LUND UNIVERSITY

Master Thesis

Orbit-simulator for downstream processes

by

Andreas Malmström

Department of Chemical Engineering
Lund University
Sweden

March 25, 2022

Supervisor: **Doctor Niklas Andersson**
Examiner: **Professor Bernt Nilsson**

Postal address
PO-Box 124
SE-221 00 Lund, Sweden
Web address
www.lth.se/chemeng

Visiting address
Naturvetarvägen 14

Telephone
+46 46-222 82 85
+46 46-222 00 00

Printed in Sweden by Media-Tryck.
Lund 2022

Preface

The work described in this master thesis was performed at the department of chemical engineering at Lunds Tekniska Högskola, Lund University. Working in a pandemic era, the work was to a large extent conducted at the home office.

Acknowledgement

I would like to extend my immense gratitude to doctor Niklas Andersson for his valuable guidance and assistance, concerning the control system Orbit and the master thesis process as a whole.

Furthermore, I would like to thank professor Bernt Nilsson for presenting this master thesis project to me and for his enthusiasm in teaching at the department of chemical engineering.

I would also like to thank Barncancerfonden for the valuable support they have provided my brother, his partner and their daughter in her fight against leukemia, and for the support they have provided me and my parents during this difficult time.

Further, this work would have been a lot more challenging without the motivation I have received from my girlfriend and I am forever grateful for her encouraging words along the work on this master thesis.

Finally, I would like to quote one of our times great poets:

> *”Last but not least: I want to thank me, I want to thank me for believing in me, I want to thank me for doing all this hard work. I want to thank me for having no days off. I want to thank me for never quitting, ... , I want to thank me for just being me at all times.”*

-Snoop Dogg (Broadus 2018)

Thank You!

Abstract

In an ever more digitilized society, the transition is palpable also in the pharmaceutical industry. The control system Orbit was developed at the department of chemical engineering at Lund University, to be able to automate downstream processes based on the ÄKTA-system. The control system has a built-in simulator, enabling the user to run simulated experiments instead of actual physical experiments. The aim of this master thesis was to extend the amount of models within the simulator, allowing for more chromatography types and membranes, but also processes with multiple unit operations to be simulated.

Models for IEX, SEC, HIC, and AC were implemented into the simulator. Also, a LRMFT modeling approach was added and compared to the already implemented LRM approach. An attempt to implement a UFDF model was also made, but with no success. Further on, a two column process was simulated. Lastly, a calibration method was developed and a parameter estimation for a single column, and a multiple column process was conducted.

The results suggest it to be possible to simulate the added models. The LRMFT modeling approach was, however, found to overestimate the adsorption, and no reason to why was found within the time frame of this master thesis. Further on, it was not possible to simulate membranes, due to every connected stream being interpreted as an outgoing stream by the simulator. The results from the two column process suggest it to be possible to simulate multiple columns within the same process. The parameter estimation indicated it to be possible to estimate parameters in single and multiple columns within the same process, however the robustness of the calibration method should be increased. For any future work, the models should be validated against experimental data.

Sammanfattning

I ett allt mer digitaliserat samhälle, så är omställningen påtaglig även inom läkemedelsindustrin. Kontrollsystemet Orbit utvecklades vid Lunds Universitets avdelning för kemiteknik, för att möjliggöra automatisering av nedströmsprocesser baserade på ÄKTA-system. Kontrollsystemet har en inbyggd simulator, som tillåter en användare att genomföra simulerade experiment istället för fysiska. Målet med detta examensarbete var att utöka mängden modeller i simulatorn, för att möjligöra simulering av fler kromotografityper och membran, men även av processer med flera olika enhetsoperationer.

Modeller för IEX, SEC, HIC samt AC blev inarbetade i simulatorn. Vidare så lades ett LRMFT-modelleringstillvägagångssätt till, och jämfördes med det redan inarbetade LRM-tillvägagångssättet. En ansats att inarbeta en modell för UFDF gjordes också, men utan framgång. Dessutom, simulerades en process med två kolonner. Slutligen utvecklades en kalibreringsmetod och en parameterskattning utfördes i processer med en respektive flera kolonner.

Resultaten antyder att det är möjligt att simulera de tillagda modellerna. LRMFT-modelleringstillvägagångssättet visade sig överskatta adsorptionen och varför så var fallet kunde inte fastställas inom tidsramen för detta examensarbete. Vidare var det inte möjligt att simulera membran, på grund av att simulatorn tolkade varje ström kopplad till membranet som en utgående ström. Resultaten från processen med två kolonner tyder på att det är möjligt att simulera flera kolonner i samma process. Parameterskattningen indikerar på att det är möjligt att skatta parametrar i en samt flera kolonner i samma process, dock bör robustheten hos kalibreringsmetoden utökas. För framtida arbete bör experimentell data tas fram, så att modellerna kan valideras.

Popular science summary

Like how the artist captures the reality in her paintings, does the engineer capture the reality in his models. Not with paint, but with mathematics. In this work, the motive was the separation processes of the pharmaceutical industry and the canvas the coding language Python. In an ever more digitilized society, the digital transition is palpable also in the pharmaceutical industry. Processes are being automatized and simulated, in an attempt to increase production figures and competitiveness.

One program for such automation is Orbit, developed at the department of chemical engineering at Lunds University. Orbit is used to automatize so called down stream processes, processes used for purifying protein solutions in the production of pharmaceuticals. Orbit also has an in-built simulator, allowing the user to either conduct a physical or virtual experiment. The aim of this master thesis was to expand the amount of models in the simulator, extending its usefulness.

To the simulator, models for chromatography columns were added. Chromatography columns can be described as tubes, filled with a packing material, often in the shape of porous beads. The proteins being led through the column as a solution, will be separated due to physio-chemical interactions with the packing material. Four models for four different types of chromatography were added to the simulator. One model for ion exchange chromatography, a chromatography type where protein sticks to the packing material, due to charge, and is released by gradually increasing the salt concentration. One model for size exclusion chromatography, a chromatography type where the pores of the packing material causes different sized proteins to have differing retention times. One model for hydrophobic interaction chromatography, a chromatography type where high salt concentrations in the protein solution will cause the protein to stick, due to hydrophobic interaction, to the packing material to then be released by lowering the salt concentration. Lastly, one model for pH-dependent affinity chromatography, a chromatography model where the protein sticks to the packing material then to be released by lowering the pH. Further, a new modeling approach was also added to the simulator. Attempts were also made to implement a model for membranes, but with no success.

Some of these models, the ion exchange chromatography model and the size exclusion

chromatography model, were then used to simulate a process with two columns. Also, a calibration method was developed and a parameter estimation for one and multiple columns were made, to be able to estimate the parameters found in the models.

The results suggest it possible to simulate the different models added to the simulator, but the lack of experimental data meant that no validation of the models could be made. The new modeling approach was found to overestimate the amount of protein sticking to the packing material, and no reason to why could be found within the time frame of this master thesis. The simulation of the two column process indicate that it is possible to simulate multiple types of chromatography within the same process. The parameter estimation showed, that the developed calibration method in some cases was able to estimate the parameters not only in one but also multiple columns.

Conclusions made from this work, is that it is possible to capture the behavior of different types of chromatography columns within the Orbit-simulator, but experimental data is needed for validation of the models. Also, it is possible to simulate multiple columns of different types within the same process. Further on, the simple calibration method is, under some circumstances, capable of estimating the parameters, which could be useful for some applications, but for any future work it is recommended that the robustness of the method is increased.

