by a single closed vessel with no in- or outflows of reactants and products during operation. In BRs, the reactor volume is supplied with reactants at the prior to operation time in accordance with a set of procedures (recipe), e.g. arising from scale-up of laboratory-scale experiments [13]. As such, it is the operating conditions, e.g. temperature, pressure, etc. which are manipulated during operation of a BR, as oppose to the in- and outflow of reactants and products. In BRs, products are typically extracted from the reactor once the reaction is complete, i.e. the concentrations of reactants are zero. Products extracted from BRs are easily identifiable due to the uniformity of the product extracted from a particular batch. This makes BRs ideal for production of foods, beverages, drug and other specialty chemicals. The identifiability makes it possible to withdraw only a single batch if a problem is identified instead of an entire product [3, 14]. Chapter 3 present first-principle ordinary differential equation (ODE) models describing masses or concentrations. The models are derived from mass balances. Appendix B describes the Jacobian wrt. the states, as well as implementations of a BR model in Matlab and Python.

Fed-batch (or semi-batch) reactors describes STRs where reactants are continuously supplied to the reactor during operation. FBRs share some of the same advantages of BRs in batch identifiability, etc., but introduce operational complexities in the continuous (or discrete) addition of reactants. The fed-batch process was developed during the 1920s for more efficient production of baker's yeast [14]. One of the advantages that FBRs have other BRs, is when high concentrations of some reactants have an inhibiting

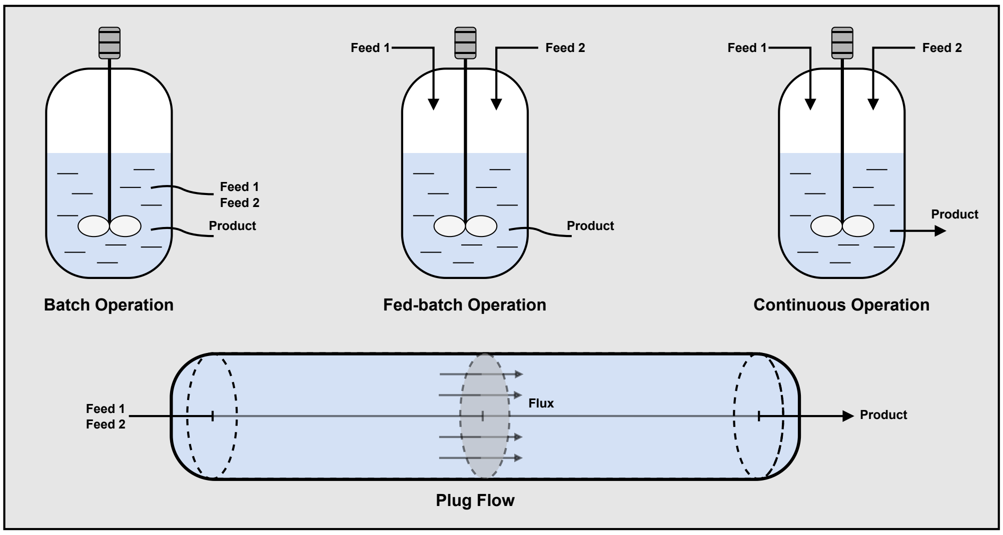


Figure 1.1: Illustration of stirred tank (top) and plug flow reactors (bottom). Stirred tank reactors are illustrated in batch operation (top left), fed-batch operation (top middle), and continuous operation (top right). In batch operation, reactants (feed 1 and 2) are not added during operation. In fed-batch operation, reactants (feed 1 and 2) are added during operation. In continuous operation, reactants (feed 1 and 2) and products flow continuously in and out of the reactor..

(or toxic) effect on the productivity. In fed-batch operation, the concentrations of reactants can be stabilised at concentrations which are non-inhibiting to the productivity of the process [9, 4]. Chapter 4 present first-principle ODE models describing volume and mass or concentrations. The models are derived from volume and mass balances. Appendix C describes Jacobians wrt. states and inputs, as well as implementations of an FBR model in Matlab and Python.

Continuous operation of STRs is characterised by continuous feed (inflow) of reactants, as well as continuous outflow of product (reactor fluid). Traditionally, CSTRs have been applied in industry, but most commonly operated at steady state [3]. Non-steady-state operation of the CSTR require the study of transient behaviour and thus presents additional operation challenges [5, 15]. However, since start-up necessarily involved such transients, it is necessary to understand the behaviour [14]. Chapter 5 present first-principles ODE models describing volume and mass or concentrations. The models are derived from volume and mass balances. Appendix D describes Jacobians wrt. states and inputs, as well as implementations of a CSTR model in Matlab and Python.

The PFR is another example of a reaction in continuous operation. PFRs are pipe reactors, in which reactants flow from the inlet through the pipe to the pipe outlet where products are harvested. PFRs are commonly applied in the biochemical and pharmaceutical industries for microbial growth and product clean-up, e.g. waste-water treatment and monoclonal antibody production [7, 6]. Chapter 6 presents a first-principle partial differential equation (PDE) model describing concentrations. The model is based on mass balances. Additionally, we describe an ODE model based on a finite-volume discretisation of the PDE model. Appendix E describes Jacobians wrt. states and inputs, as well as implementation of the ODE model of the PFR in Matlab and Python.

In Chapter 7, we present numerical examples. We present an exothermic chemical reaction example and a fermentation example of single-cell protein production. For the example reactive systems, we apply the modelling framework, i.e. define the stoichiometric matrix, kinetics, reaction rates, and productions. We implement both examples of reactive systems in all four reactor types; BR, FBR, CSTR with constant volume, and the finite-volume discretisation of the PFR. For each of the examples, we also present Jacobians of the rate of the reaction vectors wrt. the concentrations.

In Appendix A, we present methods of describing gas-liquid mass transfer in reactions where both liquids and gasses are present in the reactors, e.g. fermentation processes involving aerobic and/or methanotrophic microorganisms.

Chapter 8 presents conclusions.

CHAPTER 2

Reactive Systems

In this section, we present a framework for modelling reactive systems. We introduce the stoichiometric matrix, defining the system stoichiometry in terms of the stoichiometric coefficients. We introduce the rate of reaction vector, describing the rate at which the reactions occur in the presence of reactants and products. We introduce the production vector, describing the production or consumption of chemical components in the system. Finally, we present kinetic models for chemical and biochemical systems.

2.1 Modelling Framework

Consider a general reactive system of N reactions with M components in the form

$$\mathrm{a}_{1,1}\,\mathrm{C}_1 + \mathrm{a}_{1,2}\,\mathrm{C}_2 + \cdots + \mathrm{a}_{1,\mathrm{M}}\,\mathrm{C}_\mathrm{M} \longrightarrow \mathrm{b}_{1,1}\,\mathrm{C}_1 + \mathrm{b}_{1,2}\,\mathrm{C}_2 + \cdots + \mathrm{b}_{1,\mathrm{M}}\,\mathrm{C}_\mathrm{M}, \qquad r_1, \tag{2.1a}$$

$$\mathrm{a}_{2,1}\,\mathrm{C}_1 + \mathrm{a}_{2,2}\,\mathrm{C}_2 + \cdots + \mathrm{a}_{2,\mathrm{M}}\,\mathrm{C}_\mathrm{M} \longrightarrow \mathrm{b}_{2,1}\,\mathrm{C}_1 + \mathrm{b}_{2,2}\,\mathrm{C}_2 + \cdots + \mathrm{b}_{2,\mathrm{M}}\,\mathrm{C}_\mathrm{M}, \qquad r_2, \tag{2.1b}$$

$$\vdots$$

$$\mathrm{a}_{\mathrm{N},1}\,\mathrm{C}_1 + \mathrm{a}_{\mathrm{N},2}\,\mathrm{C}_2 + \cdots + \mathrm{a}_{\mathrm{N},\mathrm{M}}\,\mathrm{C}_\mathrm{M} \longrightarrow \mathrm{b}_{\mathrm{N},1}\,\mathrm{C}_1 + \mathrm{b}_{\mathrm{N},2}\,\mathrm{C}_2 + \cdots + \mathrm{b}_{\mathrm{N},\mathrm{M}}\,\mathrm{C}_\mathrm{M}, \qquad r_N. \tag{2.1c}$$

We describe the stoichiometry in the reactive system via the stoichiometric matrix

$$S = \begin{array}{c} \begin{array}{cccc} \mathrm{C}_1 & \mathrm{C}_2 & \cdots & \mathrm{C}_M \end{array} \\ \left[\begin{array}{cccc} b_{1,1} - a_{1,1} & b_{1,2} - a_{1,2} & \cdots & b_{1,M} - a_{1,M} \\ b_{2,1} - a_{2,1} & b_{2,2} - a_{2,2} & \cdots & b_{2,M} - a_{2,M} \\ \vdots & \vdots & \ddots & \vdots \\ b_{N,1} - a_{N,1} & b_{N,2} - a_{N,2} & \cdots & b_{N,M} - a_{N,M} \end{array}\right] \end{array} \begin{array}{c} \\ \mathrm{r}_1 \\ \mathrm{r}_2 \\ \vdots \\ \mathrm{r}_N \end{array} . \tag{2.2}$$

Rows and columns in the matrix represent reactions and chemical components, respectively. As such, elements in the stoichiometric matrix are differences between the amount produced and consumed for all components in all reactions. We write the reaction rates as the vector function

$$r(c) = \begin{bmatrix} r_1(c) \\ r_2(c) \\ \vdots \\ r_N(c) \end{bmatrix}, \qquad c = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_M \end{bmatrix}, \tag{2.3}$$

where $c_j = [C_j]$ are the concentrations of component $j \in \mathcal{C}$ and where $\mathcal{C} = \{C_1, C_2, \cdots, C_M\}$ is the set of chemical components involved in the reactions. We describe the production of each of the chemical components in terms of the stoichiometric matrix and reaction rates, as

$$R(c) = S^T r(c). \tag{2.4}$$

2.2 Rate of reaction

In this section, we describe kinetic models for chemical and biochemical systems. For chemical systems, we present reaction kinetics involving partial and complete reactions, as well as describing kinetics in systems with temperature variations influencing the reaction rate. For biochemical systems, we present growth kinetics for uninhibited, substrate inhibited, and product inhibited growth in the microorganism, i.e. rate at which metabolic reactions take place in the cell, e.g. cell growth or production of co-metabolic compounds.

2.2.1 Chemical processes

In chemical reactions, on the form (2.1), we describe the reaction rate for each reaction, r_i for $i \in \{1, 2, \dots, N\}$, as

$$r_i(c) = k(T) \prod_{j=1}^{M} c_j^{a_{i,j}} - k_r(T) \prod_{j=1}^{M} c_j^{b_{i,j}}, \tag{2.5}$$

where $c_j = [C_j]$ is the concentrations of C_j. For complete reactions, i.e. reactions where $k_r(T)$ is negligible, we describe the reaction rates as

$$r_i(c) = k(T) \prod_{j=1}^{M} c_j^{a_{i,j}}. \tag{2.6}$$

In this case, the reaction rate coefficient, $k(T)$, describes the effect temperature has on the rate of reaction.

Arrhenius expression
We describe temperature dependence by the Arrhenius expression

$$k(T) = k_0 \exp\left(-\frac{E_a}{RT}\right), \tag{2.7}$$

where k_0 is the pre-exponential factor, E_a is the activation energy, T is the temperature, and R is the ideal gas constant.

2.2.2 Bioprocesses

In the case of bioprocesses, i.e. fermentation processes, we consider the stoichiometry in the form presented in (2.1). In these systems, we consider the first component to be the biomass, i.e. the cell concentration, $C_1 = X$. As such, we describe the reaction rate for each reaction, r_i for $i \in \{1, 2, \dots, N\}$, as

$$r_i(c) = \mu(c) c_X, \tag{2.8}$$

where $\mu(\cdot)$ is the growth rate, or metabolic reaction rate, and c_X is the biomass concentration catalysing the reaction.

Growth rate - $\mu(c)$

In biochemical systems, we describe the growth rate in terms of the product of specific growth rates related to each of the chemical reactants, as

$$\mu(c) = \mu_{\max}(T) \prod_{j=2}^{M} \mu_{C_j}(c) \tag{2.9}$$

or in terms of the minimum of the specific growth rates, as

$$\mu(c) = \mu_{\max}(T) \min \left\{ \mu_{C_2}(c), \mu_{C_3}(c), \dots, \mu_{C_M}(c) \right\}. \tag{2.10}$$

The coefficient $\mu_{\max}(T)$ describes the maximum growth rate of the reaction.

Maximum growth rate
The maximum growth rate is commonly considered constant

$$\mu_{\max}(T) = \mu_{\max}. \tag{2.11}$$

In cases where temperature variations influence the maximum growth rate, we apply the Arrhenius expression

$$\mu_{\max}(T) = k_0 \exp\left(-\frac{E_a}{RT}\right), \tag{2.12}$$

where T is the temperature, k_0 is the pre-exponential factor, E_a is the activation energy of growth process, and R is the ideal gas constant.

Specific growth rate - $\mu_{C_j}(c)$

The specific growth rates defines the growth on specific substrates used in the metabolism of the cell.

Uninhibited growth (Monod kinetics)
The uninhibited growth on a specific chemical substrate is described by the Michaelis–Menten (or Monod) expression

$$\mu_{C_j}(c) = \frac{c_j}{K_{C_j} + c_j}, \tag{2.13}$$

where K_{C_j} is the half-velocity constant, i.e. when the concentration $c_j = K_{C_j}$ the specific growth rate $\mu_{C_j}(c) = 0.5$.

Substrate inhibition (Haldane kinetics)
The specific growth rate on a substrate, inhibited by the concentration of the same substrate, is described by the Haldane expression

$$\mu_{C_j}(c) = \frac{c_j}{K_{C_j} + c_j + c_j^2/K_{I,C_j}}, \tag{2.14}$$

where K_{I,C_j} is the substrate inhibition constant.

Product inhibition
The specific growth rate on a substrate, inhibited by the concentration of a product, C_p for $p \in \{2, 3, \cdots, M\}$ and $p \neq j$, is described by

$$\mu_{C_j}(c) = \frac{c_j}{K_{C_j} + c_j} \frac{1}{1 + c_p/K_{I,C_p}}, \tag{2.15}$$

or

$$\mu_{C_j}(c) = \frac{c_j}{K_{C_j} + c_j} \left(1 - \frac{c_p}{c_{p,\max}}\right), \tag{2.16}$$

where K_{I,C_p} is the product inhibition constant and $c_{p,\max}$ is the maximum concentration of the inhibiting product.

CHAPTER 3

Batch Reactors

In this chapter, we describe batch reactors (BRs). In section 3.1, we describe mass balances for the BR. Section 3.2 describes a first-principles ODE model for the BR based on mass balances. Figure 3.1 illustrates a BR.

3.1 First-principles description

In this section, we present mass (mole) balances for BRs. We consider the general mass balance equation [3]

$$\text{Accumulated} = \text{Inflow} - \text{Outflow} + \text{Generated}. \tag{3.1}$$

Mass balances

Over a small time-step, Δt, we consider the well-mixed BR with inflow, outflow, generated, and accumulated terms, as

$$\text{Inflow} = 0, \tag{3.2a}$$
$$\text{Outflow} = 0, \tag{3.2b}$$
$$\text{Generated} = RV\Delta t. \tag{3.2c}$$

The accumulated mass is the change in mass over the small time-step, Δt,

$$\text{Accumulated} = n(t + \Delta t) - n(t). \tag{3.3}$$

As such, we may describe the mass balance equation for the batch reactor as

$$n(t + \Delta t) - n(t) = RV\Delta t. \tag{3.4}$$

The same mass balance may be described as

$$\frac{n(t + \Delta t) - n(t)}{\Delta t} = RV, \tag{3.5}$$

which for $\Delta t \longrightarrow 0$, results in the ordinary differential equation

$$\frac{dn}{dt} = RV. \tag{3.6}$$

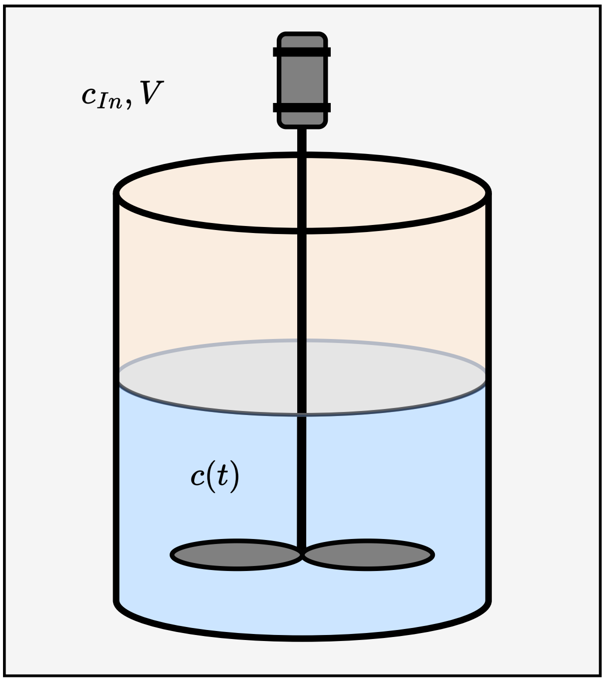


Figure 3.1: Illustration of a batch stirred tank reactor..

3.2 Modelling

The dynamics of the BR is described by the ODE model

$$\frac{dn}{dt} = R(c)V, \qquad n(t_0) = n_0. \tag{3.7}$$

where n are the masses and/or total energy, e.g. heat, in the system and V is the constant liquid volume of the reactor. The production term is defined by (2.4), as $R(c) = S^T r(c)$. The concentrations are $c = n/V$.

Concentration model:
Alternatively, the model can be expressed in terms of concentrations. We may write the model (3.7) as

$$\frac{dc}{dt} = R(c). \tag{3.8}$$

The derivation is as follows:

$$\frac{dn}{dt} = R(c)V \qquad \Longleftrightarrow \tag{3.9a}$$

$$\frac{d(cV)}{dt} = R(c)V \qquad \Longleftrightarrow \qquad \text{(substitute } n = cV\text{)} \tag{3.9b}$$

$$V\frac{dc}{dt} = R(c)V \qquad \Longleftrightarrow \tag{3.9c}$$

$$\frac{dc}{dt} = R(c). \tag{3.9d}$$

CHAPTER 4

Fedbatch Reactors

In this chapter, we describe fed-batch reactors (FBRs). In section 4.1, we describe volume and mass balances for the FBR. Section 4.2 describes a first-principles ODE model of the FBR based on volume and mass balances. Figure 4.1 illustrates the FBR.