Populärvetenskaplig sammanfattning

Likt konstnären avbildar ingenjören verkligheten, men inte med målarfärg utan med matematik. I detta arbete var motivet läkemedelsindustrins seperationsprocesser och målarduken kodprogrammet Python. I ett allt mer digitaliserat samhälle gör sig den digitala utvecklingen påtaglig även inom läkemedelsindustrin. Processer blir automatiserade och simulerade för att öka produktionssiffror och konkurrenskraft.

Ett program för sådan automatisering är Orbit, utvecklat vid Lunds Universitets avdelning för kemiteknik. Orbit används för att automatisera så kallade nedströmsprocesser, processer som renar lösningar med protein för att framställa läkemedel. I Orbit finns även en inbyggd simulator som låter användaren välja mellan att utföra ett fysiskt experiment eller ett simulerat. Detta examensarbetes mål var att utöka mängden matematiska modeller i simulatorn.

Till simulatorn lades därför modeller för kromotografikolonner till. Kromotografikolonner kan bäst liknas vid tuber, fyllda med ett packningsmaterial, ofta i form av små porösa kulor. När proteinlösningen leds genom tuben kommer fysiska och/eller kemiska interaktioner med packningsmaterialet leda till seperation av proteinerna. Fyra modeller för fyra olika typer av kromotografityper inarbetades i simulatorn. En modell för jonbyteskromotografi, dvs kromotografi där protein genom laddning fastnar på packningsmaterialet, för att sedan lösgöras genom att gradvis höja saltkoncentrationen. En modell för gelfiltrering, dvs kromotografi där packningsmaterialets porer gör att protein med olika storlek får olika uppehållstid i kolonnen. En modell för hydrofob interaktionskromotografi, dvs kromotografi där höga salthalter i lösningen gör att proteinet fastnar på packningsmaterialet genom hydrofobisk interaktion, för att sedan lösgöras när salthalten sänks. Till sist, en modell för pH-beroende affinitetskromotografi, dvs kromotografi där proteinet fastnar på packningsmaterialet för att sedan lösgöras genom att sänka pH. Ett nytt modelleringstillvägagångssätt lades också till i simulatorn. Försök gjordes även att inarbeta en modell för membran, som bäst kan liknas vid ett filter, men utan framgång.

Med några av dessa modeller, simulerades en process med två olika kolonner, en jonbytes- och en gelfiltreringskolonn. Dessutom utvecklades en kalibreringsmetod och därefter genomfördes en parameterskattning för en samt flera kolonner, för att kunna uppskatta de parametrar som återfinns i modellerna.

Resultaten visar på att det är möjligt att simulera de olika kromotografityper som lades till i simulatorn, men avsaknaden av experimentell data gjorde att modellerna inte kunde valideras. Det nya modelleringstillvägagångssättet visade sig överskatta mängden protein som fastnade på packningsmaterialet och ingen anledning till varför kunde fastställas inom tidsramen för detta arbete. Simuleringen av en process med två kolonner lyckades vilket tyder på att det är möjligt att simulera flera olika typer av kolonner i samma process. Parameterskattningen visade att den utvecklade kalibreringsmetoden i vissa fall lyckas skatta parametrarna i en men även flera kolonner.

Slutsatser man kan dra från detta arbete är att det är möjligt att avbilda flera olika typer av kromotografi i Orbit-simulatorn, men att experimentell data behövs för validering av modellerna. Dessutom är det möjligt att simulera flera olika typer av kromotografi i samma process. Fortsättningsvis, den simpla kalibreringsmetoden är kapabel till att skatta parametrar under vissa förutsättningar, vilket lämpar sig för vissa tillämpningar, men för framtida arbete bör metodens robusthet utökas.

Contents

1 Introduction

In an ever more digitilized society, the transition is palpable also in the pharmaceutical industry. Automation of continuous downstream processes, i.e. processes used for purification of pharmaceuticals, is being implemented to reach higher production figures and to facilitate competitiveness in an increasingly competitive market (Izmaylov, Saraev, and Barinova 2021; Klutz et al. 2015; Leweke and Lieres 2018). In this development to a more automated industry, simulation of downstream processes has gained interest throughout the years, due to it being an efficient tool for evaluating and developing process designs. The benefits are clear; decreased resource consumption, lower operating costs, and increased process optimization possibilities. (Schmidt-Traub, Schulte, and Seidel-Morgenstern 2020; Gomis-Fons et al. 2019)

One such system for automating and simulating downstream processes is the control system Orbit. The control system was developed at the department of chemical engineering at Lund University, to be able to automate downstream processes based on the ÄKTA-system. The control system has a built-in simulator, enabling the user to run simulated experiments instead of actual physical experiments. This is a powerful tool and can be used for process design improvements, aiding understanding of the process, and pose as a cost and resource effective alternative to physical experiments. (Andersson 2018) However, the current simulator is in need of more models, to extend its usefulness.

1.1 Aim

The aim of this work is to extend the amount of models in the Orbit simulator. The following unit operations are considered:

- Ion exchange chromatography
- Size exclusion chromatography

- Hydrophobic interaction chromatography
- pH-dependent affinity chromatography
- Ultrafiltration and dialfiltration

With models for these unit operations a process with multiple unit operations will be constructed to test the simulators capability of simulating processes containing multiple unit operations.

To help reach the aim the following research questions are asked:

- How does an increased model complexity, due to the introduction of a film transfer, affect the simulations of chromatography columns?
- Is it possible to add additional chromatography models to the simulator, thus enabling for multiple chromatography types to be modeled within the same process?
- Can size exclusion chromatography be simulated with the Orbit-simulator?
- Is it possible to simulate membranes with the Orbit-simulator?
- How can the models be calibrated against data?
- Is it possible to estimate parameters in multiple chromatography columns, within the same process?

2 Background

This section will present the background knowledge which this master thesis project builds on. It gives a description of different types of chromatography columns, membranes, modeling approaches for chromatography columns and membranes, the ÄKTA Pure system, previously developed chromatography simulators, and lastly a description of the control system Orbit and its in-built simulator.

2.1 Liquid chromatography

Liquid chromatography is a separation process often used in the pharmaceutical industry as a preperative method for purifying compounds in complex mixtures (Schmidt-Traub, Schulte, and Seidel-Morgenstern 2020). Several different types of liquid chromatography exists as of today, but, the basic principle, however, is very similar for the different types. This section gives a brief presentation of the basic working principle of liquid chromatography and thereafter more detailed descriptions of some different types of liquid chromatography.

2.1.1 The basics of chromatography

The chromatography column consists of a tube packed with a porous material, often in the shape of beads, see Figure 2.1. This packing material is called the stationary phase. The sample is led through the column, in liquid form, and physio-chemical interactions between the components in the liquid sample, called the mobile phase, and the stationary phase, will lead to differing retention times for the components. This will allow for pooling of the target, leading to separation. (Linde 2021)

2.1.2 Size exclusion chromatography (SEC)

In size exclusion chromatography the proteins are, as the name suggests, separated due to size. The pores in these chromatography columns are designed in such a way that large molecules will be unable to pass some or all of the pores, resulting in a

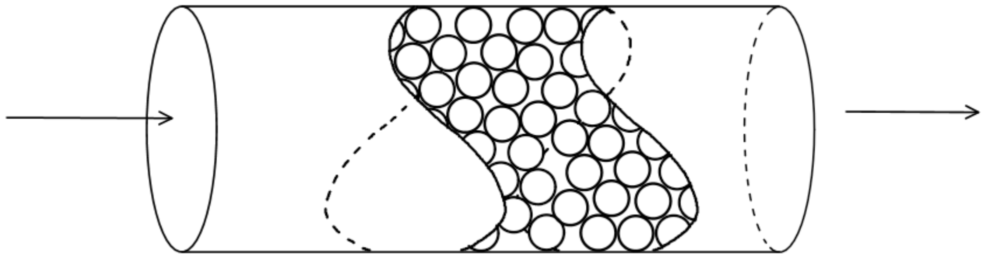


Figure 2.1 – A schematic drawing of the working principle of a chromatography column.

shorter retention time compared to smaller molecules. (Knox and Scott 1984)

2.1.3 Ion-exchange chromatography (IEX)

Ion-exchange chromatography utilizes ion-exchangers which interact with the charged moieties, present at the surface of some proteins and other biomolecules, to separate the desired compound. The unit consists of a column loaded with an ion-exchanger, often in the form of porous beads, in a so called base matrix. This voided matrix provides a large surface area, promoting adsorption of the charged moieties. (Jungbauer and R. Hahn 2009)

The ion-exchange chromatography separation process consists of a row of different phases. First there is a loading phase, where the column is loaded with a salt-solution, a so called buffer, to saturate the ligands with counter ions. Then there is an equilibration phase, where a buffer is loaded to the column to stabilize the pH to a desired level. Following this phase the protein solution is loaded and compounds attach to the base matrix. The unbound material is then washed out with equilibration buffer during a washing phase. When all unbound material has been washed out, a mixture of equilibration buffer (Buffer A) and the buffer used for saturation of counter ions (Buffer B) is loaded. The amount of Buffer B in the mixture is gradually increased, often in a linear manner, during the loading. This causes the salt concentration to gradually increase and different proteins to detach and to be washed out at different concentrations. This phase is called the gradient phase. After the gradient phase the column is regenerated by, once again, saturating it with counter ions. (Jungbauer and R. Hahn 2009)

2.1.4 Hydrophobic interaction chromatography (HIC)

Hydrophobic interaction chromatography utilizes hydrophobic ligands, attached to the stationary phase, to separate the proteins through hydrophobic interaction with non-polar groups on the surface of the protein. High concentrations of salt in the mobile phase will increase adsorption, and by lowering the salt concentration, elution is achieved. Different salts can either promote or impede hydrophobic interaction between the protein and the stationary phase, depending on the salt-ions interaction behavior with water molecules. Salt-ions resulting in a structuring behavior of the water molecules promote hydrophobic interaction, where as salt-ions resulting in a randomized structuring of the water molecules impede hydrophobic interaction. (Queiroz, Tomaz, and Cabral 2001)

2.1.5 Affinity chromatography (AC)

Affinity chromatography utilizes affinity ligands for the separation of proteins or other biomolecules, resulting in separation with high specificity. In this type of chromatography, a so called application buffer is used to promote binding of the target protein to the ligands. (Hage and Matsuda 2015)

The target protein is then eluted either through non-specific elution or biospecific elution. During non-specific elution, if the binding of the target protein is weak, the column can once again be flushed with application buffer, causing the target protein to elute. But more often, an approach similar to the gradient phase in ion-exchange chromatography is applied during non-specific elution in an affinity chromatography column. Here a second mobile phase, an elution buffer, is used in gradually increasing concentrations to cause the target protein to be removed from the column. Sometimes the elution is caused by a change in pH (Nilsson and Andersson 2017). If a competing agent is present in the elution buffer, it makes for biospecific elution. The target protein is released due to dissociation or mass action. (Hage and Matsuda 2015)

2.2 Modeling approaches for chromatography columns

Modeling of chromatography columns have been conducted for a long time and different approaches of varying complexity have been presented, see Table 2.1, (Schmidt-Traub, Schulte, and Seidel-Morgenstern 2020). The physio-chemical interactions in the different types of chromatography columns are many times captured by an adsorption term in these models, often expressed through an internal film mass transfer, an external film mass transfer, or through kinetics between adsorption and desorption. In each of these different ways of expressing the adsorption term, an adsorption description can be defined, be it linear, Langmuir, LeVan-Vermeulen isotherms or any other description (Bellot and Condoret 1993), to be able to capture the physio-chemical interactions in a specific type of chromatography column. (Gu 2015; Nilsson and Andersson 2017)

Table 2.1 – List of different modeling approaches for chromatography columns. Adapted from Schmidt-Traub et.al. (Schmidt-Traub, Schulte, and Seidel-Morgenstern 2020).

Model approach	**Considered physio-chemical effects**
Ideal model	• Convection • Adsorption equilibrium
Thomas Model	• Convection • Adsorption kinetics

Dispersive Model	• Convection • Dispersion • Adsorption equilibrium
Transport Model	• Convection • Adsorption equilibrium • Mass transfer resistance
Reactive-dispersive model	• Convection • Dispersion • Adsorption kinetic
Transport-dispersive model	• Convection • Dispersion • Adsorption equilibirum • Mass transfer resistance

General rate model	• Convection • Dispersion • Adsorption • Mass transfer resistance • Adsorption equilibrium or kinetic

In this work only transport-dispersive and reaction-dispersive models, so called lumped rate models (LRM), will be considered. These models provide high enough accuracy for most chromatography types considered in this work. (Schmidt-Traub, Schulte, and Seidel-Morgenstern 2020) Also, the already implemented modeling approach is of the type LRM, and the simulator is designed according to this. An implementation of a general rate modeling approach would perhaps provide greater accuracy for some chromatography types but would result in changes to the simulator, which were deemed to lie outside the limits of this work.

2.2.1 Reaction-Dispersive models

The reaction-dispersive models consider the convective and dispersive transport in the interstitial volume of the column but an average concentration, same as the bulk concentration, is assumed for the intraparticulate volume and intraparticulate diffusion is neglected. (GoSilico 2021; Schmidt-Traub, Schulte, and Seidel-Morgenstern 2020)

Further on, the reaction-dispersive models also build on the following assumptions, (Schmidt-Traub, Schulte, and Seidel-Morgenstern 2020):

1. No concentration gradient in the radial direction of the column
2. Isothermal conditions

3. Incompressible liquid
4. Homogenous adsorbent bed
5. The eluent is inert
6. Adsorption kinetics are rate limiting

Equation 2.1 describes the material balance in the mobile phase in the column and Equation 2.2 the material balance in the stationary phase, in a reaction-dispersive model for a general column type. (Nilsson and Andersson 2017) This is the current modeling approach within the simulator.