4.1 First-principles description

In this section, we present mass (mole) balances for FBRs. We consider the general mass balance equation [3]

$$\text{Accumulated} = \text{Inflow} - \text{Outflow} + \text{Generated}. \tag{4.1}$$

Volume balances

Over a small time-step, Δt, we consider the well-mixed FBR with inflow, outflow, generated, and accumulated terms, as

$$\text{Inflow} = e^T F(t)\Delta t, \tag{4.2a}$$

$$\text{Outflow} = 0, \tag{4.2b}$$

$$\text{Generated} = 0. \tag{4.2c}$$

The accumulated term may be described by the difference in volume over the time-step, Δt, as

$$\text{Accumulated} = V(t + \Delta t) - V(t). \tag{4.3}$$

As such, we may describe the volume balance equation for the FBR as

$$V(t + \Delta t) - V(t) = e^T F(t)\Delta t. \tag{4.4}$$

The same volume balance may be described as

$$\frac{V(t + \Delta t) - V(t)}{\Delta t} = e^T F(t), \tag{4.5}$$

which for $\Delta t \longrightarrow 0$, results in the ordinary differential equation

$$\frac{dV}{dt} = e^T F(t). \tag{4.6}$$

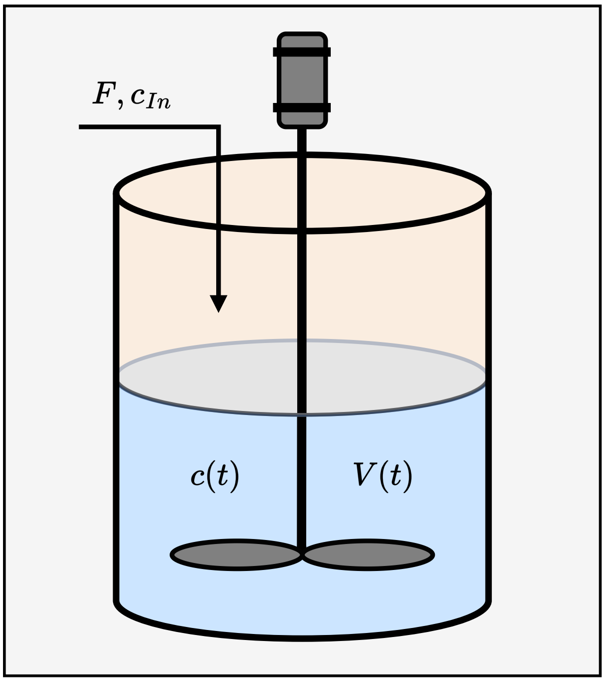


Figure 4.1: Illustration of a fed-batch stirred tank reactor.

Mass balances

Defining the small time-step, Δt, we may describe the right-hand side terms as follow; inflow

$$\text{Inflow} = C_{In}F(t)\Delta t, \tag{4.7a}$$
$$\text{Outflow} = 0, \tag{4.7b}$$
$$\text{Generated} = RV(t)\Delta t. \tag{4.7c}$$

The accumulated term is the difference in mass over the time-step, Δt, as

$$\text{Accumulated} = n(t+\Delta t) - n(t). \tag{4.8}$$

As such, we may describe the mass balance equation for the FBR as

$$n(t+\Delta t) - n(t) = C_{In}F(t)\Delta t + RV(t)\Delta t. \tag{4.9}$$

The same mass balance may be described as

$$\frac{n(t+\Delta t) - n(t)}{\Delta t} = C_{In}F(t) + RV(t), \tag{4.10}$$

which for $\Delta t \longrightarrow 0$, results in the ordinary differential equation

$$\frac{dn}{dt} = C_{In}F(t) + RV(t). \tag{4.11}$$

4.2 Modelling

Consider the system of ordinary differential equations describing mass and energy balances in a FBR

$$\frac{dV}{dt} = e^T F, \qquad V(t_0) = V_0, \tag{4.12a}$$

$$\frac{dn}{dt} = C_{In}F + R(c)V, \qquad n(t_0) = n_0, \tag{4.12b}$$

where V is the reactor volume, n are masses, molecules, and/or total energy, e.g. heat, in the system, F are the manipulated inlet flows of concentrations C_{In}, and $e \in \{1\}^{n_u}$. The production term is defined by (2.4), as $R(c) = S^T r(c)$. The concentrations are $c = n/V$.

Concentration model:
Alternatively, the model can be expressed in terms of concentrations. We may write the model (4.12) as

$$\frac{dV}{dt} = e^T F, \qquad V(t_0) = V_0, \tag{4.13a}$$

$$\frac{dc}{dt} = \left(C_{\mathrm{In}} - ce^T\right)\frac{F}{V} + R(c), \qquad c(t_0) = c_0. \tag{4.13b}$$

The derivation is as follows:

$$\frac{dn}{dt} = C_{\mathrm{In}}F + R(c)V \qquad \Longleftrightarrow \tag{4.14a}$$

$$\frac{d(cV)}{dt} = C_{\mathrm{In}}F + R(c)V \qquad \Longleftrightarrow \qquad \text{(substitute, } n = cV\text{)} \tag{4.14b}$$

$$\frac{dc}{dt}V + c\frac{dV}{dt} = C_{\mathrm{In}}F + R(c)V \qquad \Longleftrightarrow \qquad \text{(product rule)} \tag{4.14c}$$

$$\frac{dc}{dt} = \frac{C_{\mathrm{In}}F + R(c)V - c\frac{dV}{dt}}{V} \qquad \Longleftrightarrow \tag{4.14d}$$

$$\frac{dc}{dt} = C_{\mathrm{In}}\frac{F}{V} - ce^T\frac{F}{V} + R(c) \qquad \Longleftrightarrow \qquad \text{(insert, } \frac{dV}{dt}\text{)} \tag{4.14e}$$

$$\frac{dc}{dt} = \left(C_{\mathrm{In}} - ce^T\right)\frac{F}{V} + R(c). \tag{4.14f}$$

CHAPTER 5

Continuous Stirred Tank Reactors

In this chapter, we describe continuous stirred tank reactors (CSTRs). In section 5.1, we describe volume and mass balances for the CSTR in the cases where volume is variable and constant. Section 5.2 described first-principles ODE models for CSTRs with variable and constant volume. Figure 5.1 illustrates a CSTR.

5.1 First-principles description

In this section, we present mass (mole) balances for CSTRs with varible and constant volume. We consider the general mass balance equation [3]

$$\text{Accumulated} = \text{Inflow} - \text{Outflow} + \text{Generated}. \tag{5.1}$$

volume balance

Over a small time-step, Δt, we consider the CSTR with inflow, outflow, generated, and accumulated terms, as

$$\text{Inflow} = e^T F(t)\Delta t, \tag{5.2a}$$
$$\text{Outflow} = e_{\text{Out}}^T F_{\text{Out}}(t)\Delta t, \tag{5.2b}$$
$$\text{Generated} = 0. \tag{5.2c}$$

The accumulated term may be described by the difference in volume over the time-step, Δt, as

$$\text{Accumulated} = V(t+\Delta t) - V(t). \tag{5.3}$$

As such, we may describe the volume balance equation for the CSTR as

$$V(t+\Delta t) - V(t) = e^T F(t)\Delta t - e_{\text{Out}}^T F_{\text{Out}}(t)\Delta t. \tag{5.4}$$

The same volume balance may be described as

$$\frac{V(t+\Delta t) - V(t)}{\Delta t} = e^T F(t) - e_{\text{Out}}^T F_{\text{Out}}(t), \tag{5.5}$$

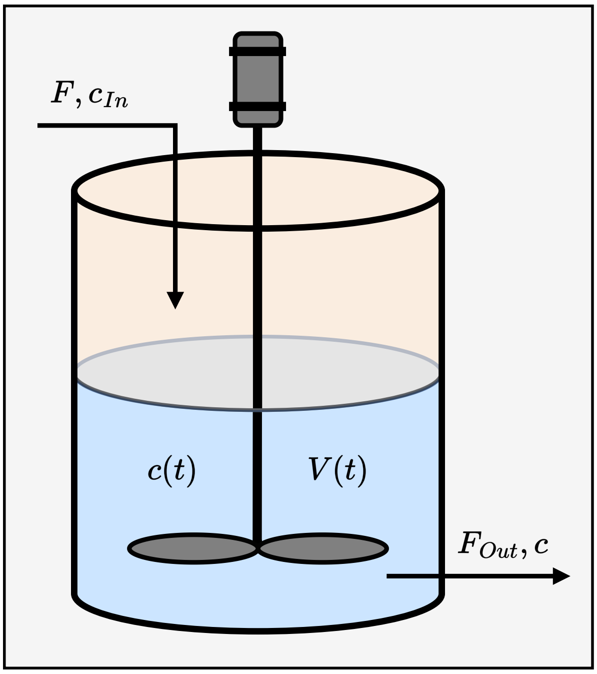


Figure 5.1: Illustration of a continuous stirred tank reactor.

which for $\Delta t \longrightarrow 0$ results in the ODE

$$\frac{dV}{dt}(t) = e^T F(t) - e_{\text{Out}}^T F_{\text{Out}}(t). \tag{5.6}$$

mass balance

Over a small time-step, Δt, we consider the CSTR with inflow, outflow, generated, and accumulated terms, as

$$\text{Inflow} = C_{In} F(t) \Delta t, \tag{5.7a}$$
$$\text{Outflow} = c(t) e_{\text{Out}}^T F_{\text{Out}}(t) \Delta t, \tag{5.7b}$$
$$\text{Generated} = R V(t) \Delta t. \tag{5.7c}$$

The accumulated term may be described by the difference in mass over the time-step, Δt, as

$$\text{Accumulated} = n(t + \Delta t) - n(t). \tag{5.8}$$

As such, we may describe the mass balance equation for the CSTR as

$$n(t+\Delta t) - n(t) = C_{In} F(t)\Delta t - c(t) e_{\text{Out}}^T F_{\text{Out}}(t)\Delta t + RV\Delta t. \tag{5.9}$$

The same mass balance may be described as

$$\frac{n(t+\Delta t) - n(t)}{\Delta t} = C_{In} F(t) - c(t) e_{\text{Out}}^T F_{\text{Out}}(t) + RV(t), \tag{5.10}$$

which for $\Delta t \longrightarrow 0$, results in the ordinary differential equation

$$\frac{dn}{dt} = C_{In} F(t) - c(t) e_{\text{Out}}^T F_{\text{Out}}(t) + RV(t). \tag{5.11}$$

5.2 Modelling

In this section, we present models describing CSTRs with variable and constant volume.

5.2.1 Variable volume

Consider the system of ordinary differential equations describing mass and energy balances in a continuous stirred tank reactor

$$\frac{dV}{dt} = e^T F - e_{\text{Out}}^T F_{\text{Out}}, \qquad V(t_0) = V_0, \tag{5.12a}$$

$$\frac{dn}{dt} = C_{\text{In}} F - c e_{\text{Out}}^T F_{\text{Out}} + R(c)V, \qquad n(t_0) = n_0, \tag{5.12b}$$

where V is the reactor volume, n are masses, molecules, and/or total energy, e.g. heat, in the system, F are the manipulated inlet flows of concentrations C_{In}, and $e \in \{1\}^{n_{u,\text{In}}}$ and $e_{\text{Out}} \in \{1\}^{n_{u,\text{Out}}}$ are vectors of ones. The production term is defined by (2.4), as $R(c) = S^T r(c)$. The concentrations are $c = n/V$.

Concentration model:
Alternatively, the model can be expressed in terms of concentrations. We may write model (5.12) as

$$\frac{dV}{dt} = e^T F - e_{\text{Out}}^T F_{\text{Out}}, \qquad V(t_0) = V_0, \tag{5.13a}$$

$$\frac{dc}{dt} = \left(C_{\text{In}} - c e^T\right) \frac{F}{V} + R(c), \qquad c(t_0) = c_0. \tag{5.13b}$$

The derivation is a follows:

$$\begin{aligned}
\frac{dn}{dt} &= C_{\text{In}}F - ce_{\text{Out}}^T F_{\text{Out}} + R(c)V && \Longleftrightarrow && && (5.14a)\\
\frac{d(cV)}{dt} &= C_{\text{In}}F - ce_{\text{Out}}^T F_{\text{Out}} + R(c)V && \Longleftrightarrow && (\text{substitute, } n = cV) && (5.14b)\\
\frac{dc}{dt}V + c\frac{dV}{dt} &= C_{\text{In}}F - ce_{\text{Out}}^T F_{\text{Out}} + R(c)V && \Longleftrightarrow && (\text{product rule}) && (5.14c)\\
\frac{dc}{dt} &= \frac{C_{\text{In}}F - ce_{\text{Out}}^T F_{\text{Out}} + R(c)V - c\frac{dV}{dt}}{V} && \Longleftrightarrow && && (5.14d)\\
\frac{dc}{dt} &= \frac{C_{\text{In}}F - ce^T F + R(c)V}{V} && \Longleftrightarrow && (\text{insert, } \frac{dV}{dt}) && (5.14e)\\
\frac{dc}{dt} &= \left(C_{\text{In}} - ce^T\right)\frac{F}{V} + R(c). && && && (5.14f)
\end{aligned}$$

5.2.2 Constant volume

A special case of the model presented in (5.12) is where $e^T F = e_{\text{Out}}^T F_{\text{Out}}$. This describes a constant volume CSTR. We describe such a system as

$$\frac{dn}{dt} = \left(C_{\text{In}} - ce^T\right) F + R(c)V, \qquad n(t_0) = n_0, \qquad (5.15)$$

where V is the reactor volume, n are masses, molecules, and/or total energy, e.g. heat, in the system, F are the manipulated inlet flows of concentrations C_{In}, and $e \in \{1\}^{n_{u,\text{In}}}$ is a vector of ones. The production term is defined by (2.4), as $R(c) = S^T r(c)$.

Concentration model:
In the constant volume case, we may similarly expressed the model in terms of concentrations as

$$\frac{dc}{dt} = \left(C_{\text{In}} - ce^T\right) \frac{F}{V} + R(c), \qquad c(t_0) = c_0. \qquad (5.16)$$

The derivation is as follows:

$$\begin{aligned}
\frac{dn}{dt} &= \left(C_{\text{In}} - ce^T\right) F + R(c)V && \Longleftrightarrow && && (5.17a)\\
\frac{d(cV)}{dt} &= \left(C_{\text{In}} - ce^T\right) F + R(c)V && \Longleftrightarrow && (\text{substitute } n = cV) && (5.17b)\\
V\frac{dc}{dt} &= \left(C_{\text{In}} - ce^T\right) F + R(c)V && \Longleftrightarrow && && (5.17c)\\
\frac{dc}{dt} &= \left(C_{\text{In}} - ce^T\right) \frac{F}{V} + R(c). && && && (5.17d)
\end{aligned}$$

CHAPTER 6

Plug Flow Reactors

In this chapter, we describe plug flow reactors (PFRs). In section 6.1, we describe mass balances for the PFR. Section 6.2 describes a first-principles partial differential equation (PDE) model for the PFR based on mass balances. Section 6.3 present an ODE model for the PFR based on a finite-volume discretisation of the PDE model. Figure 6.1 illustrates a PFR.

6.1 First-principles description

In this section, we present mass (mole) balances for PFRs. We consider the general mass balance equation [3]

$$\text{Accumulated} = \text{Inflow} - \text{Outflow} + \text{Generated}. \tag{6.1}$$

mass balances

We define the small time-step, Δt. Furthermore, we consider the partial reactor volume, ΔV, of dimensions $\Delta V = A\Delta z$, where Δz is a small spatial step. We consider the fluxes in and out of each partial reactor volume through a pipe cross-section, $N(t, z)$, as

$$\text{Inflow} = AN(t, z)\Delta t, \tag{6.2a}$$
$$\text{Outflow} = AN(t, z + \Delta z)\Delta t, \tag{6.2b}$$

and generated within each volume

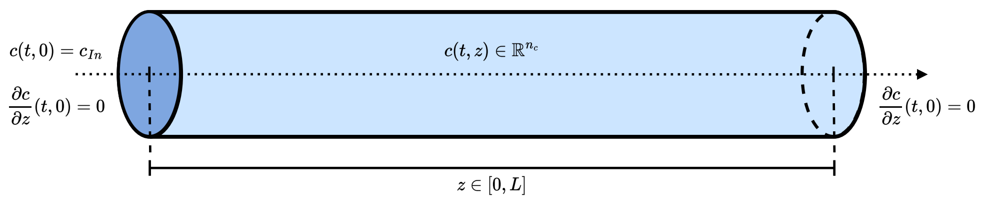
$$\text{Generated} = AR\Delta z\Delta t. \tag{6.3}$$

The accumulated term is described by the difference in mass over the time-step, Δt, as

$$\begin{aligned} \text{Accumulated} &= n(t + \Delta t, z) - n(t, z) && (6.4a)\\ &= A\Delta z\frac{(n(t + \Delta t, z) - n(t, z))}{A\Delta z} && (6.4b)\\ &= A\Delta z\left(c(t + \Delta t, z) - c(t, z)\right). && (6.4c)\end{aligned}$$

As such, we may describe the mass balance equation for the PFR as

$$A\Delta z\left(c(t + \Delta t, z) - c(t, z)\right) = AN(t, z)\Delta t - AN(t, z + \Delta z)\Delta t + AR\Delta z\Delta t. \tag{6.5}$$



$c(t,0) = c_{In}$
$\frac{\partial c}{\partial z}(t,0) = 0$
$c(t,z) \in \mathbb{R}^{n_c}$
$\frac{\partial c}{\partial z}(t,0) = 0$
$z \in [0, L]$

Figure 6.1: Illustration of a plug flow reactor (PFR) of length L.

The same mass balance may be described as

$$c(t+\Delta t, z) - c(t,z) = \frac{N(t,z) - N(t,z+\Delta z)}{\Delta z}\Delta t + R\Delta t \tag{6.6a}$$

$$= -\frac{N(t,z+\Delta z) - N(t,z)}{\Delta z}\Delta t + R\Delta t, \tag{6.6b}$$

and as

$$\frac{c(t+\Delta t, z) - c(t,z)}{\Delta t} = -\frac{N(t,z+\Delta z) - N(t,z)}{\Delta z} + R. \tag{6.7}$$

For $\Delta t \longrightarrow 0$ and $\Delta z \longrightarrow 0$, this results in the partial differential equation

$$\frac{\partial c}{\partial t} = -\frac{\partial N}{\partial z} + R. \tag{6.8}$$

6.2 Modelling

In this section, we present a models describing the PFR. We present a first-principles PDE model based on mass balances. Furthermore, we present an ODE model based on a finite-volume discretisation of the PDE model. Consider the system of partial differential equations

$$\frac{\partial c}{\partial t} = -\frac{\partial N}{\partial z} + R(c), \qquad c(t_0, z) = c_0(z), \tag{6.9}$$

where $c(t,z)$ are concentrations, $N(.)$ are fluxes, and $R(.)$. The production term is defined by (2.4), as $R(c) = S^T r(c)$. We compute the fluxes, as

$$N(t,z) = v(t)c(t,z) - D\frac{\partial c}{\partial z}(t,z), \tag{6.10}$$

where $v(t)c(t,z)$ is the convective flow and $-D\frac{\partial c}{\partial z}(t,z)$ is the dispersive flow. The linear velocity, v, is computed as

$$v(t) = \frac{F_t(t)}{A}, \qquad F_t(t) = e^T F(t), \tag{6.11}$$

where $F(t) \in \mathbb{R}^{n_u}$ are the manipulated inlet flows, $F_t(t) \in \mathbb{R}$ is the total flow, $A \in \mathbb{R}$ is the cross-sectional area of the pipe, and $e \in \{1\}^{n_u}$ is a vector of ones. D is a diagonal matrix of dispersion constants.