$$\frac{\partial c_i}{\partial t} = D_{ax,i}\frac{\partial^2 c_i}{\partial z^2} - \frac{v_{sup}}{\epsilon_a}\frac{\partial c_i}{\partial z} - \frac{1-\epsilon_c}{\epsilon_a} r_{ads,i} \tag{2.1}$$

$$\frac{\partial q_i}{\partial t} = r_{ads,i} \tag{2.2}$$

ϵ_a is calculated according to Equation 2.3, where ϵ_p is the particle porosity. (Nilsson and Andersson 2017)

$$\epsilon_a = \epsilon_c + (1-\epsilon_c)\epsilon_p \tag{2.3}$$

2.2.2 Transport-Dispersive models

The transport-dispersive models introduce a liquid film linear driving force, combining the internal and external mass transfer resistances into one mass transfer coefficient, k_{eff}. This allows for concentrations inside the particles to be calculated. Transport-dispersive models build on mostly the same assumptions as the reaction-dispersive models but with the addition of the lumped film transfer coefficient being independent of axial dispersion. It also assumes spherical particles and no assumption of the adsorption kinetics being rate limiting is made. The material balance of the bulk phase is now described by Equation 2.4 and the material balance of the particle phase by Equation 2.5. The material balance of the stationary phase is described in the same way as in the reaction-dispersive model, see Equation 2.2, and will likewise vary between different column types. (Schmidt-Traub, Schulte, and Seidel-Morgenstern 2020)

$$\frac{\partial c_i}{\partial t} = D_{ax,i}\frac{\partial^2 c_i}{\partial z^2} - \frac{v_{sup}}{\epsilon_c}\frac{\partial c_i}{\partial z} - \frac{1-\epsilon_c}{\epsilon_c}(\epsilon_p\frac{\partial c_{p,i}}{\partial t} + (1-\epsilon_p)\frac{\partial q_i}{\partial t}) \tag{2.4}$$

$$\epsilon_p\frac{\partial c_{p,i}}{\partial t} + (1-\epsilon_p)\frac{\partial q_i}{\partial t} = k_{eff,i}\frac{3}{r_p}(c_i - c_{p,i}) \tag{2.5}$$

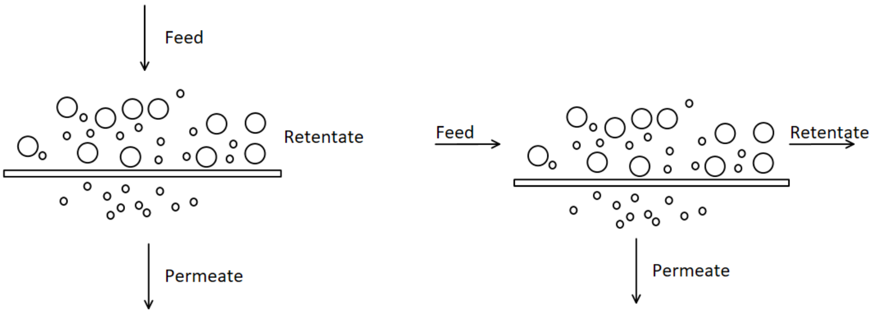
Transport-dispersive models are often used in applications where mass transport is considered to be rate limiting (Schmidt-Traub, Schulte, and Seidel-Morgenstern 2020). In this work no such limitation is made, however. If the film transfer is set to not be rate limiting, the results of the transport-dispersive models should mimic the results of the reaction-dispersive models, since the intraparticulate concentration will approach the value of the bulk concentration, but with the important difference of a film still being modeled. Because of this, the terms LRM (Lumped Rate Model) and LRMFT (Lumped Rate Model with Film Transfer) are used in this work, where LRM indicates models not modeling a film and LRMFT indicates models modeling a film.

2.3 Membranes

This section will describe the basic working principles of membranes and then, in more detail, describe UFDF.

2.3.1 The basics of membranes

A membrane acts as a porous barrier, allowing for selective transport and separation of molecules (Saxena et al. 2009). The flow of the sample can either be perpendicular or tangential to the membrane, referred to as normal flow filtration (NFF) and tangential flow filtration (TFF) respectively, see Figure 2.2. UFDF can be run as NFF or TFF depending on application, but only TFF will be considered in this master thesis as this is said to be standard for UFDF in the industry (Challener 2018). During TFF, the tangential flow hits the membrane at an angle, promoting backflow through the membrane, thus decreasing the plugging rate. (Lutz 2015).



(a) Schematic drawing of normal flow filtration.

(b) Schematic drawing of tangential flow filtration.

Figure 2.2

TFF has one entering feed stream and two exiting streams, one being the retentate stream and the other one being the permeate stream. The product can be in either the retentate or the permeate, depending on the application. For each stream a pressure drop can be defined, and is given as the pressure difference between the retentate and the feed and the permeate and the feed, referred to as delta P and transmembrane pressure (TMP) respectively. Since the pressure varies along the flow direction of the sample during TFF, often the average pressure for the feed/retentate side is used, in calculations of TMP. (Lutz 2015)

2.3.2 Ultrafiltration diafiltration (UFDF)

Ultrafiltration refers to filtration through a membrane with a pore size of 5-150 nm and is often used as a separation method in downstream processing, most commonly for final product formulation. It can be run in a dialysis mode, i.e. diafiltration, where a buffer solute is exchanged through diffusion across the membrane. This allows for displacement of old buffer, in favor of new.

The ultrafiltration is often operated in batch mode, where feed solution is stored in a tank and pumped to the membrane. The retentate is then recycled to the feed tank, to once again be filtered through the membrane. When switching over to diafiltration, new buffer is added to the tank. This causes the old buffer in addition with contaminants to pass through the membrane. The entire operation of an ultrafiltration/diafiltration process therefore requires multiple steps. (Lutz 2015)

The UFDF can also be operated in a continuous mode, where new buffer is added to the feed tank in the same rate as the flow of permeate. This results in no volume change in the feed tank while still purifying the product. (*Ultrafiltration for Bioprocessing* 2018)

2.4 Modeling approaches for UFDF

A UFDF run in batch mode can be modeled by dividing it into two parts, one UF part and one DF part, and setting up a system balance. By assuming no sieving and no osmotic pressure, making it into an ideal UFDF, the UF part can be described by Equation 2.6 and Equation 2.7. This part is run for a set time, after which it changes to DF. (*Ultrafiltration for Bioprocessing* 2018)

$$\frac{dV}{dt} = -JA \tag{2.6}$$

$$\frac{dc}{dt} = \frac{JA}{V}c \tag{2.7}$$

For the DF part, a volume of new buffer is added to the volume that is left after UF has been run, so that the original volume is once again reached. The new concentration is calculated according to Equation 2.8. If an ideal continuous UFDF is to be modeled, there will be no change in volume i.e. Equation 2.6 is set to zero. (*Ultrafiltration for Bioprocessing* 2018)

$$c_1 = \frac{V_{UF}}{V_0}c_0 \tag{2.8}$$

Due to fouling, polarization, osmotic pressure, etc, the membrane flux, J, will change during the filtration, and different ways of estimating the flux have been proposed in the literature. Hagen-Poiseuilles law, see Equation 2.9, is often proposed for modeling of flow through microporous structures. In this law the flux is estimated through pressure drop, viscosity, density, and channel dimensions. (Cheryan 1998;

Ultrafiltration for Bioprocessing 2018)

$$J = \frac{\epsilon d_p^2 TMP}{32\Delta x\mu} \tag{2.9}$$

The channel dimensions are sometimes replaced by an estimated retention factor, R_M, see Equation 2.10. Effects such as polarization and fouling can then be captured by adding an increased retention factor, R_G, to the original retention factor, R_M. (*Ultrafiltration for Bioprocessing* 2018)

$$J = \frac{TMP}{\mu R_M} \tag{2.10}$$

However, membrane processes handling proteins often show an insensitivity to changes in TMP. This insensitivity is explained by a phenomena called concentration polarization. Here, a layer of particles is amassed at the surface of the membrane at the retentate side, forming a "cake" or "gel layer" reducing the flux through the membrane. Increasing the TMP, will only result in a thicker "cake" and no increase in flux is seen, and decreasing the TMP will not remove this "cake". (Cheryan 1998) Because of this, no models based on TMP will be considered in this master thesis.

Ways of estimating the membrane flux without using TMP, is therefore desirable for UFDF processes handling proteins. One way is by using the so called gel model. The flux is in this model described by Equation 2.11. (Lutz 2015) This model assumes a boundary layer, with a concentration gradient, leading to a gel layer at the surface of the membrane. The flux is then said to be limited by mass transfer, more precisely the rate of which the solute is transported through the boundary layer, back into the bulk phase. (Cheryan 1998) This model will be used for modeling of UFDF in this master thesis.

$$J = k \cdot ln(\frac{c_g}{c}) \tag{2.11}$$

2.5 The ÄKTA Pure system

The ÄKTA Pure system is a modular chromatography system developed by Cytiva, for purification of proteins, peptides and nucleic acids, see 2.3. The system hardware, available as a 25 mL or 150 mL system, is controlled by the UNICORN software, and is highly versatile due to its modular design. Yet, some components come as standard. These are: two system pumps, a mixer, an injection valve, a UV monitor, and a conductivity monitor. (Cytiva 2021)

The two system pumps consist of two pairs of pump heads, creating a continuous flow rate to the mixer. A system pressure monitor is connected to the pumps allowing for automatic adjustment of the flow rate. (Cytiva 2021)

The mixer mixes the buffers used during gradient runs, and its chamber volume can be adjusted for the desired setup. The two available ÄKTA system sizes come with different options for mixer chamber volumes. (Cytiva 2021)

The injection valve makes it possible to apply samples to the system using different techniques. The sample can be applied through manually filled sample loops or SuperLoops, through sample loops or SuperLoops filled using a sample pump, or by directly applying the sample to the column using a sample pump. (Cytiva 2021)

The UV monitor can either be a fixed wavelength type or variable wavelength type. The monitor can be used to detect contaminants, labeled proteins or target molecules. (Cytiva 2021)

The conductivity monitor measures the conductivity of the buffers and samples, and automatically adjusts for variations caused by temperature. This enables for monitoring of the gradient. (Cytiva 2021)

In addition to the standard components, a vast array of additional components can be added to the setup.

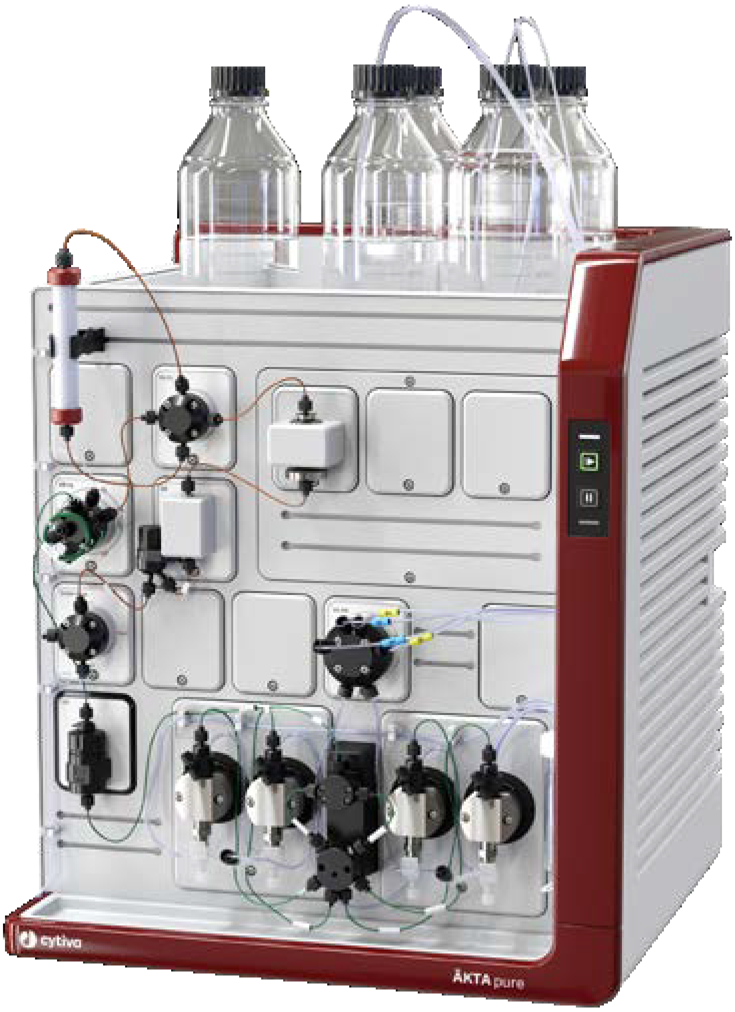


Figure 2.3 – The ÄKTA™ pure chromatography system. Image source: *ÄKTA ™ pure - Chromatography Systems* (Cytiva 2021).

2.6 The Orbit control system

The Orbit control system was developed at the department of Chemical Engineering at Lund University and is used to control chromatography processes run on ÄKTA systems, yet this work will limit itself to ÄKTA Pure systems. ÄKTA systems are traditionally run with the Unicorn software. This software has several limitations and Orbit, being entirely developed in Python, was designed to overcome these limitations. It is able to do so due to the Unicorn library being offered through a communication protocol, OPC. This enables another software, like Orbit, to communicate with the ÄKTA systems through the Unicorn server. (Andersson 2018)

Orbit has therefore been developed from the methods available from Unicorns OPC server, using the Python package OpenOPC to communicate with the server. Orbit is, like the Unicorn OPC server, built in a tree structure and some of the methods in Orbit are direct mappings to the Unicorn-methods. Other methods in Orbit are more advanced, adding functionality to the system. (Andersson 2018)

When controlling a process using Orbit, three different scripts are defined; a process script, a system script, and a script describing the phases of the process. The process script describes different sets of instructions and events for the phases and processes defined for the setup. The system script describes the different unit operations and how they are connected. In the script describing the phases used in the process, the phases are defined and put together, forming a process, thus constructing a set of instructions and events for the control system. (Andersson 2018)

2.6.1 The in-built simulator

The Orbit control system also has an in-built simulator, allowing for virtual lab runs to be made. Simulation of downstream processes has been of interest for quite some time, however, and many different chromatography simulation software have been developed prior to Orbit. A collection can be seen in Table 2.2. Some of these software, like CADET for example, are based on faster coding languages and superior numerical solving techniques compared to Orbit. (Leweke and Lieres 2018) However, unlike Orbit, these software do not allow the user to choose to either run a physical experiment or a simulation of an automatically generated setup within the same software.

Table 2.2 – Different chromatography simulation software developed prior to Orbit.