6.2.1 Boundary conditions

We consider Danckwerts' boundary conditions [1]. As such, we consider an inlet flow of known concentrations and no dispersive flow through either end of the reactor pipe.

Inlet conditions

We consider the boundary conditions at the inlet

$$c(t,0) = c_{In}(t), \qquad \frac{\partial c}{\partial z}(t,0) = 0. \tag{6.12}$$

This yields the fluxes at the inlet (with no dispersion)

$$N(t,0) = v(t)c_{In}(t). \tag{6.13}$$

Outlet conditions

We consider the boundary condition at the outlet

$$\frac{\partial c}{\partial z}(t,L) = 0. \tag{6.14}$$

This condition defines the outlet flux (with no dispersion)

$$N(t,L) = v(t)c(t,L). \tag{6.15}$$

6.3 Finite-volume discretisation

In this section, we present a finite-volume discretisation of the PFR. Figure 6.2 illustrates the discretisation of the PFR. Consider the finite set of equally sized reactor volumes

$$\Delta V_i = \Delta V = A\Delta z, \qquad i \in \{1, 2, \ldots, N_z\}, \tag{6.16}$$

where ΔV is the size of the volumes, A is the cross-sectional area of the reactor pipe assumed to be constant along the reactor, and Δz is the spatial discretisation. As such, the total length of the pipe is $L = N_z \Delta z$. We assume each volume to be well-mixed. The centre of each volume is chosen as discretisation point, such that

$$z_i = \left(i - \frac{1}{2}\right)\Delta z, \qquad i \in \{1, 2, \ldots, N_z\}. \tag{6.17}$$

The concentrations $c(t, z_i) = c_i(t)$ thus represent the concentration in each discrete volume. Similarly, we describe the positions of the in- and out-let faces of each volume, as

$$z_{i+1/2} = i\Delta z, \qquad i \in \{0, 1, \ldots, N_z\}. \tag{6.18}$$

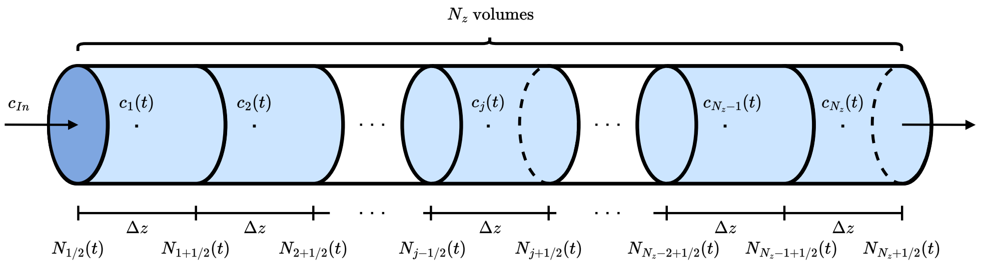


Figure 6.2: Discretisation of a plug flow reactor (PFR) of length L. The spatial dimension of the reactor is discretised using a finite-volume discretisation with N_z discrete volumes of size Δz.

We describe the fluxes through each face of the discretisation, as

$$N(t, z + i\Delta z) = N_{i+1/2}(t), \qquad i \in \{0, 1, \dots, N_z\}. \tag{6.19}$$

We consider the fluxes descried in (6.10), and apply a central difference approximation for the spatial derivative of the concentrations, as

$$\frac{\partial c_{i+1/2}}{\partial z}(t) = \frac{c_{i+1}(t) - c_i(t)}{\Delta z}, \qquad i \in \{1, 2, \dots, N_z - 1\} \tag{6.20}$$

Considering the boundary conditions, we compute the fluxes at the volume faces, as

$$N_{i+1/2}(t) = v(t)c_{In}(t), \qquad i \in \{0\}, \tag{6.21a}$$

$$N_{i+1/2}(t) = v(t)c_i(t) - D\frac{c_{i+1}(t) - c_i(t)}{\Delta z}, \qquad i \in \{1, 2, \dots, N_z - 1\}, \tag{6.21b}$$

$$N_{i+1/2}(t) = v(t)c_i(t), \qquad i \in \{N_z\}. \tag{6.21c}$$

We apply a central difference approximation to describe the spatial derivative of the flux through each of the volumes, as

$$\frac{\partial N_i}{\partial z} \approx \frac{N_{i+1/2}(t) - N_{i-1/2}(t)}{\Delta z}. \tag{6.22}$$

As such, we can describe the finite-volume discretisation of the PFR PDE in the form of the ODE

$$\frac{dc_i}{dt}(t) = -\frac{N_{i+1/2}(t) - N_{i-1/2}(t)}{\Delta z} + R(c_i(t)), \qquad i \in \{1, 2, \dots, N_z\}. \tag{6.23}$$

6.3.1 ODE - finite-volume discretisation

Consider the ODE system arising from the finite-volume discretisation of the PFR

$$\frac{dc_i}{dt}(t) = -\frac{N_{i+1/2}(t) - N_{i-1/2}(t)}{\Delta z} + R(c_i(t)), \qquad i \in \{1, 2, \dots, N_z\}, \tag{6.24}$$

with initial conditions $c_i(t_0) = c_{i,0}$. The fluxes, $N_{i+1/2}(t) = N(t, z + (i + 1/2)\Delta z)$ for $i \in \{0, 1, \dots, N_z\}$, are computed as

$$N_{i+1/2}(t) = v(t) c_{In}(t), \qquad i \in \{0\}, \tag{6.25a}$$

$$N_{i+1/2}(t) = v(t) c_i(t) - D \frac{c_{i+1}(t) - c_i(t)}{\Delta z}, \qquad i \in \{1, 2, \dots, N_z - 1\}, \tag{6.25b}$$

$$N_{i+1/2}(t) = v(t) c_i(t), \qquad i \in \{N_z\}. \tag{6.25c}$$

The concentrations are $c_i(t) = c(t, z + i\Delta z)$ for $i \in \{1, 2, \dots, N_z\}$.

CHAPTER 7
Numerical Examples

In this chapter, we present examples of an exothermic chemical reaction and fermentation process for the production of single-cell protein in batch, fed-batch, continuous stirred tank, and plug flow reactors, respectively.

7.1 Exothermic reaction in adiabatic reactor

In this section, we consider the exothermic reaction in an adiabatic reactor as presented by [12, 15]. Consider the stoichiometric description

$$\mathrm{Na_2S_2O_3} + 2\,\mathrm{H_2O_2} \longrightarrow \frac{1}{2}\,\mathrm{Na_2SO_4} + \frac{1}{2}\,\mathrm{Na_2S_3O_6} + 2\,\mathrm{H_2O}, \qquad r_1. \tag{7.1}$$

Sodium thiosulfate is oxidized by hydrogen peroxide to form sodium sulfate and sodium trithionate. The reaction is exothermic, i.e. generates heat. In the model, we consider the reactants, $A = \mathrm{Na_2S_2O_3}$ and $B = H_2O_2$, as well as the heat generated by the reaction, T. As such, the modelled stoichiometry is

$$\mathrm{A} + 2\,\mathrm{B} \longrightarrow \beta\mathrm{T}, \qquad r_1. \tag{7.2}$$

The stoichiometric matrix arising from (7.2) is

$$S = \begin{array}{cccc} \mathrm{A} & \mathrm{B} & \mathrm{T} & \\ [-1 & -2 & \beta] & \mathrm{r_1} \end{array} . \tag{7.3}$$

The stoichiometric coefficient for the heat generated by the reaction, β, is computed as

$$\beta = -\frac{\Delta H_r}{\rho c_P}, \tag{7.4}$$

where ΔH_r is the enthalpy of the reaction, ρ is the density of the medium, and c_P is the specific capacity. The rate of reaction is computed as

$$r(c) = r_1(c) = k(c_T)c_A c_B, \qquad k(c_T) = k_0 \exp\left(-\frac{E_a}{R}c_T\right), \tag{7.5}$$

where $k(c_T)$ is the rate constant governed by the Arrhenius expression as presented in (2.7) and c_i for $i \in \{A, B, T\}$ are concentrations of the components A, B, and T. The concentration of heat, c_T [K], corresponds to the temperature in the reactor. The relevant constants are presented in Table 7.1.

variable	**value**	**unit**	**description**
ρ	1.0	kg/L	density
c_p	4.186	kJ/(kg·K)	specific heat capacity
k_0	exp(24.6)	L/(mol·s)	Arrhenius constant
E_a/R	8500.0	K	activation energy
ΔH_r	-560.0	kJ/mol	reaction enthalpy

Table 7.1: Parameters for the examples involving the exothermic reaction.

Jacobian of r wrt. c - $\frac{\partial r}{\partial c}$ The Jacobian of the reaction rate wrt. the concentrations

$$\begin{aligned}\frac{\partial r}{\partial c} &= \frac{\partial}{\partial c}\left(k(c_T)c_A c_B\right) &\text{(7.6a)}\\ &= \frac{\partial k}{\partial c}c_A c_B + k(c_T)\frac{\partial c_A}{\partial c}c_B + k(c_T)c_A\frac{\partial c_B}{\partial c}. &\text{(7.6b)}\end{aligned}$$

The Jacobian of the kinetics function wrt. the concentrations is

$$\frac{\partial k}{\partial c} = \begin{bmatrix}0 & 0 & \frac{\partial k}{\partial c_T}\end{bmatrix}, \qquad (7.7)$$

where

$$\begin{aligned}\frac{\partial k}{\partial c_T} &= \frac{\partial}{\partial c}\left(k_0 \exp\left(-\frac{E_a}{R}c_T\right)\right) &\text{(7.8a)}\\ &= k_0\frac{\partial}{\partial c}\left(\exp\left(-\frac{E_a}{R}c_T\right)\right) &\text{(7.8b)}\\ &= -k_0\frac{E_a}{R}\exp\left(-\frac{E_a}{R}c_T\right). &\text{(7.8c)}\end{aligned}$$

The Jacobian of the concentration of component A wrt. the concentrations is

$$\frac{\partial c_A}{\partial c} = \begin{bmatrix}1 & 0 & 0\end{bmatrix}. \qquad (7.9)$$

The Jacobian of the concentration of component B wrt. the concentration is

$$\frac{\partial c_B}{\partial c} = \begin{bmatrix}0 & 1 & 0\end{bmatrix}. \qquad (7.10)$$

7.1.1 Batch reactor

Consider an adiabatic, i.e. thermally insulated, BR described by (3.7). Table 7.2 describes the parameters and initial values of the numerical experiment. Figure 7.1 presents results.

Table 7.2: Parameters for the adiabatic BR example involving the exothermic reaction.

variable	value	unit	description
V	0.105	L	reactor volume
$n_{A,0}$	0.084	mol	initial value of n_A
$n_{B,0}$	0.126	mol	initial value of n_B
$n_{T,0}$	28.733	K L	initial value of n_T

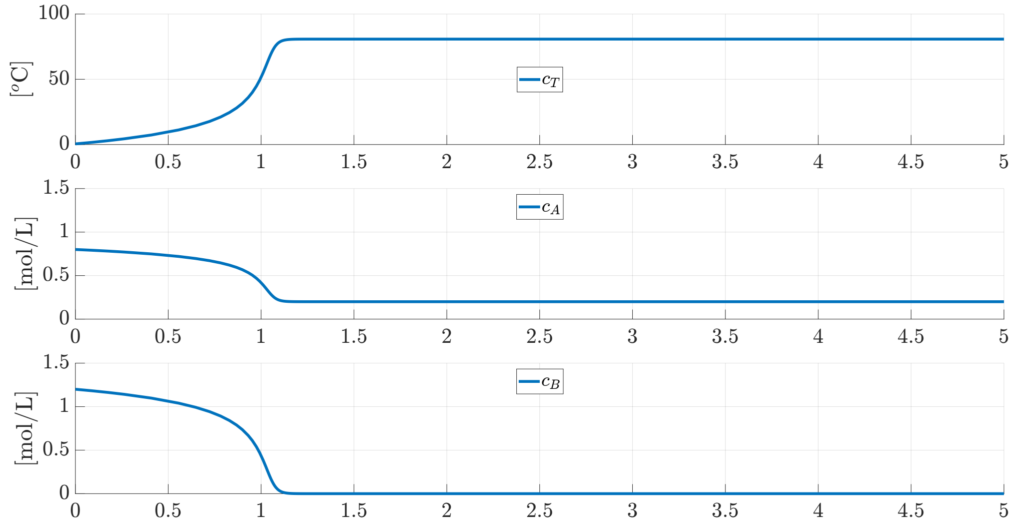


Figure 7.1: Open-loop simulation of an adiabatic BR with an exothermic reaction.

7.1.2 Fed-batch reactor

Consider an adiabatic, i.e. thermally insulated, FBR described by (4.12). Table 7.3 describes the parameters and initial values of the numerical experiment. Figure 7.2 presents results.

Table 7.3: Parameters for the adiabatic FBR example involving the exothermic reaction.

variable	value	unit	description
$c_{A,in}$	1.6/2	mol/L	inlet concentration of A
$c_{B,in}$	2.4/2	mol/L	inlet concentration of B
$c_{T,in}$	273.65	K	inlet temperature
V_0	0.0105	L	initial value of V
$n_{A,0}$	0.0084	mol	initial value of n_A
$n_{B,0}$	0.0126	mol	initial value of n_B
$n_{T,0}$	2.8733	K L	initial value of n_T

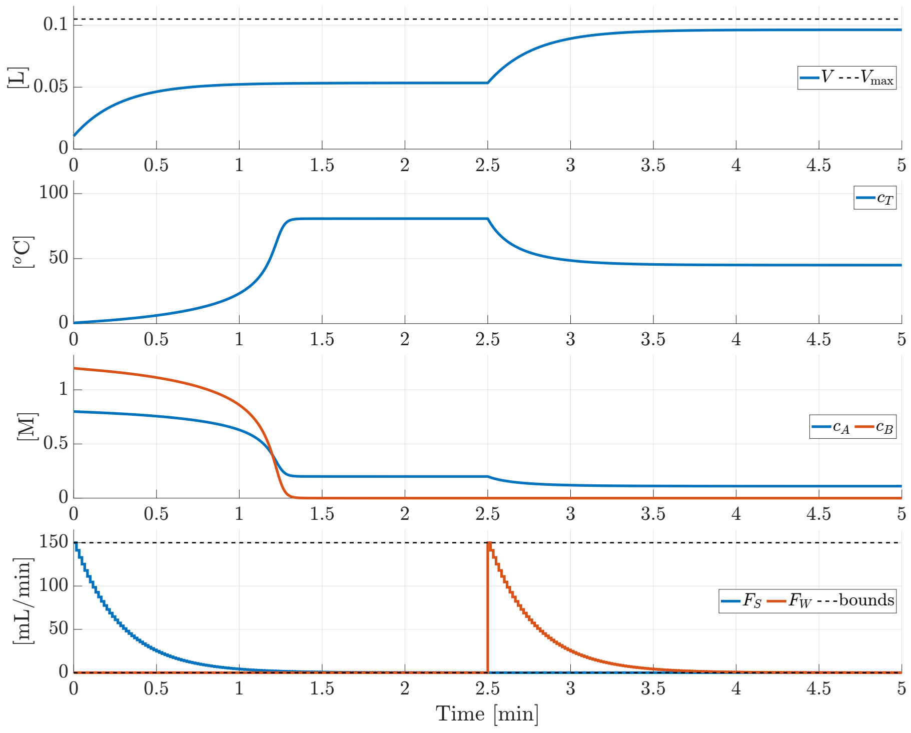


Figure 7.2: Open-loop simulation of FBR with an exothermic reaction.

7.1.3 Continuous stirred tank reactor

Consider an adiabatic, i.e. thermally insulated, CSTR described by (5.12). Table 7.4 describes the parameters and initial vales of the numerical experiment. Figure 7.3 presents results.

Table 7.4: Parameters for the adiabatic CSTR example involving the exothermic reaction.

variable	**value**	**unit**	**description**
$c_{A,in}$	1.6/2	mol/L	inlet concentration of A
$c_{B,in}$	2.4/2	mol/L	inlet concentration of B
$c_{T,in}$	273.65	K	inlet temperature
V_0	0.105	L	reactor volume
$n_{A,0}$	0.084	mol	initial value of n_A
$n_{B,0}$	0.126	mol	initial value of n_B
$n_{T,0}$	28.73	mol	initial value of n_C

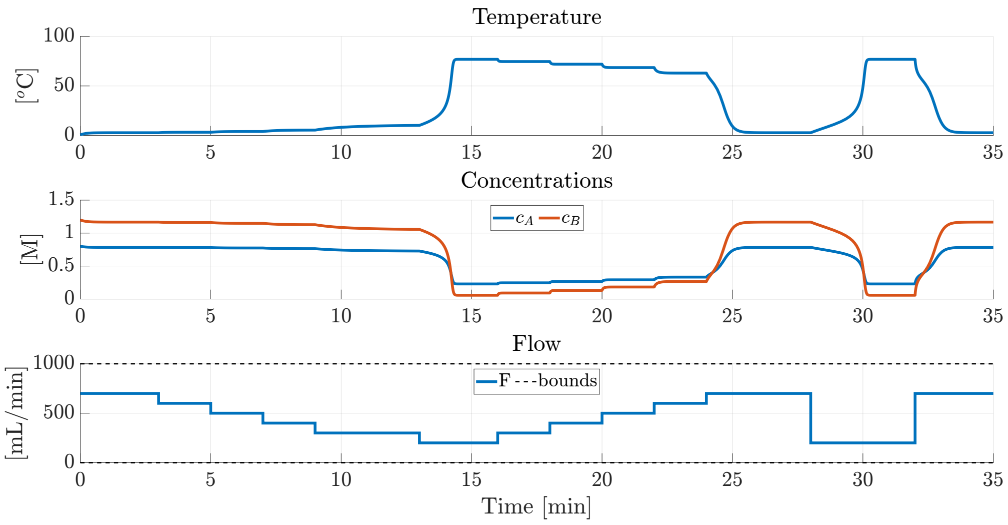


Figure 7.3: Open-loop simulation of CSTR with an exothermic reaction.

7.1.4 Plug flow reactor

Consider an adiabatic, i.e. thermally insulated, PFR described by (6.9). Table 7.5 describes the parameters and initial vales of the numerical experiment. Figure 7.4 presents results.

Table 7.5: Parameters for the adiabatic PFR example involving the exothermic reaction.

variable	value	unit	description
A	0.1	m^2	cross-sectional area
L	10.0	m	reactor length
D_A	0.029	m^2/s	diffusion constant for A
D_B	0.029	m^2/s	diffusion constant for B
D_T	0.143	m^2/s	diffusion constant for T
N_z	20	-	number of discrete volumes
$c_{A,in}$	1.6/2	mol/L	inlet concentration of A
$c_{B,in}$	2.4/2	mol/L	inlet concentration of B
$c_{T,in}$	273.65	K	inlet temperature
$c_{A,0}$	0.0	mol/L	initial value of c_A
$c_{B,0}$	0.0	mol/L	initial value of c_B
$c_{T,0}$	273.65	K	initial value of c_T
F_S	600.00	mL/min	constant substrate flow-rate
F_W	100.00	mL/min	constant water flow-rate

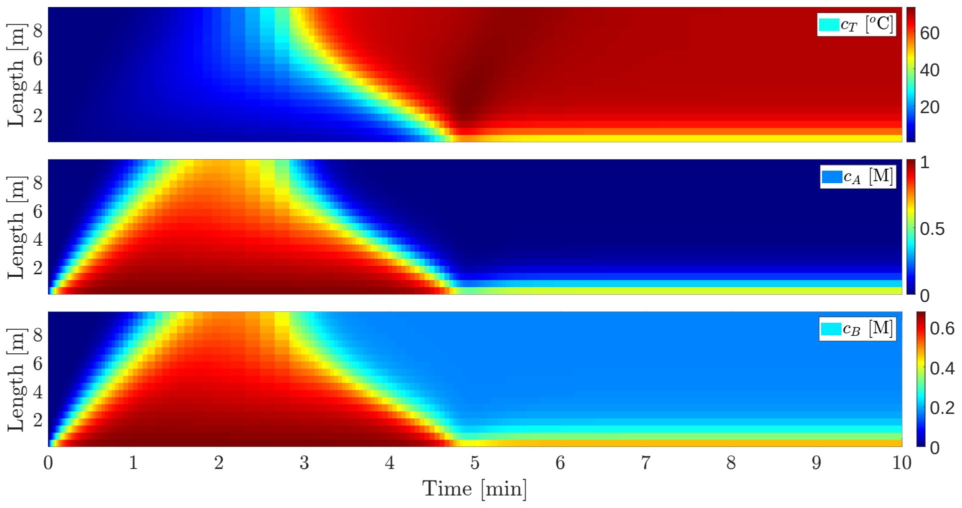


Figure 7.4: Open-loop simulation of PFR with an exothermic reaction.

7.2 Single-cell protein production

In this section, we consider the single-cell protein production model presented by [8, 2, 9]. Consider the stoichiometric description

$$\begin{aligned} &1.778\,\mathrm{CH_3OH} + 0.510\,\mathrm{HNO_3} + 0.779\,\mathrm{O_2} + \mathrm{X} \longrightarrow \\ &\quad 2\,\mathrm{X} + 0.654\,\mathrm{CO_2} + 1.414\,\mathrm{H_2O}\,, \end{aligned} \qquad r_1(c). \tag{7.11}$$

The bacterium *Methylococcus capsulatus* (Bath) metabolises a carbon source, methanol, and a nitrogen source, nitric acid, under aerobic conditions, i.e. in the presence of oxygen, to grow. In the growth reaction, the bacterium produces carbon dioxide and water. In the model, we consider the reactant, $S = \mathrm{CH_3OH}$, as well as the bacteria (biomass), $X = \mathrm{CH_{1.8}O_{0.5}N_{0.2}}$. As such, the modelled stoichiometry is

$$1.778\,\mathrm{S} + \mathrm{X} \longrightarrow 2\,\mathrm{X}\,, \qquad r_1. \tag{7.12}$$

The stoichiometric matrix arising from (7.12)

$$S = \begin{array}{cc} \mathrm{X} & \mathrm{S} \\ {[1} & {-1.778]} \end{array} \; \mathrm{r_1} \; . \tag{7.13}$$

The rate of reaction is computed as

$$r(c) = r_1(c) = \mu(c)c_X, \qquad \mu(c) = \mu_{\max}\mu_S(c), \tag{7.14}$$

The growth rate, $\mu(c)$, is computed as presented in (2.9). The specific growth rates are computed as

$$\mu_S(c) = \frac{c_S}{K_S + c_S + c_S^2/K_{I,S}}. \tag{7.15}$$

The relevant constants are presented in Table 7.2.

Jacobian of r wrt. c - $\frac{\partial r}{\partial c}$

The Jacobian of the reaction rate wrt. the concentrations is

$$\frac{\partial r}{\partial c} = \frac{\partial}{\partial c}\left(\mu(c)c_X\right) \tag{7.16a}$$

$$= \frac{\partial \mu}{\partial c}(c)c_X + \frac{\partial c_X}{\partial c}\mu(c). \tag{7.16b}$$

The Jacobian of the growth rate is

$$\frac{\partial \mu}{\partial c} = \frac{\partial}{\partial c}\left(\mu_{\max}\mu_S(c)\right) \tag{7.17a}$$

$$= \mu_{\max}\frac{\partial \mu_S}{\partial c}. \tag{7.17b}$$

Table 7.6: Parameters for the examples involving the exothermic reaction.

variable	value	unit	description
$\mu_{\max}$	0.370	1/h	maximum growth rate
K_S	0.021	kg/m^3	kinetic parameter for S
$K_{I,S}$	0.380	kg/m^3	kinetic inhibition parameter for S

The Jacobian of the specific growth rate on substrate is

$$\frac{\partial \mu_S}{\partial c} = \frac{\partial}{\partial c}\left(\frac{c_S}{K_S + c_S + c_S^2/K_{I,S}}\right) \tag{7.18a}$$

$$= \frac{1}{g(c)^2}\left(\frac{\partial g}{\partial c}h(c) - \frac{\partial h}{\partial c}g(c)\right), \tag{7.18b}$$

where

$$g(c) = c_S, \qquad h(c) = K_S + c_S + c_S^2/K_{I,S}, \tag{7.19a}$$

$$\frac{\partial g}{\partial c} = \begin{bmatrix} 0 & 1 \end{bmatrix}, \qquad \frac{\partial h}{\partial c} = \begin{bmatrix} 0 & 1 + \frac{2}{K_{I,S}} c_S \end{bmatrix}. \tag{7.19b}$$

The Jacobian of the biomass concentration wrt. the concentrations is

$$\frac{\partial c_X}{\partial c} = \begin{bmatrix} 1 & 0 \end{bmatrix}. \tag{7.20}$$

7.2.1 Batch reactor

Consider a BR described by (3.7). Table 7.7 describes the parameters and initial values of the numerical experiment. Figure 7.5 presents results.

Table 7.7: Parameters for the BR for single-cell protein production.

variable	**value**	**unit**	**description**
V	1.0	m^3	liquid reactor volume
$n_{X,0}$	0.05	kg/m^3	initial value of n_X
$n_{S,0}$	1.0	kg/m^3	initial value of n_S

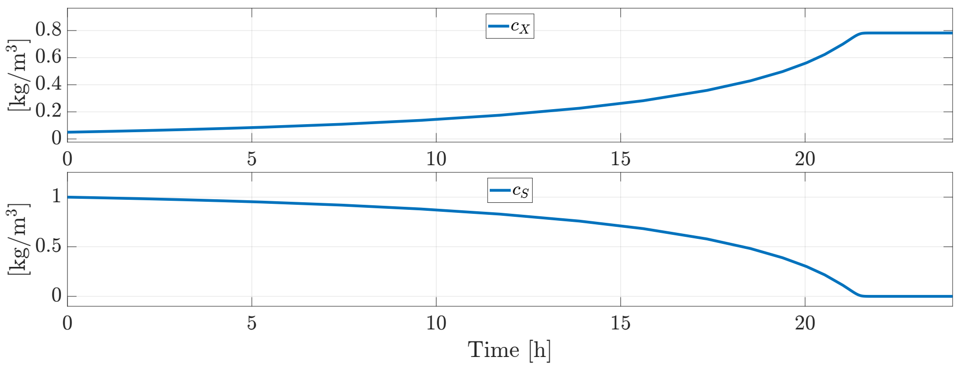


Figure 7.5: Open-loop simulation of BR for single-cell protein production.

7.2.2 Fed-batch reactor

Consider a FBR described by (4.12). Table 7.8 describes the parameters and initial values of the numerical experiment. Figure 7.6 presents results.

Table 7.8: Parameters for the FBR for single-cell protein production.

variable	**value**	**unit**	**description**
$c_{W,In}$	$[0.0, 0.0]^T$	m^3/h	inlet concentration in water inlet
$c_{S,In}$	$[0.0, 10.0]^T$	m^3/h	inlet concentration of substrate inlet
V_0	0.1	m^3	initial value of V
$n_{X,0}$	0.5	kg/m^3	initial value of n_X
$n_{S,0}$	1.0	kg/m^3	initial value of n_S

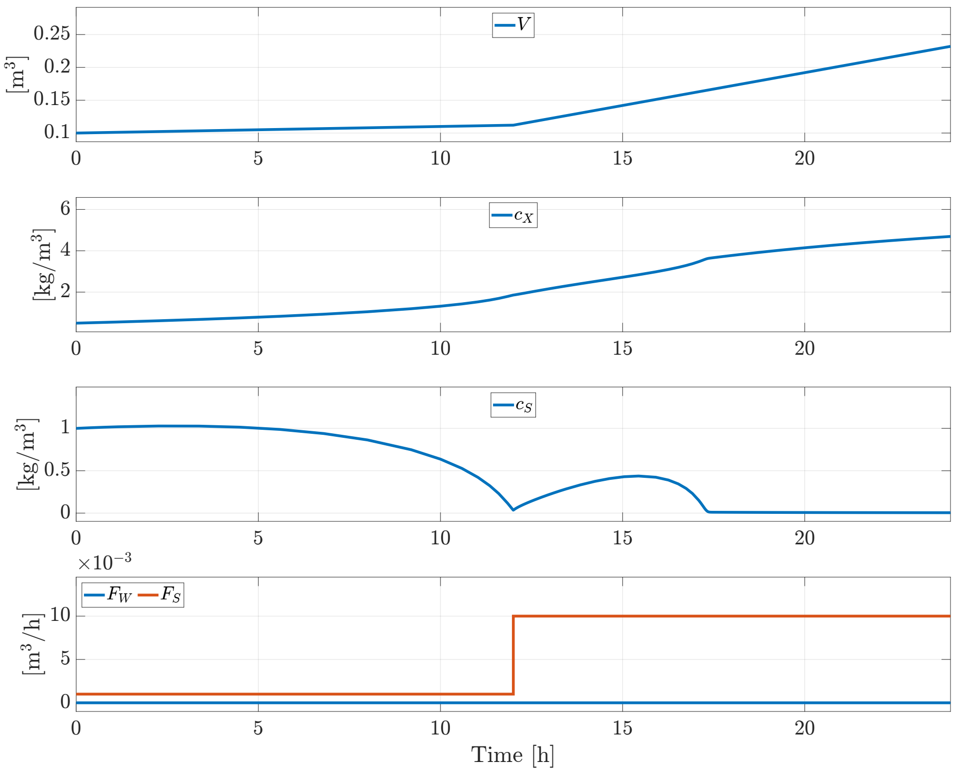


Figure 7.6: Open-loop simulation of FBR for single-cell protein production.

7.2.3 Continuous stirred tank reactor

Consider a CSTR described by (5.12). Table 7.9 describes the parameters and initial values of the numerical experiment. Figure 7.7 presents results.

Table 7.9: Parameters for the CSTR for single-cell protein production.

variable	value	unit	description
$c_{W,In}$	$[0.0, 0.0]^T$	m^3/h	inlet concentration in water inlet
$c_{S,In}$	$[0.0, 10.0]^T$	m^3/h	inlet concentration of substrate inlet
V_0	1.0	m^3	initial value of V
$n_{X,0}$	0.5	kg/m^3	initial value of n_X
$n_{S,0}$	1.0	kg/m^3	initial value of n_S

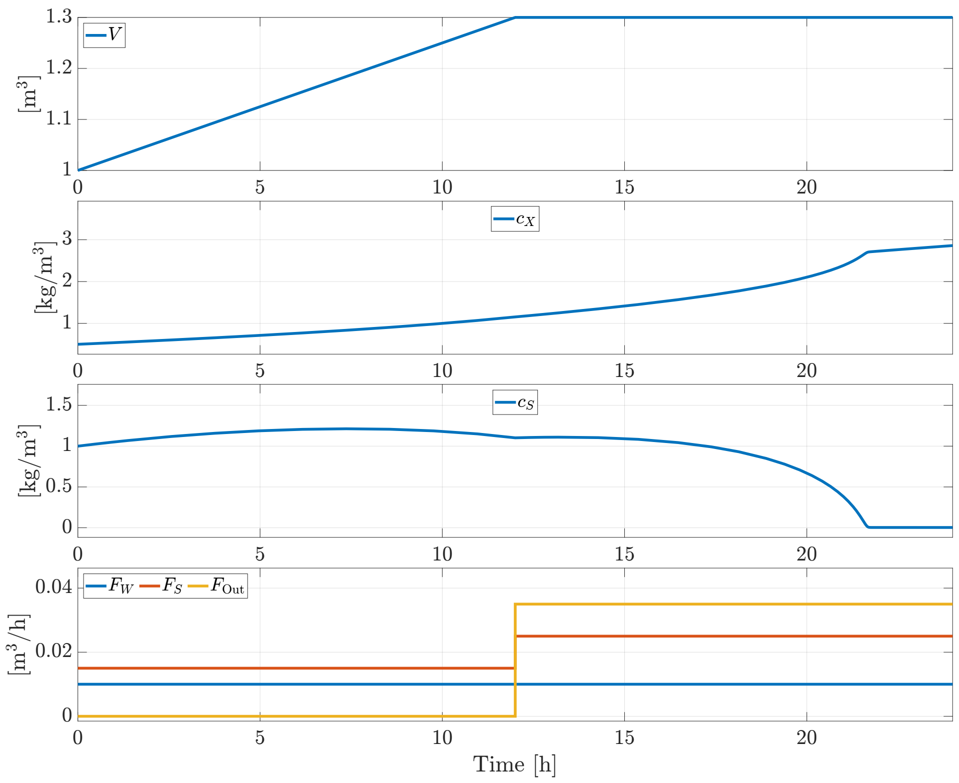


Figure 7.7: Open-loop simulation of CSTR for single-cell protein production.

7.2.4 Plug flow reactor

Consider a PFR described by (6.9). Table 7.10 describes the parameters and initial values of the numerical experiment. Figure 7.8 presents results.

Table 7.10: Parameters for the PFR for single-cell protein production.

variable	value	unit	description
A	0.1	m^2	cross-sectional area of pipe
L	10.0	m	reactor length
D_X	103.68	m^2/h	dispersion rate for biomass
D_S	103.68	m^2/h	dispersion rate for substrate
N_z	20	-	number of discrete volumes
$c_{W,In}$	$[0.0, 0.0]^T$	m^3/h	inlet concentration in water inlet
$c_{S,In}$	$[0.0, 10.0]^T$	m^3/h	inlet concentration of substrate inlet
$n_{X,0}$	0.5	kg/m^3	initial value of n_X
$n_{S,0}$	1.0	kg/m^3	initial value of n_S
F_W	0.05	m^3/h	inlet flow-rate of water
F_S	0.01	m^3/h	inlet flow-rate of substrate

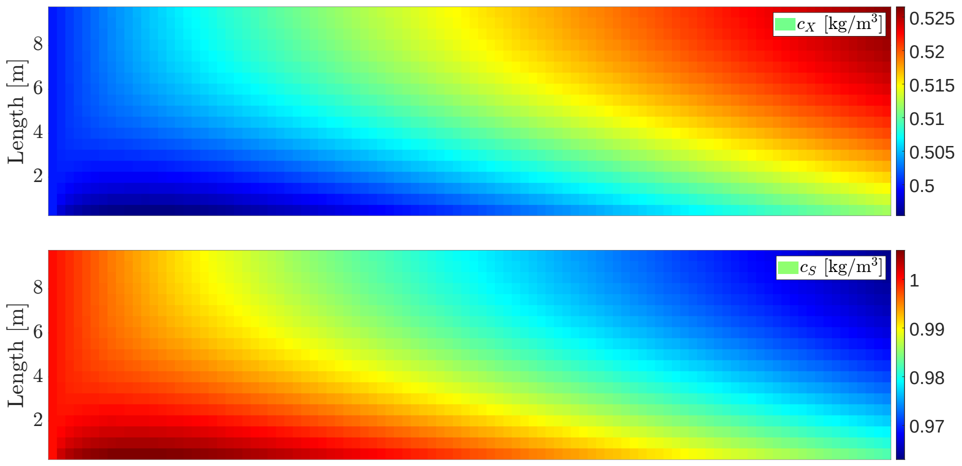


Figure 7.8: Open-loop simulation of PFR for single-cell protein production.

CHAPTER 8
Conclusions

In this technical report, we described a modelling framework for reactive systems. We introduced the stoichiometric matrix and rate of reaction vector in the general case. We presented kinetic descriptions for chemical and biochemical systems, involved in computing the reaction rates for those systems. We described and derived ordinary differential equation models for batch, fedbatch, and continuous stirred tank reactors. We presented a partial differential equation model the plug flow reactor based on mass balances and formulated an ordinary differential equation model derived from a finite-volume discretisation of the partial differential equation model. Jacobian information and implementations in Matlab and Python are described in the appendix. We presented two example reactive systems; an exothermic chemical reaction in an adiabatic environment and a fermentation for single-cell protein production. We conducted numerical experiments in the four presented reactor types for example reactive systems. We describe gas-liquid mass transfer for reactive systems involving both liquid and gas phases in the appendix.