Software	**Description**
Aspen Chromatography ™	A flowsheet simulator developed by AspenTech for chromatographic separation processes. (AspenTech 2022)
Virtual Column ™	Ion chromatography separation simulator developed by Dionex. (ThermoFischer 2022)
ChromWorks ™	Chromatographic simulator developed by Ypso-Facto. (Ypso-Facto 2022)
ChromX	A mechanistic modeling software for chromatography processes. Developed at Karlsruhe Institute of Technology but as of 2020 marketed by Cytiva and GoSilico. (Cytiva 2020; T. Hahn, Heuveline, and Hubbuch 2012)
Chromulator-IEX	IEX chromatography simulator based on general rate modeling approach. Developed at Ohio University. (Gu 2022)
Preperative Chromatography Simulator (PCS)	Simulator, based on Matlab, for chromatography columns. Developed at Lund University. (Lund University 2018)
Chromatography Analysis and Design Toolkit (CADET)	Chromatography simulator based on C++ but with a Python frontend. Developed at the Research Center Jülich. The simulator is open source and free to use. (Leweke and Lieres 2018)

The Orbit-simulator allows for an entire setup, based on the ÄKTA system, to be simulated, including tubes, columns, valves etc. During these runs, no communication with the ÄKTA-system is needed, since the simulator is based on models described within Orbit itself. Also, the simulation is generated from the control scripts used for the experimental setup, providing the flexibility to choose between a virtual or physical experiment. Only minor additions to the script describing the phases of the process must be made, by defining the components and the buffers of the system. (Olsson 2019)

The classes and methods, already present in the simulator, are explained in Table 2.3, and the outline of the Orbit simulator is as follows:

```
class Component
class Buffer
def system:
class orbitSim
    __init__(self, S)
    runLoop(self)
    guiListen(self)
    guiSend(self)
    savesim(self)
    logItemSolution(self, item, folder, nameSuffix, where = 'outlet')
    getSolutionTime(self)
    getItemSolution(self)
    addTube(self, tube, Dax)
    addUnit(self, unit, Dax)
    getInit(self, item)
    addItem(self, z0, idx0, item, lastitem, pumpHead, Dax, port, port0)
    update_runc(self, S, Dax)
    setBuffers(self)
    runsim(self, instructions, plotFlag)
    plotItems(self, items, comps, where = 'outlet')
    plotsim(self, S, t, z, flag=[0,1,0])
    plotSimMeshSetUp(self, S, item, time)
    updateItemMesh(self, item, time)
    plotSimTimeUV(self)
    updateTimeUV(self, time)
    plotSimTimeCond(self)
    updateTimeCond(self, time)
```

Table 2.3 – List of already implemented classes and methods in the Orbit-simulator.

Class	Methods	
Component	getMw	Returns the molecular weight of a component.
	getK_uv	Returns UV-constant.
	setK_kin	Sets the kinetic coefficient to the given value.
	setH_0	Sets the equilibrium Henry coefficient to the given value.
	setbeta	Sets the β-value to the given value.
	getCompObject-Index	If provided with component name, returns index of component.
Buffer	fillSystem	Fills the System with the current buffer.
	fill	Fills provided unit with buffer.
	connect	Connects provided port or unit to buffer.
orbitSim	runLoop	Creates loop used for GUI.
	guiListen	Manages data from the GUI.
	guiSend	Sends message to GUI.
	savesim	Saves simulation data.
	logItemSolution	Saves the results of the simulation.
	getSolutionTime	Returns solution time.
	getItemSolution	Returns solution for provided item.
	addTube	Sets initial values and discretization for tubes.

	addUnit	Sets initial values and discrecization for units.
	getInit	Returns the initial values for a unit.
	addItem	Adds a new item to be simulated.
	update_runc	Finds the flow path, sets inflow and outflow indices, and sets direction for all tubes and columns.
	setComponents	Creates a list of components.
	setBuffers	Gather information about the components, buffers, and pumpheads. Also prints this information.
	runsim	Runs the simulation.
	plotItems	Plots the results for the items?.
	plotsim	Plots the results of the simulation?.
	plotSimMesh-SetUp	Sets up a mesh for an item?.
	updateItemMesh	Updates the mesh.
	plotSimTimeUV	Plots UV-signal?.
	updateTimeUV	Updates the axes created in plotSimTimeUV.
	plotSimTime-Cond	Plots the conductivity signal.
	updateTime-Cond	Updates the axes created in plotSimTime-Cond.

The simulator uses the method of lines to solve the mathematical models put up in the system-function. Here, the items are discretized through the finite volume method, turning the system of partial differential equations (PDEs) into a system of ordinary differential equations (ODEs) and/or differential algebraic equations (DAEs). These are then solved, in the runsim-method, with the solve_ivp-function, imported from the Python package scipy.integrate. The function solve_ivp solves the system of ODEs by integrating the system, given an inital value (The Scipy Community 2022a). To keep the time of the simulations relatively short, a grid size of 50 is used throughout this work.

3 Methods

This section will cover the programming made in Orbit, the comparison between different modeling approaches, and the parameter estimation.

For the testing of the models, a previously developed Orbit-script for a simple fake experiment called Cyto, was used, see Appendix B.1. This experiment contained two components, one called "Salt" and another called "Cyto" and these were used in varying concentrations in three different buffers, see Table 3.1. Some physical and kinetic data were predefined for the components within the script. The experiment consisted of six phases; one wash phase called "flush1", one injection phase called "inject", another wash phase called "flush2", one gradient phase called "eluate", one elution phase called "finalEluation", and one last wash phase called "clear". These phases were used to build a process, which could be simulated with the in-built Orbit-simulator.

The experiment set-up can be seen in Figure 3.1. In the original fake experiment, "Col1" in Figure 3.1 indicates an IEX chromatography column described through a simplified SMA adsorption description. The column type was, however, changed while testing, but some design parameters for the column were kept constant throughout the testing. These design parameters can be seen in Table 3.2.

The parameters used in the different models, except for the simplified SMA adsorption description, presented in the following sections have not been verified by physical experiments, but was rather adjusted for a reasonable peak to appear. The

Table 3.1 – Buffers defined in the fake experiment CYTO.

	Buffer 1 [$mol\ m^{-3}$]	**Buffer 2** [$mol\ m^{-3}$]	**Buffer 3** [$mol\ m^{-3}$]
Salt	0.5	500	0
Cyto	0	0	0.08

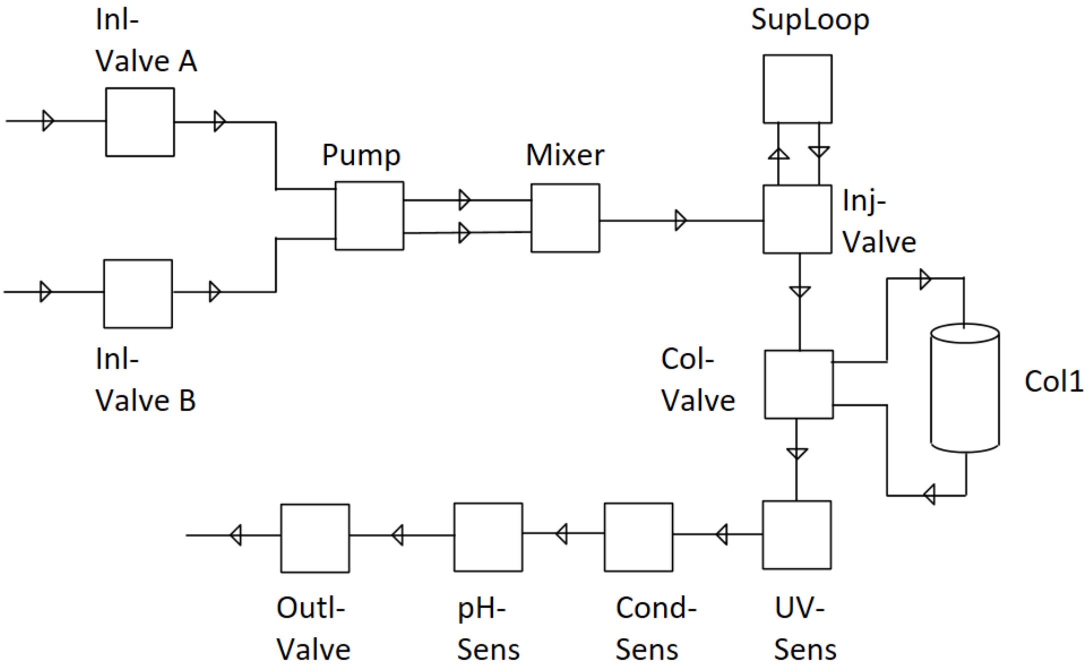


Figure 3.1 – Block diagram for the fake experiment Cyto.