APPENDIX A
Gas-liquid Mass Transfer

In this chapter, we describe methods of computing the gas-liquid mass transfer in chemical and biochemical systems. We consider the sets of components purely dissolved in the liquid phase, i.e. with no gas phase, and components in the gas and dissolved liquid phases, as

$$\mathcal{C}_l, \qquad \mathcal{C}_g, \tag{A.1}$$

respectively. From these components, we define the mass (or mole count) vector

$$n = \begin{bmatrix} n_l \\ n_d \\ n_g \end{bmatrix}, \tag{A.2}$$

where n_l is a vector of the components in $\mathcal{C}_l$ and n_d and n_g are vectors of components in $\mathcal{C}_g$ dissolved in the liquid phase and in gas phase, respectively. We define the active reactor volume as a function of the volumes of the gas and liquid phases, as

$$V = V_l + V_g. \tag{A.3}$$

As such, we define the concentrations of the components in the gas and liquid phases, as

$$c = \begin{bmatrix} c_l \\ c_d \\ c_g \end{bmatrix}. \tag{A.4}$$

The concentrations of components dissolved in the liquid phase are computed as

$$c_l = \frac{n_l}{V_l} = \frac{n_l}{(1-\epsilon)V}, \qquad c_d = \frac{n_d}{V_l} = \frac{n_d}{(1-\epsilon)V}, \tag{A.5}$$

where ϵ is the gas volume fraction. The concentrations of components in the gas phase are computed as

$$c_g = \frac{n_g}{V_g} = \frac{n_g}{\epsilon V}, \tag{A.6}$$

such that the gas volume is $V_g = \epsilon V$. Gas-liquid mass transfer is described in detail in [14].

A.1 Mass transfer rate

In chemical and biochemical reactions, we consider volumetric gas-liquid mass transfer rate

$$J_{gl}(c) = k_L a \left(c_{\text{Sat}} - c_d\right), \tag{A.7}$$

where k_La are the volumetric mass transfer coefficients, c_{Sat} are the saturation concentrations, and $(c_{\text{Sat}} - c_d)$ are the driving forces.

mass transfer coefficient - k_La

The mass transfer coefficient, k_La, describes the rate at which components are transferred between liquid and gas phase. The value of k_La depends on, e.g. reactor type, power input, as well as the specific surface (interfacial) area, a. The value of k_La can be determined, e.g. experimentally, estimated using computational fluid dynamics (CFD), or applying model-based techniques (e.g. state or parameter estimation).

saturation factor - γ

The saturation factors, γ, is computed using Henry's law, as

$$\gamma = RT\text{diag}\left(H^{cp}\right)^{-1}, \tag{A.8}$$

where R is the ideal gas constant, T is the temperature, and H^{cp} is vector of Henry's constants for each of the components. A comprehensive list of Henry's constants can be found in [10] and is published online in [11].

gas phase fraction - ϵ

Consider the volume description for a reactor

$$V = V_l + V_g, \tag{A.9}$$

where V is the total reactor volume, V_l is the volume of the liquid phase, and V_g is the volume of the gas phase. The gas phase fraction, ϵ, describes the relative volume fractions of the gas and liquid phases in the reactor. In practice, the value of ϵ will depend on, e.g. reactor type and dimensions, mixing and viscosity, as well as flows of gasses and liquids in the reactor. The gas phase fraction is computed as

$$\epsilon = \frac{V_g}{V} = \frac{V_g}{V_l + V_g}. \tag{A.10}$$

Note that we can compute similar fractions as a function of ϵ from (A.9), e.g. the liquid phase fraction $(1.0 - \epsilon)$ and gas to liquid phase fraction $\epsilon/(1.0 - \epsilon)$. The volume of the liquid phase, V_l, is given by the flows of liquid in and out of the reactor. Evaporation can also be considered in computing the liquid phase volume. We apply the ideal gas law to compute the gas volume as

$$V_g = e^T n_g \frac{RT}{P}, \tag{A.11}$$

where $e^T n_g = \sum n_g$ is the sum of molecules in the gas phase, R is the ideal gas constant, T is the temperature, P is the pressure, and V_g is the gas phase volume. In plug flow reactors, i.e. flow through pipes, at steady state, we compute the gas phase fraction as

$$\epsilon = \frac{F_g}{F_l + F_g}, \tag{A.12}$$

where F_g is the total flow-rate of gas and F_l is the total flow-rate of liquid through the reactor. We assume in all cases that the gasses are dispersed as bubbles in the liquid, discounting head space in the reactor, as the interfacial area is small. In this report, we will assume that temperature and pressure are constant, unless explicitly modelled. As such, the volume will change with the mass of the gasses in the reactor, as described by (A.11).

saturation concentration - c_{Sat}

The saturation concentration, c_{Sat}, describes the equilibrium concentration for the gas phase dissolved in the liquid phase. We compute the saturation concentration as

$$c_{\mathrm{Sat}} = \frac{\epsilon}{1.0 - \epsilon} \gamma c_g. \tag{A.13}$$

APPENDIX B Batch Reactor - Derivations and Implementations

In this chapter, we present Jacobians and implementations in Matlab and Python of the BR model presented in (3.7).

B.1 Jacobians

In this section, we present Jacobians for the batch reactor model (3.7). We consider the right-hand side function

$$f(x, \theta) = R(c)V, \tag{B.1}$$

where $x = n$ and $c = n/V$.

B.1.1 Jacobian of f wrt. x - $\frac{\partial f}{\partial x}$

We describe the Jacobian of the function (B.1) wrt. x, as

$$\begin{aligned} \frac{\partial f}{\partial x} &= \frac{\partial f}{\partial n} && \text{(B.2a)} \\ &= S^T \frac{\partial r}{\partial c}. && \text{(B.2b)} \end{aligned}$$

Jacobian of f wrt. n

$$\begin{aligned} \frac{\partial f}{\partial n} &= \frac{\partial}{\partial n}\left(R(c)V\right) && \text{(B.3a)} \\ &= \frac{\partial R}{\partial c}(c)\frac{\partial c}{\partial n}V && \text{(B.3b)} \\ &= \frac{\partial R}{\partial c}(c)\frac{1}{V}V && \text{(B.3c)} \\ &= \frac{\partial R}{\partial c}. && \text{(B.3d)} \end{aligned}$$

The Jacobian of the production term is

$$\frac{\partial R}{\partial c} = \frac{\partial}{\partial c}\left(S^T r(c)\right) \tag{B.4a}$$
$$= S^T \frac{\partial r}{\partial c}. \tag{B.4b}$$

Thus

$$\frac{\partial f}{\partial n} = S^T \frac{\partial r}{\partial c}. \tag{B.5}$$

B.2 Implementations

B.2.1 Matlab

```
function [f, dfdx, dfdu, dfdd] = driftBR(t, x, u, d, theta)
%------------------------------------------------------------------
%   Authour(s):
%       Marcus Krogh Nielsen
%
%   Email:
%       mkrni@dtu.dk
%
%------------------------------------------------------------------
%   Call:
%       [f, dfdx, dfdu, dfdd] = driftBR(t, x, u, d, theta)
%
%   Description:
%       Drift function for a batch reactor.
%
%   Inputs:
%       t       :     time
%       x       :     states
%       u       :     manipulated variables
%       d       :     disturbances
%       theta   :     parameters
%
%   Outputs:
%       f       :     state drift function
%       dfdx    :     drift derivative wrt. states
%       dfdu    :     drift derivative wrt. manipulated variables
%       dfdd    :     drift derivative wrt. disturbances
%
%------------------------------------------------------------------

%% Parameters

```

```
% Size
nx = theta.nx;
nu = theta.nu;
nd = theta.nd;

% Kinetics
rFun = theta.rFun;
rPar = theta.rPar;
S    = theta.S;

% Parameters
V = theta.V;

% Variables
% ...
% states
n = x;
c = n/V;

%% Function

% Productions rates
[r,  ] = rFun(t, c, rPar);
R      = S'*r;

% Evaluation
f      = R*V;

%% Derivative

if nargout > 1
    % kinetics derivative wrt. concentrations
    [ , drdc] = rFun(t, c, rPar);

    % drift Jacobian wrt. states (x)
    % ...
    % dfdx
    dfdx = S'*drdc;

    % drift Jacobian wrt. inputs (u)
    % ...
    % dfdu
    dfdu = zeros(nx, nu);

    % drift Jacobian wrt. disturbances (d)
    % ...
    % dfdd
    dfdd = zeros(nx, nd);

end

end
```

./code/matlab/driftBR.m.

B.2.2 Python

```
## Imports
import numpy as np


###############################################################################
##    Authour(s):
##        Marcus Krogh Nielsen
##
##    Email:
##        mkrni@dtu.dk
##
###############################################################################
##    Call:
##        f, dfdx, dfdu, dfdd = driftBR(t, x, u, d, theta, nargout)
##
##    Description:
##        Drift function for a batch reactor.
##
##    Inputs:
##        t        :      time
##        x        :      states
##        u        :      manipulated variables
##        d        :      disturbances
##        theta    :      parameters
##
##    Outputs:
##        f        :      state drift function
##        dfdx     :      drift derivative wrt. states
##        dfdu     :      drift derivative wrt. manipulated variables
##        dfdd     :      drift derivative wrt. disturbances
##
###############################################################################
def driftBR(t, x, u, theta, nargout=1):
    ## Parameters

    # Size
    nx = theta.nx
    nu = theta.nu
    nd = theta.nd

    # Kinetics
    rFun = theta.rFun
    rPar = theta.rPar
    S    = theta.S
```

```python

    # Parameters
    V = theta.V

    # Variables
    # ...
    # states
    n = x
    c = n/V


    ## Function

    # Production rates
    r = rFun(t, c, rPar, nargout=1)
    R = S.T@r

    # Evaluate drift term
    f = R*V


    ## Derivatives
    if nargout > 1:
        # Kinetics drivative wrt. concentrations
        _, drdc = rFun(t, x, rPar, nargout=2)

        # Drift Jacobian wrt. states (x)
        # ...
        # dfdx
        dfdx = S.T@drdc

        # Drift Jacobian wrt. inputs (u)
        # ...
        # dfdu
        dfdu = np.zeros(nx, nu)

        # Drift Jacobian wrt. disturbances (d)
        # ...
        # dfdd
        dfdd = np.zeros(nx, nd)

        return f, dfdx, dfdu, dfdd

    # Return statement
    return f
```

./code/python/driftBR.py.

APPENDIX C Fed-batch Reactor - Derivations and Implementations

In this chapter, we present Jacobians and implementations in Matlab and Python of the FBR model presented in (4.12).

C.1 Derivatives

In this section, we present Jacobians for the FBR model (4.12). We consider the right-hand side function

$$f(x,u,\theta) = \begin{bmatrix} f_V(x,u,\theta) \\ f_n(x,u,\theta) \end{bmatrix} \tag{C.1a}$$

$$= \begin{bmatrix} e^T F \\ C_{In} F + R(c)V \end{bmatrix}, \tag{C.1b}$$

where the states $x = [V, n]$, the concentrations $c = n/V$, and $u = F$.

C.1.1 Jacobian of f wrt. x - $\frac{\partial f}{\partial x}$

We describe the Jacobian of the function (C.1) wrt. x, as

$$\frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial f_V}{\partial V} & \frac{\partial f_V}{\partial n} \\ \frac{\partial f_n}{\partial V} & \frac{\partial f_n}{\partial n} \end{bmatrix} \tag{C.2a}$$

$$= \begin{bmatrix} 0 & 0 \\ R(c) - S^T \frac{\partial r}{\partial c} c & S^T \frac{\partial r}{\partial c} \end{bmatrix}. \tag{C.2b}$$

Jacobian of f_V wrt. V

$$\frac{\partial f_V}{\partial V} = \frac{\partial}{\partial V}\left(e^T F\right) \tag{C.3a}$$

$$= 0. \tag{C.3b}$$

Thus

$$\frac{\partial f_V}{\partial V} = 0. \tag{C.4}$$

Jacobian of f_V wrt. n

$$\begin{aligned}\frac{\partial f_V}{\partial n} &= \frac{\partial}{\partial n}\left(e^T F\right) &\text{(C.5a)}\\ &= 0. &\text{(C.5b)}\end{aligned}$$

Thus

$$\frac{\partial f_V}{\partial n} = 0. \tag{C.6}$$

Jacobian of f_n wrt. V

$$\begin{aligned}\frac{\partial f_n}{\partial V} &= \frac{\partial}{\partial V}\left(C_{In}F + R(c)V\right) &\text{(C.7a)}\\ &= \frac{\partial}{\partial V}\left(C_{In}F\right) + \frac{\partial}{\partial V}\left(R(c)V\right). &\text{(C.7b)}\end{aligned}$$

The Jacobian of the inlet flow term is

$$\frac{\partial}{\partial V}\left(C_{In}F\right) = 0. \tag{C.8}$$

The Jacobian of the production term is

$$\begin{aligned}\frac{\partial}{\partial V}\left(R(c)V\right) &= \frac{\partial R}{\partial V}V + R(c)\frac{\partial V}{\partial V} &\text{(C.9a)}\\ &= \frac{\partial R}{\partial V}V + R(c). &\text{(C.9b)}\end{aligned}$$

The Jacobian of the production function is

$$\begin{aligned}\frac{\partial R}{\partial V} &= \frac{\partial R}{\partial c}\frac{\partial c}{\partial V} &\text{(C.10a)}\\ &= S^T\frac{\partial r}{\partial c}\frac{\partial c}{\partial V} &\text{(C.10b)}\\ &= -S^T\frac{\partial r}{\partial c}\frac{n}{V^2}. &\text{(C.10c)}\end{aligned}$$

Thus

$$\begin{aligned}\frac{\partial f_n}{\partial V} &= R(c) - S^T\frac{\partial r}{\partial c}\frac{n}{V^2}V &\text{(C.11a)}\\ &= R(c) - S^T\frac{\partial r}{\partial c}\frac{n}{V} &\text{(C.11b)}\\ &= R(c) - S^T\frac{\partial r}{\partial c}c. &\text{(C.11c)}\end{aligned}$$

Jacobian of f_n wrt. n

$$\frac{\partial f_n}{\partial n} = \frac{\partial}{\partial n}\left(C_{In}F + R(c)V\right) \tag{C.12a}$$
$$= \frac{\partial}{\partial n}\left(C_{In}F\right) + \frac{\partial}{\partial n}\left(R(c)V\right). \tag{C.12b}$$

The Jacobian of the inlet flow term is

$$\frac{\partial}{\partial n}\left(C_{In}F\right) = 0. \tag{C.13}$$

The Jacobian of the production term is

$$\frac{\partial}{\partial n}\left(R(c)V\right) = \frac{\partial R}{\partial n}V \tag{C.14a}$$
$$= S^T\frac{\partial r}{\partial n}V \tag{C.14b}$$
$$= S^T\frac{\partial r}{\partial c}\frac{\partial c}{\partial n}V \tag{C.14c}$$
$$= S^T\frac{\partial r}{\partial c}\frac{1}{V}V \tag{C.14d}$$
$$= S^T\frac{\partial r}{\partial c}. \tag{C.14e}$$

Thus

$$\frac{\partial f_n}{\partial n} = S^T\frac{\partial r}{\partial c}. \tag{C.15}$$

C.1.2 Jacobian of f wrt. u - $\frac{\partial f}{\partial u}$

We describe the Jacobian of the function (C.1) wrt. u, as

$$\frac{\partial f}{\partial u} = \begin{bmatrix} \frac{\partial f_V}{\partial F} \\ \frac{\partial f_n}{\partial F} \end{bmatrix} \tag{C.16a}$$
$$= \begin{bmatrix} e^T \\ C_{In} \end{bmatrix}. \tag{C.16b}$$

Jacobian of f_V wrt. F

$$\frac{\partial f_V}{\partial F} = \frac{\partial}{\partial F}\left(e^T F\right) \tag{C.17a}$$
$$= e^T\frac{\partial F}{\partial F} \tag{C.17b}$$
$$= e^T. \tag{C.17c}$$

Thus

$$\frac{\partial f_V}{\partial F} = e^T. \tag{C.18}$$

Jacobian of f_n wrt. F

$$\frac{\partial f_n}{\partial F} = \frac{\partial}{\partial F}\left(C_{In}F + R(c)V\right) \tag{C.19a}$$
$$= \frac{\partial}{\partial F}\left(C_{In}F\right) + \frac{\partial}{\partial F}\left(R(c)V\right). \tag{C.19b}$$

The Jacobian of the inlet flow term is

$$\frac{\partial}{\partial F}\left(C_{In}F\right) = C_{In}\frac{\partial F}{\partial F} \tag{C.20a}$$
$$= C_{In}. \tag{C.20b}$$

The Jacobian of the production term is

$$\frac{\partial}{\partial F}\left(R(c)V\right) = 0. \tag{C.21}$$

Thus

$$\frac{\partial f_n}{\partial F} = C_{In}. \tag{C.22}$$

C.2 Implementations

C.2.1 Matlab

```matlab
function [f, dfdx, dfdu, dfdd] = driftFBR(t, x, u, d, theta)
%--------------------------------------------------------------------------
%   Authour(s):
%       Marcus Krogh Nielsen
%
%   Email:
%       mkrni@dtu.dk
%
%--------------------------------------------------------------------------
%   Call:
%       [f, dfdx, dfdu, dfdd] = driftFBR(t, x, u, d, theta)
%
%   Description:
%       Drift function for a fedbatch reactor.
%
%   Inputs:
%       t       :       time
%       x       :       states
%       u       :       manipulated variables
%       d       :       disturbances
%       theta   :       parameters
```

```
%
%   Outputs:
%       f       :     state drift function
%       dfdx    :     drift derivative wrt. states
%       dfdu    :     drift derivative wrt. manipulated variables
%       dfdd    :     drift derivative wrt. disturbances
%
%-----------------------------------------------------------------

%% Parameters

% Size
nx = theta.nx;
nu = theta.nu;
nd = theta.nd;

% Parameters
CIn = theta.CIn;
e   = ones(nu, 1);

% Kinetics
rFun = theta.rFun;
rPar = theta.rPars;
S    = theta.S;

% Variables
% ...
% states
V = x(1);
n = x(2:end);
c = n/V;
% inputs
F = u;


%% Function

% Productions rates
[r,  ] = rFun(t, c, rPar);
R      = S'*r;

% Evaluation
f = [e'*F       ;
     CIn*F + R*V];

%% Derivative

if nargout > 1
    % kinetics derivative wrt. concentrations
    [ , drdc] = rFun(t, c, rPar);

    % drift Jacobian wrt. states (x)
    % ...
    % pre-allocation
    dfdx = zeros(nx, nx);
    % dfVdV
    dfdx(1      ,1      ) = 0.0;
    % dfVdn
    dfdx(1      ,2:end) = 0.0;
    % dfndV
    dfdx(2:end,1      ) = S'*drdc*n/V + R;
    % dfndn
    dfdx(2:end,2:end) = S'*drdc;

    % drift Jacobian wrt. inputs (u)
    % ...
    % pre-allocation;
    dfdu = zeros(nx, nu);
    % dfVdF
    dfdu(1      ,:) = e';
    % dfndF
    dfdu(2:end ,:) = CIn;

    % drift Jacobian wrt. disturbances (d)
    % ...
    % pre-allocation
    dfdd = zeros(nx, nd);
    % dfdd


end

end
```

./code/matlab/driftFBR.m.