Table 3.2 – Col1 design parameters kept constant throughout testing.

ϵ_c [-]	ϵ_p [-]	**Volume** $[mL]$
0.45	0.856	1

values of the parameters in the different models can be seen in Table 3.3.

Table 3.3 – List of model parameters resulting in the graphs presented in this section. The parameter values for the simplified SMA adsorption description were selected from Johan Olssons masters thesis (Olsson 2019). The rest of the values have no experimental basis and are therefore only to be seen as ad hoc.

Column Type	**Parameters**	**Units**
IEX-SMA	$k_{kin} = 1 \cdot 10^{-4}$ $K = 5 \cdot 10^{10}$ $q_{max} = 51$ $\nu = 4.1$	$[\frac{m^3}{mol \cdot s}]$ $[\frac{mol}{m^3}]$ $[\frac{mol}{m^3}]$ [-]
IEX-MPM	$k_{kin} = 1 \cdot 10^{-4}$ $K_0 = 5 \cdot 10^{8}$ $q_{max} = 51$ $\beta = 4.1$	$[\frac{m^3}{mol \cdot s}]$ $[\frac{mol}{m^3}]$ $[\frac{mol}{m^3}]$ [-]
SEC	$0 < K_e < 1$	[-]
HIC	$k_{kinM} = 1$ $K_{0M} = 50$ $q_{max} = 51$ $\gamma = 0.04$	$[\frac{m^3}{mol \cdot s}]$ $[\frac{m^3}{mol}]$ $[\frac{mol}{m^3}]$ $[\frac{m^3}{mol}]$
AC	$K_{0,IgG} = 4$ $pH_{ref} = 9$ $k_f = 1 \cdot 10^{-2}$ $q_{max} = 51$	$[\frac{mol}{m^3}]$ [-] $[\frac{1}{s}]$ $[\frac{mol}{m^3}]$

	$n = 4$	[-]

3.1 Increasing the flexibility of the modifier component

A modifier boolean was added as an attribute to the Component object, hence removing the need for modifier components to be added first to the list of components in the system. Setting the modifier tag to True will automatically put the component first in the list of components.

3.2 Adding models for chromatography columns

3.2.1 Introducing a LRMFT modeling approach

The already implemented LRM modeling approach was complemented by the possibility to choose between a LRM and a LRMFT modeling approach. This was achieved by introducing a third state variable, called c_p, to the simulator, when LRMFT is selected. Because of this introduction, all state variable vectors had to be extended by a number of elements equal to the number of components defined for the system. Also, a mass transfer coefficient was assigned to the Component-class, enabling for different mass transfer rates through the film for different components to be selected. A model attribute was introduced to the Column-class, allowing for LRM or LRMFT to be selected for any column. Also to the Column-class, a particle diameter attribute was introduced, needed for the calculation of film transfer rate.

The mass transfer coefficient in the LRMFT was then varied to investigate its influence on the system. The mass transfer coefficient was initially set to a high value, $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$, for all components, as to not be the limiting factor for transport into the intraparticulate volume. It was then lowered to $k_{eff} = 1 \cdot 10^{-5}\ m\ s^{-1}$, $k_{eff} = 1 \cdot 10^{-7}\ m\ s^{-1}$, and lastly to $k_{eff} = 1 \cdot 10^{-10}\ m\ s^{-1}$. The parameter values in the simplified SMA adsorption description was set to the values presented in Table 3.3, for all values of the mass transfer coefficient.

3.2.2 Adding an IEX-MPM adsorption description

The already implemented simplified SMA adsorption isotherm, see Equation 3.1, was complemented with a MPM adsorption isotherm as described in (Nilsson and Andersson 2017), see Equation 3.2.

$$r_{SIM,i} = k_{kin,i}(K_i c_i (1 - \sum_{j=1}^{n} \frac{q_j}{q_{max}})^{\beta} - s^{\beta} q_i) \tag{3.1}$$

$$r_{MPM,i} = k_{kinM,i}(K_{0,i} q_{max,i} c_i (1 - \frac{\sum_{j=1}^{n} q_{i,j}}{q_{max,i}}) - s^{\nu_i} q_i) \tag{3.2}$$

The type of isotherm is selected by setting the column type attribute in the Column-object to either IEX_SMA, for a SMA isotherm, or IEX_MPM, for a MPM isotherm. A dictionary was added to the Component-class, to be able to provide the parameters used in each adsorption description, for the components of the system. A dictionary of default parameters was added, so if no parameters are provided by the user, the default parameters will be selected and a simplified SMA isotherm used. These parameters, however, were selected in such a manner that no adsorption will be modeled, i.e. by setting k_{kin} to zero. The specific values of the remaining parameters in the adsorption description is thus not of importance, since the adsorption term will become zero for any values selected.

3.2.3 Adding an adsorption description for HIC

An adsorption description describing HIC, as described in (Nilsson and Andersson 2017), was also added to the simulator. Here, the dependency on salt concentration is captured by modifying the equilibrium constant by an exponential term in a kinetic version of a Langmuir adsorption, see Equation 3.3. In this model, the salt ions are assumed to promote hydrophobic interaction and it is therefore not possible to capture the different behaviors caused by different salts as described in Section 2.1.4. The isotherm is selected by setting the column type attribute in the Column-object to HIC.

Table 3.4 – Buffers defined for the testing of the HIC-model.

	Buffer 1 $[mol\ m^{-3}]$	**Buffer 2** $[mol\ m^{-3}]$	**Buffer 3** $[mol\ m^{-3}]$
Salt	500	0.5	500
Cyto	0	0	0.08

$$r_{HIC,i} = k_{kinM,i}(K_{0M,i}e^{\gamma_i s}q_{max,i}c_i(1 - \sum_{j=1}^{n} \frac{q_j}{q_{max,j}}) - q_i) \tag{3.3}$$

To be able to test the HIC-model, the buffers in the original fake experiment Cyto were changed, see Table 3.4, but the phases were kept from the original fake experiment Cyto.