C.2.2 Python

```
## Imports
import numpy as np


###############################################################################
##    Authour(s):
##         Marcus Krogh Nielsen
##
##    Email:
##         mkrni@dtu.dk
##
###############################################################################
##    Call:
```

```
##        f, dfdx, dfdu, dfdd = driftFBR(t, x, u, d, theta, nargout)
##
##   Description:
##       Druft function for a fedbatch reactor.
##
##   Inputs:
##       t        :     time
##       x        :     states
##       u        :     manipulated variables
##       d        :     disturbances
##       theta    :     parameters
##
##   Outputs:
##       f        :     state drift function
##       dfdx     :     drift derivative wrt. states
##       dfdu     :     drift derivative wrt. manipulated variables
##       dfdd     :     drift derivative wrt. disturbances
##
############################################################################
def driftFBR(t, x, u, theta, nargout=1):
    ## Parameters

    # Size
    nx = theta.nx
    nu = theta.nu
    nd = theta.nd

    # Inlet concentrations
    CIn = theta.CIn
    e   = np.ones(nu, 1)

    # Kinetics
    rFun = theta.rFun
    rPar = theta.rPar
    S    = theta.S

    # Variables
    # ...
    # states
    V = x[0]
    n = x[1:]
    c = n/V
    # inputs
    F = u


    ## Function

    # Production rates
    r, _ = rFun(t, c, rPar)
    R    = S.T@r

```

```python
    # Evaluate drift term
    # ...
    # fV
    fV = e.T@F
    # fn
    fn = CIn@F + S.T@R
    # f
    f = np.array([[fV], [fn]])


    ## Derivatives
    if nargout > 1:
        # Kinetics drivative wrt. concentrations
        _, drdc = rFun(t, x, rPar)

        # Drift Jacobian wrt. states (x)
        # ...
        # pre-allocation
        dfdx = np.zeros(nx, nx)
        # dfVdV
        dfdx[0  ,0  ] = 0.0
        # dfVdn
        dfdx[1:,0  ] = 0.0
        # dfndV
        dfdx[0  ,1:] = S.T@drdc@c + R
        # dfndn
        dfdx[1:,1:] = S.T@drdc

        # Drift Jacobian wrt. inputs (u)
        # ...
        # pre-allocation
        dfdu = np.zeros(nx, nu)
        # dfVdF
        dfdu[0  ,:] = e.T
        # dfndF
        dfdu[1:,:] = CIn

        # Drift Jacobian wrt. disturbances (d)
        # ...
        # dfdd
        dfdd = np.zeros(nx, nd)

        return f, dfdx, dfdu, dfdd

    # Return statement
    return f
```

./code/python/driftFBR.py.

APPENDIX D Continuous Stirred Tank Reactor - Derivations and Implementations

In this chapter, we present Jacobians and implementations in Matlab and Python of the CSTR model presented in (5.12).

D.1 Derivatives - variable volume

In this section, we present Jacobians for the CSTR model (5.12). We consider the right-hand side function

$$f(x,u,\theta) = \begin{bmatrix} f_V(x,u,\theta) \\ f_n(x,u,\theta) \end{bmatrix} \tag{D.1a}$$

$$= \begin{bmatrix} e^T F - e_{\text{Out}}^T F_{\text{Out}} \\ C_{\text{In}} F - c e_{\text{Out}}^T F_{\text{Out}} + R(c)V \end{bmatrix}, \tag{D.1b}$$

where the states $x = [V, n]$, the concentrations $c = n/V$, and $u = [F, F_{\text{Out}}]$.

D.1.1 Jacobian of f wrt. x - $\frac{\partial f}{\partial x}$

We describe the Jacobian of the function (D.1) wrt. x, as

$$\frac{\partial f}{\partial x} = \begin{bmatrix} \frac{\partial f_V}{\partial V} & \frac{\partial f_V}{\partial n} \\ \frac{\partial f_n}{\partial V} & \frac{\partial f_n}{\partial n} \end{bmatrix} \tag{D.2a}$$

$$= \begin{bmatrix} 0 & 0 \\ \frac{c}{V} e_{\text{Out}}^T F_{\text{Out}} - S^T \frac{\partial r}{\partial c} c + R(c) & S^T \frac{\partial r}{\partial c} - I_{nx-1} \frac{1}{V} e_{\text{Out}}^T F \end{bmatrix}. \tag{D.2b}$$

Jacobian of f_V wrt. V

$$\frac{\partial f_V}{\partial V} = \frac{\partial}{\partial V}\left(e^T F - e_{\text{Out}}^T F_{\text{Out}}\right) \tag{D.3a}$$
$$= 0. \tag{D.3b}$$

Thus

$$\frac{\partial f_V}{\partial V} = 0. \tag{D.4}$$

Jacobian of f_V wrt. n

$$\frac{\partial f_V}{\partial n} = \frac{\partial}{\partial n}\left(e^T F - e_{\text{Out}}^T F_{\text{Out}}\right) \tag{D.5a}$$
$$= 0. \tag{D.5b}$$

Thus

$$\frac{\partial f_V}{\partial n} = 0. \tag{D.6}$$

Jacobian of f_n wrt. V

$$\frac{\partial f_n}{\partial V} = \frac{\partial}{\partial V}\left(C_{\text{In}} F - c e_{\text{Out}}^T F_{\text{Out}} + R(c)V\right) \tag{D.7a}$$
$$= \frac{\partial}{\partial V}\left(C_{\text{In}} F\right) - \frac{\partial}{\partial V}\left(c e_{\text{Out}}^T F_{\text{Out}}\right) + \frac{\partial}{\partial V}\left(R(c)V\right). \tag{D.7b}$$

The Jacobian of the inlet flow term is

$$\frac{\partial}{\partial V}\left(C_{\text{In}} F\right) = 0. \tag{D.8}$$

The Jacobian of the outlet flow term is

$$\frac{\partial}{\partial V}\left(c e_{\text{Out}}^T F_{\text{Out}}\right) = \frac{\partial c}{\partial V} e_{\text{Out}}^T F_{\text{Out}} \tag{D.9a}$$
$$= -\frac{n}{V^2} e_{\text{Out}}^T F_{\text{Out}} \tag{D.9b}$$
$$= -\frac{c}{V} e_{\text{Out}}^T F_{\text{Out}}. \tag{D.9c}$$

The Jacobian of the production term is

$$\begin{aligned}
\frac{\partial}{\partial V}\left(R(c)V\right) &= \frac{\partial R}{\partial V}V + R(c)\frac{\partial V}{\partial V} && \text{(D.10a)}\\
&= S^T\frac{\partial r}{\partial V}V + R(c) && \text{(D.10b)}\\
&= S^T\frac{\partial r}{\partial c}\frac{\partial c}{\partial V}V + R(c) && \text{(D.10c)}\\
&= -S^T\frac{\partial r}{\partial c}\frac{n}{V^2}V + R(c) && \text{(D.10d)}\\
&= -S^T\frac{\partial r}{\partial c}\frac{n}{V} + R(c) && \text{(D.10e)}\\
&= -S^T\frac{\partial r}{\partial c}c + R(c). && \text{(D.10f)}
\end{aligned}$$

Thus

$$\frac{\partial f_n}{\partial V} = \frac{c}{V}e_{\text{Out}}^T F_{\text{Out}} - S^T\frac{\partial r}{\partial c}c + R(c). \tag{D.11}$$

Jacobian of f_n wrt. n

$$\begin{aligned}
\frac{\partial f_n}{\partial n} &= \frac{\partial}{\partial n}\Big(C_{\text{In}}F - ce_{\text{Out}}^T F_{\text{Out}} + R(c)V\Big) && \text{(D.12a)}\\
&= \frac{\partial}{\partial n}\left(C_{\text{In}}F\right) - \frac{\partial}{\partial n}\Big(ce_{\text{Out}}^T F_{\text{Out}}\Big) + \frac{\partial}{\partial n}\left(R(c)V\right). && \text{(D.12b)}
\end{aligned}$$

The Jacobian of the inlet flow term is

$$\frac{\partial}{\partial n}\left(C_{\text{In}}F\right) = 0. \tag{D.13}$$

The Jacobian of the outlet flow term is

$$\begin{aligned}
\frac{\partial}{\partial n}\Big(ce_{\text{Out}}^T F_{\text{Out}}\Big) &= \frac{\partial c}{\partial n}e_{\text{Out}}^T F_{\text{Out}} && \text{(D.14a)}\\
&= -I_{nx-1}\frac{1}{V}e_{\text{Out}}^T F_{\text{Out}}. && \text{(D.14b)}
\end{aligned}$$

The Jacobian of the production term is

$$\begin{aligned}
\frac{\partial}{\partial n}\left(R(c)V\right) &= \frac{\partial R}{\partial n}V && \text{(D.15a)}\\
&= S^T\frac{\partial r}{\partial n}V && \text{(D.15b)}\\
&= S^T\frac{\partial r}{\partial c}\frac{\partial c}{\partial n}V && \text{(D.15c)}\\
&= S^T\frac{\partial r}{\partial c}\frac{1}{V}V && \text{(D.15d)}\\
&= S^T\frac{\partial r}{\partial c}. && \text{(D.15e)}
\end{aligned}$$

Thus

$$\frac{\partial f_n}{\partial n} = \frac{c}{V} e_{\text{Out}}^T F_{\text{Out}} - S^T \frac{\partial r}{\partial c} - R(c). \tag{D.16}$$

D.1.2 Jacobian of f wrt. u - $\frac{\partial f}{\partial u}$

We describe the Jacobian of the function (D.1) wrt. u, as

$$\begin{aligned} \frac{\partial f}{\partial u} &= \begin{bmatrix} \frac{\partial f_V}{\partial F} & \frac{\partial f_V}{\partial F_{\text{Out}}} \\ \frac{\partial f_n}{\partial F} & \frac{\partial f_n}{\partial F_{\text{Out}}} \end{bmatrix} &\text{(D.17a)} \\ &= \begin{bmatrix} e^T & -e_{\text{Out}}^T \\ C_{\text{In}} & -c e_{\text{Out}}^T \end{bmatrix}. &\text{(D.17b)} \end{aligned}$$

Jacobian of f_V wrt. F

$$\begin{aligned} \frac{\partial f_V}{\partial F} &= \frac{\partial}{\partial F}\left(e^T F - e_{\text{Out}}^T F_{\text{Out}}\right) &\text{(D.18a)} \\ &= \frac{\partial}{\partial F}\left(e^T F\right) - \frac{\partial}{\partial F}\left(e_{\text{Out}}^T F_{\text{Out}}\right). &\text{(D.18b)} \end{aligned}$$

The Jacobian of the inlet flow term is

$$\begin{aligned} \frac{\partial}{\partial F}\left(e^T F\right) &= e^T \frac{\partial F}{\partial F} &\text{(D.19)} \\ &= e^T. &\text{(D.20)} \end{aligned}$$

The Jacobian of the outlet flow term is

$$\frac{\partial}{\partial F}\left(e_{\text{Out}}^T F_{\text{Out}}\right) = 0. \tag{D.21}$$

Thus

$$\frac{\partial f_V}{\partial F} = e^T. \tag{D.22}$$

Jacobian of f_V wrt. F_{Out}

$$\begin{aligned} \frac{\partial f_V}{\partial F_{\text{Out}}} &= \frac{\partial}{\partial F_{\text{Out}}}\left(e^T F - e_{\text{Out}}^T F_{\text{Out}}\right) &\text{(D.23a)} \\ &= \frac{\partial}{\partial F_{\text{Out}}}\left(e^T F\right) - \frac{\partial}{\partial F_{\text{Out}}}\left(e_{\text{Out}}^T F_{\text{Out}}\right). &\text{(D.23b)} \end{aligned}$$

The Jacobian of the inlet flow term is

$$\frac{\partial}{\partial F_{\text{Out}}}\left(e^T F\right) = 0. \tag{D.24a}$$

The Jacobian of the outlet flow term is

$$\frac{\partial}{\partial F_{\text{Out}}}\left(e_{\text{Out}}^T F_{\text{Out}}\right) = e_{\text{Out}}^T \frac{\partial F_{\text{Out}}}{\partial F_{\text{Out}}} \tag{D.25a}$$

$$= e_{\text{Out}}^T. \tag{D.25b}$$

Thus

$$\frac{\partial f_V}{\partial F_{\text{Out}}} = -e_{\text{Out}}^T. \tag{D.26}$$

Jacobian of f_n wrt. F

$$\frac{\partial f_n}{\partial F} = \frac{\partial}{\partial F}\left(C_{\text{In}}F - ce_{\text{Out}}^T F_{\text{Out}} + R(c)V\right) \tag{D.27a}$$

$$= \frac{\partial}{\partial F}\left(C_{\text{In}}F\right) - \frac{\partial}{\partial F}\left(ce_{\text{Out}}^T F_{\text{Out}}\right) + \frac{\partial}{\partial F}\left(R(c)V\right). \tag{D.27b}$$

The Jacobian of the inlet flow term is

$$\frac{\partial}{\partial F}\left(C_{\text{In}}F\right) = C_{\text{In}}\frac{\partial F}{\partial F} \tag{D.28a}$$

$$= C_{\text{In}}. \tag{D.28b}$$

The Jacobian of the outlet flow term is

$$\frac{\partial}{\partial F}\left(ce_{\text{Out}}^T F_{\text{Out}}\right) = 0. \tag{D.29}$$

The Jacobian of the production term is

$$\frac{\partial}{\partial F}\left(R(c)V\right) = 0. \tag{D.30}$$

Thus

$$\frac{\partial f_n}{\partial F} = C_{\text{In}}. \tag{D.31}$$

Jacobian of f_n wrt. F_{Out}

$$\frac{\partial f_n}{\partial F_{\text{Out}}} = \frac{\partial}{\partial F_{\text{Out}}}\left(C_{\text{In}}F - ce_{\text{Out}}^T F_{\text{Out}} + R(c)V\right) \tag{D.32a}$$

$$= \frac{\partial}{\partial F_{\text{Out}}}\left(C_{\text{In}}F\right) - \frac{\partial}{\partial F_{\text{Out}}}\left(ce_{\text{Out}}^T F_{\text{Out}}\right) + \frac{\partial}{\partial F_{\text{Out}}}\left(R(c)V\right). \tag{D.32b}$$

The Jacobian of the inlet flow term is

$$\frac{\partial}{\partial F_{\text{Out}}}\left(C_{\text{In}}F\right) = 0. \tag{D.33}$$

The Jacobian of the outlet flow term is

$$\frac{\partial}{\partial F_{\text{Out}}} = \frac{\partial}{\partial F_{\text{Out}}}\left(ce_{\text{Out}}^T F_{\text{Out}}\right) \tag{D.34a}$$

$$= ce_{\text{Out}}^T \frac{\partial F_{\text{Out}}}{\partial F_{\text{Out}}} \tag{D.34b}$$

$$= ce_{\text{Out}}^T. \tag{D.34c}$$

The Jacobian of the production term is

$$\frac{\partial}{\partial F_{\text{Out}}}\left(R(c)V\right) = 0. \tag{D.35}$$

Thus

$$\frac{\partial f_n}{\partial F_{\text{Out}}} = -ce_{\text{Out}}^T. \tag{D.36}$$

D.2 Derivative - constant volume

In this section, we present Jacobians for the CSTR model (5.15). We consider the right-hand side function

$$f(t, x, u, \theta) = \left(C_{\text{In}} - ce^T\right) F + R(c)V, \tag{D.37}$$

where the states $x = n$, the concentrations $c = n/V$, and the inputs $u = F$.

D.2.1 Jacobian of f wrt. x - $\frac{\partial f}{\partial x}$

We describe the Jacobian of the function (D.37) wrt. x, as

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial n} \tag{D.38a}$$

$$= -\frac{1}{V}e^T F + S^T \frac{\partial r}{\partial c}. \tag{D.38b}$$

Jacobian of f wrt. n

$$\frac{\partial f}{\partial n} = \frac{\partial}{\partial n}\left(\left(C_{\text{In}} - ce^T\right) F + R(c)V\right) \tag{D.39a}$$

$$= \frac{\partial}{\partial n}\left(\left(C_{\text{In}} - ce^T\right) F\right) + \frac{\partial}{\partial n}\left(R(c)V\right). \tag{D.39b}$$

$$\frac{\partial}{\partial n}\left(\left(C_{\text{In}} - ce^T\right) F\right) = \frac{\partial}{\partial n}\left(C_{\text{In}}F\right) - \frac{\partial}{\partial n}\left(ce^T F\right) \tag{D.40a}$$

$$= -\frac{\partial}{\partial c}\left(ce^T F\right)\frac{\partial c}{\partial n} \tag{D.40b}$$

$$= -I\frac{1}{V}e^T F. \tag{D.40c}$$

$$\frac{\partial}{\partial n}\left(R(c)V\right) = \frac{\partial R}{\partial n}V \tag{D.41a}$$
$$= S^T\frac{\partial r}{\partial n}V \tag{D.41b}$$
$$= S^T\frac{\partial r}{\partial c}\frac{\partial c}{\partial n}V \tag{D.41c}$$
$$= S^T\frac{\partial r}{\partial c}\frac{1}{V}V \tag{D.41d}$$
$$= S^T\frac{\partial r}{\partial c}. \tag{D.41e}$$

Thus

$$\frac{\partial f}{\partial n} = -\frac{1}{V}e^T F + S^T\frac{\partial r}{\partial c}. \tag{D.42}$$

D.2.2 Jacobian of f wrt. u - $\frac{\partial f}{\partial u}$

We describe the Jacobian of the function (D.37) wrt. u, as

$$\frac{\partial f}{\partial u} = \frac{\partial f}{\partial F} \tag{D.43a}$$
$$= C_{\text{In}} - ce^T. \tag{D.43b}$$

Jacobian of f wrt. F

$$\frac{\partial f}{\partial F} = \frac{\partial}{\partial F}\left(\left(C_{\text{In}} - ce^T\right)F + R(c)V\right) \tag{D.44a}$$
$$= \frac{\partial}{\partial F}\left(\left(C_{\text{In}} - ce^T\right)F\right) + \frac{\partial}{\partial F}\left(R(c)V\right). \tag{D.44b}$$

$$\frac{\partial}{\partial F}\left(\left(C_{\text{In}} - ce^T\right)F\right) = \left(C_{\text{In}} - ce^T\right)\frac{\partial F}{\partial F} \tag{D.45a}$$
$$= C_{\text{In}} - ce^T. \tag{D.45b}$$

$$\frac{\partial}{\partial F}\left(R(c)V\right) = 0. \tag{D.46}$$

Thus

$$\frac{\partial f}{\partial F} = C_{\text{In}} - ce^T. \tag{D.47}$$

D.3 Implementations

D.3.1 Matlab - variable volume

```
function [f, dfdx, dfdu, dfdd] = driftCSTR(t, x, u, d, theta)
%-----------------------------------------------------------------------
%   Authour(s):
%       Marcus Krogh Nielsen
%
%   Email:
%       mkrni@dtu.dk
%
%-----------------------------------------------------------------------
%   Call:
%       [f, dfdx, dfdu, dfdd] = driftFBR(t, x, u, d, theta)
%
%   Description:
%       Drift function for a continuous stirred tank reactor.
%
%   Inputs:
%       t       :     time
%       x       :     states
%       u       :     manipulated variables
%       d       :     disturbances
%       theta   :     parameters
%
%   Outputs:
%       f       :     state drift function
%       dfdx    :     drift derivative wrt. states
%       dfdu    :     drift derivative wrt. manipulated variables
%       dfdd    :     drift derivative wrt. disturbances
%
%-----------------------------------------------------------------------

%% Parameters

% Size
nx    = theta.nx;
nu    = theta.nu;
nuIn  = theta.nuIn;
nuOut = theta.nuOut;
nd    = theta.nd;

% Parameters
CIn   = theta.CIn;
e     = ones(nuIn , 1);
eOut = ones(nuOut, 1);

% Kinetics
rFun = theta.rFun;
```

```matlab
rPar = theta.rPars;
S    = theta.S;

% Variables
% ...
% states
V    = x(1);
n    = x(2:end);
c    = n/V;
% inputs
F    = u(1:nuIn);
FOut = u(nuIn+1:end);


%% Function

% Productions rates
[r,  ] = rFun(t, c, rPar);
R      = S'*r;

% Evaluation
f = [e'*F   - eOut'*FOut         ;
     CIn*F  - c*eOut'*FOut + R*V];

%% Derivative

if nargout > 1
    % kinetics derivative wrt. concentrations
    [ , drdc] = rFun(t, c, rPar);

    % drift Jacobian wrt. states (x)
    % ...
    % pre-allocation
    dfdx = zeros(nx, nx);
    % dfVdV
    dfdx(1     ,1     ) = 0.0;
    % dfVdn
    dfdx(1     ,2:end) = 0.0;
    % dfndV
    dfdx(2:end,1     ) = c/V*eOut'*FOut - S'*drdc*c + R;
    % dfndn
    Inn = eye(nx-1);
    dfdx(2:end,2:end) = S'*drdc - Inn*1/V*(eOut'*FOut);

    % drift Jacobian wrt. inputs (u)
    % ...
    % pre-allocation;
    dfdu = zeros(nx, nu);
    % dfVdF
    dfdu(1     ,1:nuIn     ) = e';
    % dfVdFOut
    dfdu(1     ,nuIn+1:end) = - eOut';
    % dfndF
    dfdu(2:end,1:nuIn     ) = CIn;
    % dfndFOut
    dfdu(2:end,nuIn+1:end) = - c*eOut';

    % drift Jacobian wrt. disturbances (d)
    % ...
    % pre-allocation
    dfdd = zeros(nx, nd);
    % dfdd


end

end
```

./code/matlab/driftCSTR.m.

D.3.2 Python - variable volume

```
## Imports
import numpy as np


##############################################################################
##   Authour(s):
##       Marcus Krogh Nielsen
##
##   Email:
##       mkrni@dtu.dk
##
##############################################################################
##   Call:
##       f, dfdx, dfdu, dfdd = driftCSTR(t, x, u, d, theta, nargout)
##
##   Description:
##       Drift function for a continuous stirred tank reactor.
##
##   Inputs:
##       t        :     time
##       x        :     states
##       u        :     manipulated variables
##       d        :     disturbances
##       theta    :     parameters
##
##   Outputs:
##       f        :     state drift function
##       dfdx     :     drift derivative wrt. states
##       dfdu     :     drift derivative wrt. manipulated variables
##        dfdd     :      drift  derivative  wrt.  disturbances
##
###############################################################################
def driftCSTR(t, x, u, theta, nargout=1):
    ## Parameters

    # Size
    nx     = theta.nx
    nu     = theta.nu
    nuIn   = theta.nuIn
    nuOut  = theta.nuOut
    nd     = theta.nd

    # Inlet concentrations
    CIn   = theta.CIn
    e     = np.ones(nuIn , 1)
    eOut = np.ones(nuOut, 1)

    # Kinetics
    rFun = theta.rFun
    rPar = theta.rPar
    S    = theta.S

    # Variables
    # ...
    # states
    V = x[0]
    n = x[1:]
    c = n/V
    # inputs
    F    = u[:nuIn]
    FOut = u[nuIn:]


    ## Function

    # Production rates
    r, _ = rFun(t, c, rPar)
    R    = S.T@r

    # Evaluate drift term
    # ...
    # fV
    fV = e.T@F - eOut.T@FOut
    # fn
    fn = CIn@F - c@eOut.T@FOut + R*V
    # f
    f = np.array([[fV], [fn]])


    ## Derivatives
    if nargout > 1:
        # Kinetics drivative wrt. concentrations
        _, drdc = rFun(t, x, rPar)

        # Drift Jacobian wrt. states (x)
        # ...
        # pre-allocation
        dfdx = np.zeros(nx, nx)
        # dfVdV
        dfdx[0  ,0  ] = 0.0
        # dfVdn
        dfdx[0  ,1:] = 0.0
        # dfndV
        dfdx[1:,0  ] = (c/V)@eOut.T@FOut - S.T@drdc@c + R
        # dfndn
    Inn = np.eye(nx-1)
        dfdx[1:,1:] = S.T@drdc - Inn*1/V*(eOut.T@F)

        # Drift Jacobian wrt. inputs (u)
        # ...
        # pre-allocation
        dfdu = np.zeros(nx, nu)
        # dfVdF
        dfdu[0  ,:nuIn] = e.T
        # dfVdFOut
        dfdu[0  ,nuIn:] = - eOut.T
        # dfndF
        dfdu[1:,:nuIn] = CIn
        # dfndFOut
        dfdu[1:,nuIn:] = - c@eOut.T

        # Drift Jacobian wrt. disturbances (d)
        # ...
        # dfdd
        dfdd = np.zeros(nx, nd)

        return f, dfdx, dfdu, dfdd

    # Return statement
    return f
```

./code/python/driftCSTR.py.

D.3.3 Matlab - constant volume

```
function [f, dfdx, dfdu, dfdd] = driftCSTRConstVol(t, x, u, d, theta)
%---------------------------------------------------------------------
%   Authour(s):
%       Marcus Krogh Nielsen
%
```

```
%   Email:
%       mkrni@dtu.dk
%
%-----------------------------------------------------------------------
%   Call:
%       [f, dfdx, dfdu, dfdd] = driftCSTRConstVol(t, x, u, d, theta)
%
%   Description:
%       Drift function for a continuous stirred tank reactor with
%   constant volume.
%
%   Inputs:
%       t       :     time
%       x       :     states
%       u       :     manipulated variables
%       d       :     disturbances
%       theta   :     parameters
%
%   Outputs:
%       f       :     state drift function
%       dfdx    :     drift derivative wrt. states
%       dfdu    :     drift derivative wrt. manipulated variables
%       dfdd    :     drift derivative wrt. disturbances
%
%-----------------------------------------------------------------------

%% Parameters

% Size
nx     = theta.nx;
nu     = theta.nu;
nd     = theta.nd;

% Parameters
V     = theta.V;
CIn   = theta.CIn;
e     = ones(nu, 1);

% Kinetics
rFun = theta.rFun;
rPar = theta.rPars;
S     = theta.S;

% Variables
% ...
% states
n     = x;
c     = n/V;
% inputs
F     = u;


```

```
%% Function

% Productions rates
[r,  ] = rFun(t, c, rPar);
R      = S'*r;

% Evaluation
f = (CIn - c*e')*F + R*V;


%% Derivative

if nargout > 1
    % kinetics derivative wrt. concentrations
    [ , drdc] = rFun(t, c, rPar);

    % drift Jacobian wrt. states (x)
    % ...
    % dfdn
    I    = eye(nx);
    dfdx = - I*1/V*(e'*F) + S'*drdc;

    % drift Jacobian wrt. inputs (u)
    % ...
    % dfdF
    dfdu = CIn - c*e';

    % drift Jacobian wrt. disturbances (d)
    % ...
    % dfdd
    dfdd = zeros(nx, nd);

end

end
```

./code/matlab/driftCSTRConstVol.m.

D.3.4 Python - constant volume

```
## Imports
import numpy as np


##############################################################################
##    Authour(s):
##        Marcus Krogh Nielsen
##
##    Email:
```

```
##        mkrni@dtu.dk
##
################################################################################
##    Call:
##        f, dfdx, dfdu, dfdd = driftCSTRConstVol(t, x, u, d, theta,
    nargout)
##
##    Description:
##        Drift function for a continuous stirred tank reactor with
##    constant volume.
##
##    Inputs:
##        t        :      time
##        x        :      states
##        u        :      manipulated variables
##        d        :      disturbances
##        theta    :      parameters
##
##    Outputs:
##        f        :      state drift function
##        dfdx     :      drift derivative wrt. states
##        dfdu     :      drift derivative wrt. manipulated variables
##        dfdd     :      drift derivative wrt. disturbances
##
################################################################################
def driftCSTRConstVol(t, x, u, theta, nargout=1):
    ## Parameters

    # Size
    nx    = theta.nx
    nu    = theta.nu
    nd    = theta.nd

    # Inlet concentrations
    V    = theta.V
    CIn  = theta.CIn
    e    = np.ones(nu, 1)

    # Kinetics
    rFun = theta.rFun
    rPar = theta.rPar
    S    = theta.S

    # Variables
    # ...
    # states
    n = x
    c = n/V
    # inputs
    F = u


    ## Function

    # Production rates
    r, _ = rFun(t, c, rPar)
    R    = S.T@r

    # Evaluate drift term
    # ...
    # fn
    fn = (CIn - c@e.T)@F + R*V
    # f
    f = np.array([[fn]])


    ## Derivatives
    if nargout > 1:
        # Kinetics drivative wrt. concentrations
        _, drdc = rFun(t, x, rPar)

        # Drift Jacobian wrt. states (x)
        # ...
    # dfdn
    Inx  = np.eye(nx)
        dfdx = - Inx*1/V*(eOut.T@F) + S.T@drdc

        # Drift Jacobian wrt. inputs (u)
        # ...
        # dfdF
        dfdu = CIn - c@e.T

        # Drift Jacobian wrt. disturbances (d)
        # ...
        # dfdd
        dfdd = np.zeros(nx, nd)

        return f, dfdx, dfdu, dfdd

    # Return statement
    return f
```

./code/python/driftCSTRConstVol.py.

APPENDIX E

Plug Flow Reactor - Derivations and Implementations

In this chapter, we present Jacobians for the PDE and finite-volume discretised ODE models, as well as implementations in Matlab and Python of the finite-volume discretisation model presented in (6.24).

E.1 Derivatives - PDE Model

In this section, we present Jacobians for the PFR model in (6.9). We consider the right-hand side function

$$f(x,u,\theta) = -\frac{\partial N}{\partial z} + R(c), \tag{E.1}$$

where

$$N(t,z) = v(t)c(t,z) - D\frac{\partial c}{\partial z}(t,z), \qquad v(t) = \frac{F_t(t)}{A}, \tag{E.2a}$$

where the states $x = c(t,z)$ and the manipulated variables $u = F(t)$.

E.1.1 Jacobian of f wrt. x - $\frac{\partial f}{\partial x}$

We describe the Jacobian of the function (E.1) wrt. x, as

$$\begin{aligned}\frac{\partial f}{\partial x} &= \frac{\partial f}{\partial c} &\text{(E.3a)}\\ &= -v(t)\frac{\partial}{\partial c}\frac{\partial c}{\partial z}(t,z) + D\frac{\partial}{\partial c}\frac{\partial^2 c}{\partial z^2} + S^T\frac{\partial r}{\partial c}. &\text{(E.3b)}\end{aligned}$$

Jacobian of f wrt. c

$$\frac{\partial f}{\partial c} = \frac{\partial}{\partial c}\left(-\frac{\partial N}{\partial z} + R(c)\right) \tag{E.4a}$$

$$= -\frac{\partial}{\partial c}\frac{\partial N}{\partial z} + \frac{\partial R}{\partial c}. \tag{E.4b}$$

The partial derivative of the flux term is

$$\frac{\partial}{\partial c}\frac{\partial N}{\partial z} = \frac{\partial}{\partial c}\frac{\partial}{\partial z}\left(v(t)c(t,z) - D\frac{\partial c}{\partial z}(t,z)\right) \tag{E.5a}$$

$$= v(t)\frac{\partial}{\partial c}\frac{\partial c}{\partial z}(t,z) - D\frac{\partial}{\partial c}\frac{\partial}{\partial z}\frac{\partial c}{\partial z} \tag{E.5b}$$

$$= v(t)\frac{\partial}{\partial c}\frac{\partial c}{\partial z}(t,z) - D\frac{\partial}{\partial c}\frac{\partial^2 c}{\partial z^2}. \tag{E.5c}$$

The partial derivative of the production term is

$$\frac{\partial R}{\partial c} = S^T\frac{\partial r}{\partial c}. \tag{E.6a}$$

Thus

$$\frac{\partial f}{\partial c} = -v(t)\frac{\partial}{\partial c}\frac{\partial c}{\partial z}(t,z) + D\frac{\partial}{\partial c}\frac{\partial^2 c}{\partial z^2} + S^T\frac{\partial r}{\partial c}. \tag{E.7}$$

E.1.2 Jacobian of f wrt. u - $\frac{\partial f}{\partial u}$

We describe the Jacobian of the function (E.1) wrt. u, as

$$\frac{\partial f}{\partial u} = \frac{\partial f}{\partial F} \tag{E.8a}$$

$$= -\frac{1}{A}\frac{\partial c}{\partial z}(t,z)e^T. \tag{E.8b}$$

Jacobian of f wrt. c

$$\frac{\partial f}{\partial F} = \frac{\partial}{\partial F}\left(-\frac{\partial N}{\partial z} + R(c)\right) \tag{E.9a}$$

$$= -\frac{\partial}{\partial F}\frac{\partial N}{\partial z} + \frac{\partial R}{\partial F}(c). \tag{E.9b}$$

The partial derivative of the flux term is

$$\frac{\partial}{\partial F}\frac{\partial N}{\partial z} = \frac{\partial}{\partial F}\frac{\partial}{\partial z}\left(v(t)c(t,z) + D\frac{\partial c}{\partial z}(t,z)\right) \tag{E.10a}$$

$$= \frac{\partial}{\partial F}\left(v(t)\frac{\partial c}{\partial z}(t,z) + D\frac{\partial^2 c}{\partial z^2}\right) \tag{E.10b}$$

$$= \frac{\partial}{\partial F}\frac{F_t(t)}{A}\frac{\partial c}{\partial z}(t,z) \tag{E.10c}$$

$$= \frac{\partial}{\partial F}\frac{1}{A}\frac{\partial c}{\partial z}(t,z)e^T F(t) \tag{E.10d}$$

$$= \frac{1}{A}\frac{\partial c}{\partial z}(t,z)e^T\frac{\partial F}{\partial F} \tag{E.10e}$$

$$= \frac{1}{A}\frac{\partial c}{\partial z}(t,z)e^T. \tag{E.10f}$$

The partial derivative of the production term

$$\frac{\partial R}{\partial F}(c) = 0. \tag{E.11a}$$

Thus, the partial derivative of the right-hand side wrt. inlet flows, F, is

$$\frac{\partial f}{\partial F} = -\frac{1}{A}\frac{\partial c}{\partial z}(t,z)e^T. \tag{E.12}$$

E.2 Derivatives - ODE Model

In this section, we present Jacobians for the PFR model in (6.24). We consider the right-hand side function

$$f(t,x,u,\theta) = -D^{(z)}_{N_z\times N_z+1,n_c}\bar{N} + \bar{R}, \tag{E.13}$$

with initial conditions $\bar{c}(t_0) = c_{i,0}$. The functions and variables in the finite-volume discretisation are

$$f = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N_z} \end{bmatrix}, \qquad \bar{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{N_z} \end{bmatrix}, \qquad \bar{N} = \begin{bmatrix} N_{1/2} \\ N_{1+1/2} \\ \vdots \\ N_{N_z+1/2} \end{bmatrix}, \qquad \bar{R} = \begin{bmatrix} R(c_1) \\ R(c_2) \\ \vdots \\ R(c_{N_z}) \end{bmatrix}. \tag{E.14}$$

The fluxes are computed as

$$\bar{N} = \begin{bmatrix} N_{1/2} \\ \bar{N}_{1:N_z-1} \\ N_{N_z+1/2} \end{bmatrix} = \begin{bmatrix} vc_{In} \\ v\bar{c}_{1:N_z-1} - DD^{(z)}_{N_z-1\times N_z,n_c}\bar{c} \\ vc_{N_z} \end{bmatrix}. \tag{E.15}$$

The matrix $D^{(z)}_{M\times M+1,n} \in \mathbb{R}^{nM\times n(M+1)}$ is the finite-difference operator. We define the forward difference approximation, equivalent to a central difference in the finite-volume due to the half step-size shift in fluxes and volume centres, as

$$D^{(z)}_{M\times M+1,n} = D^{(z)}_{M\times M+1} \otimes I_n, \qquad D^{(z)}_{M\times M+1} = \begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{bmatrix}, \tag{E.16}$$

where I_n is an identity matrix of size $n \times n$ and $\otimes$ is the Kronecker product operator.

E.2.1 Jacobian of f wrt. x - $\frac{\partial f}{\partial x}$

The Jacobian of the state equation wrt. the state, i.e. the concentrations, is

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial \bar{c}} \tag{E.17a}$$

$$= \frac{\partial}{\partial \bar{c}}\left(-D^{(z)}_{N_z\times N_z+1,n_c}\bar{N} + \bar{R}\right) \tag{E.17b}$$

$$= -D^{(z)}_{N_z\times N_z+1,n_c}\frac{\partial \bar{N}}{\partial \bar{c}} + \frac{\partial \bar{R}}{\partial \bar{c}}. \tag{E.17c}$$

The Jacobian of the fluxes are computed as

$$\frac{\partial \bar{N}}{\partial \bar{c}} = \begin{bmatrix} \frac{\partial N_{1/2}}{\partial \bar{c}} \\ \frac{\partial \bar{N}_{1:N_z-1}}{\partial \bar{c}} \\ \frac{\partial N_{N_z+1/2}}{\partial \bar{c}} \end{bmatrix}, \tag{E.18}$$

where

$$\frac{\partial N_{1/2}}{\partial \bar{c}} = \frac{\partial}{\partial \bar{c}}\left(v c_{In}\right) \tag{E.19a}$$

$$= 0, \tag{E.19b}$$

$$\frac{\partial \bar{N}_{1:N_z-1}}{\partial \bar{c}} = \frac{\partial}{\partial \bar{c}}\left(v\bar{c}_{1:N_z-1} - DD^{(z)}_{N_z-1\times N_z,n_c}\bar{c}\right) \tag{E.20a}$$

$$= v\frac{\partial \bar{c}_{1:N_z-1}}{\partial \bar{c}} - DD^{(z)}_{N_z-1\times N_z,n_c}\frac{\partial \bar{c}}{\partial \bar{c}} \tag{E.20b}$$

$$= v\begin{bmatrix} I_{n_cN_z-1} & 0 \end{bmatrix} - DD^{(z)}_{N_z-1\times N_z,n_c}, \tag{E.20c}$$

$$\frac{\partial N_{N_z+1/2}}{\partial \bar{c}} = \frac{\partial}{\partial \bar{c}}\left(v c_{N_z}\right) \tag{E.21a}$$

$$= v\frac{\partial c_{N_z}}{\partial \bar{c}} \tag{E.21b}$$

$$= v\begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \otimes I_{n_c}. \tag{E.21c}$$

The Jacobians of the productions are computed as

$$\frac{\partial \bar{R}}{\partial \bar{c}} = \begin{bmatrix} \frac{\partial R}{\partial c}(c_1) & & & \\ & \frac{\partial R}{\partial c}(c_2) & & \\ & & \ddots & \\ & & & \frac{\partial R}{\partial c}(c_{N_z}) \end{bmatrix}. \tag{E.22}$$

E.2.2 Jacobian of f wrt. u - $\frac{\partial f}{\partial u}$

The Jacobian of the state equation wrt. the inputs, i.e. the inlet flows, is

$$\begin{aligned} \frac{\partial f}{\partial u} &= \frac{\partial f}{\partial F} && \text{(E.23a)} \\ &= \frac{\partial}{\partial F}\left(-D^{(z)}_{N_z \times N_z+1, n_c}\bar{N} + \bar{R}\right) && \text{(E.23b)} \\ &= -D^{(z)}_{N_z \times N_z+1, n_c}\frac{\partial \bar{N}}{\partial F} + \frac{\partial \bar{R}}{\partial F} && \text{(E.23c)} \\ &= -D^{(z)}_{N_z \times N_z+1, n_c}\frac{\partial \bar{N}}{\partial F}. && \text{(E.23d)} \end{aligned}$$

The Jacobian of the fluxes are computed as

$$\frac{\partial \bar{N}}{\partial F} = \begin{bmatrix} \frac{\partial N_{1/2}}{\partial F} \\ \frac{\partial \bar{N}_{1:N_z-1}}{\partial F} \\ \frac{\partial N_{N_z+1/2}}{\partial F} \end{bmatrix}, \tag{E.24}$$

where

$$\begin{aligned} \frac{\partial N_{1/2}}{\partial F} &= \frac{\partial}{\partial F}\left(v c_{\text{In}}\right) && \text{(E.25a)} \\ &= \frac{\partial c_{\text{In}}}{\partial F} v + \frac{\partial v}{\partial F} c_{\text{In}}, && \text{(E.25b)} \end{aligned}$$

where

$$\begin{aligned} \frac{\partial c_{\text{In}}}{\partial F} &= \frac{\partial}{\partial F}\left(C_{\text{In}}\frac{F}{e^T F}\right) && \text{(E.26a)} \\ &= C_{\text{In}}\frac{\partial}{\partial F}\left(\frac{F}{e^T F}\right) && \text{(E.26b)} \\ &= C_{\text{In}}\left(\frac{\partial F}{\partial F}e^T F - F e^T \frac{\partial F}{\partial F}\right)\left(I e^T F F^T e\right)^{-1} && \text{(E.26c)} \\ &= C_{\text{In}}\left(I e^T F - F e^T\right)\left(e^T F F^T e\right)^{-1}, && \text{(E.26d)} \end{aligned}$$

and

$$\begin{aligned}\frac{\partial v}{\partial F} &= \frac{\partial}{\partial F}\left(\frac{e^T F}{A}\right) && \text{(E.27a)}\\ &= \frac{1}{A}e^T\frac{\partial F}{\partial F} && \text{(E.27b)}\\ &= \frac{1}{A}e^T. && \text{(E.27c)}\end{aligned}$$

$$\begin{aligned}\frac{\partial \bar{N}_{1:N_z-1}}{\partial F} &= \frac{\partial}{\partial F}\left(v\bar{c}_{1:N_z-1} - DD^{(z)}_{N_z\times N_z+1,n_c}\bar{c}\right) && \text{(E.28a)}\\ &= \bar{c}_{1:N_z-1}\frac{\partial}{\partial F}v - \frac{\partial}{\partial F}\left(DD^{(z)}_{N_z\times N_z+1,n_c}\bar{c}\right) && \text{(E.28b)}\\ &= \bar{c}_{1:N_z-1}\frac{\partial}{\partial F}\left(\frac{1}{A}e^T F\right) && \text{(E.28c)}\\ &= \frac{1}{A}\bar{c}_{1:N_z-1}e^T\frac{\partial F}{\partial F} && \text{(E.28d)}\\ &= \frac{1}{A}\bar{c}_{1:N_z-1}e^T. && \text{(E.28e)}\end{aligned}$$

$$\begin{aligned}\frac{\partial N_{N_z+1/2}}{\partial F} &= \frac{\partial}{\partial F}\left(vc_{N_z}\right) && \text{(E.29a)}\\ &= c_{N_z}\frac{\partial}{\partial F}v && \text{(E.29b)}\\ &= c_{N_z}\frac{\partial}{\partial F}\left(\frac{1}{A}e^T F\right) && \text{(E.29c)}\\ &= \frac{1}{A}c_{N_z}e^T\frac{\partial F}{\partial F} && \text{(E.29d)}\\ &= \frac{1}{A}c_{N_z}e^T. && \text{(E.29e)}\end{aligned}$$

E.3 Implementations

E.3.1 Matlab

```
function [f, dfdx, dfdu, dfdd] = driftPFR(t, x, u, d, theta)
%-----------------------------------------------------------------
%   Authour(s):
%       Marcus Krogh Nielsen
%
%   Email:
```

```
%       mkrni@dtu.dk
%
%-----------------------------------------------------------------------
%   Call:
%       [f, dfdx, dfdu, dfdd] = driftPFR(t, x, u, d, theta)
%
%   Description:
%       Drift function for a plug flow reactor.
%
%   Inputs:
%       t       :     time
%       x       :     states
%       u       :     manipulated variables
%       d       :     disturbances
%       theta   :     parameters
%
%   Outputs:
%       f       :     state drift function
%       dfdx    :     drift derivative wrt. states
%       dfdu    :     drift derivative wrt. manipulated variables
%       dfdd    :     drift derivative wrt. disturbances
%
%-----------------------------------------------------------------------

%% Parameters

% Size
% ...
% variables
nx = theta.nx;
nu = theta.nu;
nd = theta.nd;
nr = theta.nr;
% discretisation
Nz = theta.Nz;

% Parameters
% ...
% model
CIn = theta.CIn;                        % inlet concentrations
D   = theta.D;                          % diffusion constant
A   = theta.A;                          % cross-sectional area
% discretisation
Dz  = theta.Dz;                         % discretisation matrix

% Kinetics
rFun = theta.rFun;
rPar = theta.rPars;
S    = theta.S;

% Variables
% ...
% states
c = x;
% inputs
F = u;


%% Function

% Production rates
[r,  ] = rFun(t, c, rPar);
R      = S'*reshape(r, nr, Nz);
R      = R(:);

% Flow
FL = sum(F);
v  = FL/A;

% Compute dispersion
% ...
% derivative
dcdz = Dz(1:end-1*nx,1:end-1*nx)*c;
% dispersion
J    = - kron(eye(Nz-1), diag(D))*dcdz;

% Compute fluxes and spatial derivative
% ...
% inlet and outlet concentrations
cIn  = CIn*F/FL;
cL   = c(end-nx+1:end);
% flux
N    = [v*cIn; v*c(1:end-1*nx) + J; v*cL];
% flux Jacobian wrt. z
dNdz = Dz*N;

% Compute right-hand side
f = - dNdz + R;


%% Derivative

if nargout > 1
    % kinetics derivative wrt. concentrations
    [ , drdc] = rFun(t, c, rPar);

    % drift Jacobian wrt. states (x)
    % ...
    % dNdc
    dN1dc  = zeros(nx, nx*Nz);
    dcIdc  = zeros(nx*(Nz-1), nx*Nz); dcIdc(1:end, 1:nx*(Nz-1)) = eye(nx*(
        Nz-1));
    dNIdc  = v*dcIdc - kron(eye(Nz-1), diag(D))*Dz(1:end-1*nx,1:end-1*nx);
    dNNzdc = v*kron([zeros(1, Nz - 1), 1], eye(nx));
    dNdc    = [dN1dc; dNIdc; dNNzdc];
    % dRdc
    dRdc = zeros(Nz*nx, Nz*nx);
    for i=1:Nz
        dRdc((i-1)*nx+1:i*nx, 1:end) ...
            = S'*drdc((i-1)*nr+1:i*nr, 1:end);
    end
    % dfdx
    dfdx = - Dz*dNdc + dRdc;


    % drift Jacobian wrt. inputs (u)
    % ...
    % dNdF
    enu     = ones(nu, 1);
    Inu     = eye(nu);
    dcIndF = CIn*(Inu*(enu'*F) - F*enu')/(enu'*F*F'*enu);
    dvdF    = 1/A*enu';
    dN1dF   = dcIndF*v + cIn*dvdF;
    dNIdF   = 1/A*c(1:end-nx)*enu';
    dNNzdF = 1/A*c(end-nx+1:end)*enu';
    dNdF    = [dN1dF; dNIdF; dNNzdF];
    % dfdu
    dfdu    = - Dz*dNdF;

    % drift Jacobian wrt. disturbances (d)
    % ...
    % dfdd
    dfdd = zeros(nx*Nz, nd);

end

end
```

./code/matlab/driftPFR.m.

E.3.2 Python

```
## Imports
import numpy as np


###########################################################################
##   Authour(s):
##       Marcus Krogh Nielsen
##
##   Email:
##       mkrni@dtu.dk
##
```

```python
##############################################################################
##    Call:
##        f, dfdx, dfdu, dfdd = driftPFR(t, x, u, d, theta, nargout)
##
##    Description:
##        Drift function for a plug flow reactor.
##
##    Inputs:
##        t        :    time
##        x        :    states
##        u        :    manipulated variables
##        d        :    disturbances
##        theta    :    parameters
##
##    Outputs:
##        f        :    state drift function
##        dfdx     :    drift derivative wrt. states
##        dfdu     :    drift derivative wrt. manipulated variables
##        dfdd     :    drift derivative wrt. disturbances
##
##############################################################################
def driftPFR(t, x, u, theta, nargout=1):
    ## Parameters

    # Size
    # ...
    # variables
    nx = theta.nx
    nu = theta.nu
    nd = theta.nd
    nr = theta.nr
    # discretisation
    Nz = theta.Nz

    #  Parameters
    # ...
    # model
    CIn = theta.CIn
    D   = theta.D
    A   = theta.A
    # discretisation
    Dz  = theta.Dz

    # Kinetics
    rFun = theta.rFun
    rPar = theta.rPar
    S    = theta.S

    # Variables
    # ...
    # states
    c = x
    # inputs
    F = u


    ## Function

    # Production rates
    r, _ = rFun(t, c, rPar)
    R    = S.T@np.reshape(r, (nr, Nz))
    R    = np.reshape(R, (nr*Nz, 1))

    # Flow
    e  = np.ones(nu, 1)
    FL = e.T@F
    v  = FL/A

    # Compute dispersion
    # ...
    # derivative
    dcdz = Dz[:-nx, :-nx]@c
    # dispersion
    J    = np.kron(np.eye(Nz-1), np.diag(D))@dcdz

    # Compute fluxes and spatial derivative
    # ...
    # inlet and outlet concentrations
    cIn  = CIn@(F/FL)
    cL   = c[-nx:]
    # flux
    N    = np.array([[v*cIn], [v*c[:-nx] + J], [v*cL]])
    # flux spatial derivative
    dNdz = Dz@N

    # Evaluate drift term
    f = - dNdz + R


    ## Derivatives
    if nargout > 1:
        # Kinetics drivative wrt. concentrations
        _, drdc = rFun(t, x, rPar)

        # Drift Jacobian wrt. states (x)
        # ...
        # dNdc
    dN1dc  = np.zeros(nx, nx*Nz)
    dcIdc  = np.zeros(nx*(Nz-1), nx*Nz); dcIdc[:,:-nx] = np.eye(nx*(Nz-1))
    dNIdc  = v*dcIdc - np.kron(np.eye(Nz-1), np.diag(D))*Dz[:-nx, :-nx]
    aux    = np.zeros(1, Nz); aux[-1] = 1
    dNNzdc = v*np.kron(np.array(aux, np.eye(nx))
    dNdc   = np.array([[dN1dc], [dNIdc], [dNNzdc]])
    # dRdc
    dRdc    = np.zeros(nx*Nz, nx*Nz)
    for i in range(Nz):
         dRdc[i*nx:(i+1)*nx] = S.T@drdc[i*nr:(i+1)*nr, :]
    # dfdx
    dfdx = - Dz@dNdc + dRdc

        # Drift Jacobian wrt. inputs (u)
        # ...
        # dNdF
    enu = np.ones(nu, 1)
    Inu = np.eye(nu)
    dcIndF  = CIn@(Inu*(enu.T@F) - F@enu.T)/(enu.T@F@F.T@enu)
    dvdF    = 1/A*enu.T
    dN1dF   = dcIndF*v + cIn@dvdF
    dNIdF   = 1/A*c[:-nx]@enu.T
    dNNzdF  = 1/A*c[-nx:]@enu.T
    dNdF    = np.array([[dN1dF], [dNIdF], [dNNzdF]])
        # dfdu
    dfu     = - Dz@dNdF

        # Drift Jacobian wrt. disturbances (d)
        # ...
        # dfdd
    dfdd = np.zeros(nx*Nz, nd)

        return f, dfdx, dfdu, dfdd

    # Return statement
    return f
```

./code/python/driftPFR.py.

Bibliography

[1] Peter Victor Danckwerts. “Continuous flow systems: distribution of residence times.” In: *Chemical engineering science* 2.1 (1953), pp. 1–13. DOI: 10.1016/0009-2509(53)80001-1.

[2] André Drejer et al. “Economic optimizing control for single-cell protein production in a U-loop reactor.” In: *Proceedings of the 27th European Symposium on Computer Aided Process Engineering (ESCAPE)*. Barcelona, Spain. October 1-5, 2017, pp. 1759–1764. DOI: 10.1016/B978-0-444-63965-3.50295-6.

[3] H. Scott Folger. *Elements of chemical engineering*. 4th ed. Pearson education limited, 2014. ISBN: 9781292026169.

[4] A. Johnson. “The control of fed-batch fermentation processes - a survey.” In: *Automatica* 23.6 (1987), pp. 691–705. DOI: 10.1016/0005-1098(87)90026-4.

[5] John Bagterp Jørgensen et al. “Simulation of NMPC for a laboratory adiabatic CSTR with an exothermic reaction.” In: *proceedings of the 2020 European Control Conference (ECC)*. Saint Petersburg, Russia. May 12-15, 2020, pp. 202–207. DOI: 10.23919/ECC51009.2020.9143733.

[6] Morten Wahlgreen Kaysfeld et al. “Modeling and simulation of upstream and downstream processes for monoclonal antibody production.” In: *IFAC-PapersOnLine* 55.7 (2022), pp. 685–690. DOI: 10.1016/j.ifacol.2022.07.523.

[7] Mario Novak and Predrag Horvat. “Mathematical modelling and optimisation of a waste water treatment plant by combined oxygen electrode and biological waste water treatment model.” In: *Applied Mathematical Modelling* 36.8 (2012), pp. 3813–3825. DOI: https://doi.org/10.1016/j.apm.2011.11.028.

[8] Dres Foged Olsen et al. “Optimal operating points for SCP production in the U-loop reactor.” In: *proceedings of the 9th International Symposium on Dynamics and Control of Process Systems (DYCOPS 2010)*. Leuven, Belgium. July, 5-7, 2010, pp. 485–490. DOI: 10.3182/20100705-3-BE-2011.00083.

[9] Thomas Emil Ryde et al. “Optimal feed trajectories for fedbatch fermentation with substrate inhibition kinetics.” In: *proceedings of the International Symposium on Advanced Control of Chemical Processes (ADCHEM)*. Venice, Italy. June 13-16, 2021, pp. 318–323. DOI: 10.1016/j.ifacol.2021.08.261.

[10] R. Sander. “Compilation of Henry’s law constants (version 4.0) for water as solvent.” In: *Atmospheric Chemistry and Physics* 15.8 (2015), pp. 4399–4981. DOI: 10.5194/acp-15-4399-2015.

[11] Rolf. Sander. *Henry's Law Constants*. Jan. 2023. URL: `http://www.henrys-law.org/henry/`.

[12] Eskild Schroll-Fleischer et al. *Adiabatic Continuous Stirred Tank Reactor*. Tech. rep. Technical University of Denmark, 2017.

[13] Robin Smith. *Chemical process design and intergration*. John Wiley & Sons, Ltd, 2005. ISBN: 0471486817.

[14] John Villadsen, Jens Nielsen, and Gunnar Lidén. *Bioreaction engineering principles*. Springer Science+Business Media, 2011. ISBN: 9781441996879.

[15] Morten Ryberg Wahlgreen et al. "Nonlinear model predictive control for an exothermic reaction in an adiabatic CSTR." In: *proceedings of the 6th Conference on Advances in Control and Optimization of Dynamical Systems (ACODS)*. Chennai, India. Febuary 16-19, 2020, pp. 500–505. DOI: `10.1016/j.ifacol.2020.06.084`.