3.2.4 Adding an adsorption description for pH-dependent AC

A large variety of affinity chromatography columns exist, all with different interaction mechanisms between target protein and the stationary phase of the column. In this master thesis, affinity chromatography for Protein A capture of antibodies was modeled. During this type of affinity chromatography, the interaction between target protein and the stationary phase of the column is governed by the pH in the column. The behavior can be described by internal mass transfer kinetics and a modified Langmuir adsorption description, as described in (Nilsson and Andersson 2017), and is presented in Equation 3.4.

$$r_{IgG,f} = k_f \frac{K_{0,IgG}(\frac{pH}{pH_{ref}})^n q_{max} c_{IgG}}{1 + K_0(\frac{pH}{pH_{ref}})^n c_{IgG}} - q_{IgG} \tag{3.4}$$

To be able to test the AC-model, a hydrogen ion modifier component was added to the original fake experiment Cyto, replacing the previous modifier component ”Salt”. This allowed for pH to be calculated within the simulator according to Equation 3.5. The component Cyto was renamed to ”IgG”, but no other changes were made to the component. The buffers were therefore also changed, see Table 3.5, now containing hydrogen ions instead of ”Salt” and IgG instead of Cyto. The hydrogen concentrations in the buffers did not have any experimental basis, but

Table 3.5 – Buffers defined for the testing of the AC-model. Note that the H+-concentrations will result in pH-values between 3 and 6.

	Buffer 1 $[mol\ m^{-3}]$	**Buffer 2** $[mol\ m^{-3}]$	**Buffer 3** $[mol\ m^{-3}]$
H+	$1 \cdot 10^{-6}$	$1 \cdot 10^{-3}$	$1 \cdot 10^{-6}$
IgG	0	0	0.08

were set to values resulting in pH-values between 3 and 6. The IgG concentrations were set to be the same as the concentrations of Cyto in the original fake experiment Cyto. The phases from the original fake experiment Cyto were kept for the testing of the AC-model.

$$pH = -log_{10}([H^+]) \tag{3.5}$$

3.2.5 Adjusting the LRM and LRMFT for SEC

In SEC, some molecules are too large to be able to pass into some of the pores in the particles. Thus, the available pore volume, for some molecules, is smaller than the actual pore volume in the column. This was reflected by introducing a so-called exclusion factor, K_e, as described in (Nilsson and Andersson 2017). This factor ranges from 1 to 0, 1 indicating that the entire pore volume is available to the molecule and 0 indicating that the molecule is unable to pass into any of the pores. During a LRM modeling approach of a SEC column, the column was described through Equation 3.6. The available liquid volume for a certain component was described through Equation 3.7.

$$\frac{\partial c_i}{\partial t} = D_{ax}\frac{\partial^2 c_i}{\partial z^2} - \frac{v}{\epsilon_a}\frac{\partial c_i}{\partial z} \tag{3.6}$$

$$\epsilon_a = \epsilon_c + (1 - \epsilon_c)\epsilon_p K_e \tag{3.7}$$

Note that the adsorption term is omitted from Equation 3.6 since no adsorption occurs during SEC.

Since the separating mechanism during SEC is physical, there is no need for a gradient phase during the fake experiment. The process was thus altered, to only consist of one wash phase, one injection phase, and another finishing wash phase. The finishing wash phase was set to be run for 6 minutes. This time has no experimental basis, and was only selected to be able to compare the resulting figures. The times from the original fake experiment was kept for the other two phases. The fake experiment was then run for two cases, full exclusion ($K_e = 0$), and no exclusion ($K_e = 1$).

Due to the introduced possibility to choose a LRMFT modeling approach instead of a LRM approach, the SEC model also had to be adapted to the LRMFT modeling approach. The size exclusion factor used for modeling SEC was introduced wherever the particle porosity was present in the LRMFT, to be able to capture the behavior of non-accessible pores. During a LRMFT modeling approach a SEC column was therefore described by Equation 3.8 and Equation 3.9.

$$\frac{\partial c_i}{\partial t} = D_{ax,i}\frac{\partial^2 c_i}{\partial z^2} - \frac{v_{sup}}{\epsilon_c}\frac{\partial c_i}{\partial z} - \frac{1-\epsilon_c}{\epsilon_c}\epsilon_p K_e \frac{\partial c_{p,i}}{\partial t} \tag{3.8}$$

$$\epsilon_p K_e \frac{\partial c_{p,i}}{\partial t} = k_{eff,i}\frac{3}{r_p}(c_i - c_{p,i}) \tag{3.9}$$

The modeling approach was then changed to LRMFT and the modified fake experiment was run for two cases, full exclusion ($K_e = 0$), and no exclusion ($K_e = 1$) with $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$ for both cases. k_{eff} was then lowered to $1 \cdot 10^{-7}\ m\ s^{-1}$ and then even further to $1 \cdot 10^{-10}\ m\ s^{-1}$ while keeping $K_e = 1$.

3.3 Adding a model for TFF membranes

To keep things simple, the TFF membrane was first treated as a splitter, modeled as something only resembling an algebraic block. By connecting a gradient pump to the membrane, one pumphead to the inlet and one pumphead to the retentate, the flows for the feed, the retentate, and the permeate would automatically be

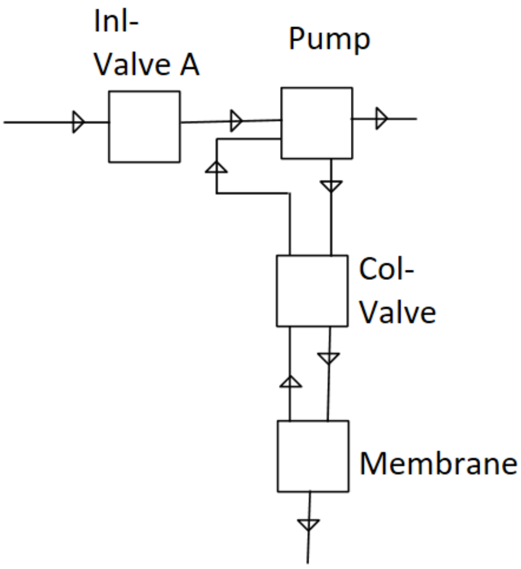


Figure 3.2 – Block diagram for the fake experiment used for testing of the membrane model.

calculated by the simulator. By assigning a retention factor, a measure of how much is retained in the retentate, to the Component-objects, the amount of component in the retentate and permeate would be calculated. The original plan was then to increase the complexity, by implementing a dynamic model for the membrane, progressing towards a gel model. This was, however, not done and the reason for this will be further explained in Section 4.3.

Due to the fake experiment Cyto, not containing any membranes, entirely new scripts were developed for the testing of the membrane model. Here, a system of one inlet valve, one column valve, one gradient pump, and one UFDF membrane was connected. The set-up can be seen in Figure 3.2. Here, the first pumphead was connected to the inlet valve, while the second pumphead was connected to the retentate, coming from the membrane. This caused the permeate flow to be automatically calculated by the simulator. The buffers from the fake experiment Cyto were kept, and the system was filled with buffer 1, and buffer 3 was connected to the inlet valve. Only one phase was created, a gradient phase with a step gradient for setting the pumpheads. See Appendix B.2 for syntax.

3.4 Development of a sensitivity method

A method, called sensitivity, allowing for sensitivity analyses to be made for a chosen parameter was developed for the OrbitSim-class in the simulator. This was achieved by first adding two methods to the Component-class, one for retrieving a specific parameter, called getParameter, and one for setting a specific parameter, called setParameter. The getParameter-method was set to take two arguments, a string with the name of the parameter and a string with the column type for which the parameter is defined. The setParameter-method was set to take three arguments, a string with the column type for which the parameter is defined, a string with the name of the parameter, and a float with the value the parameter is going to be changed to. See Appendix A.2 for syntax.

The sensitivity-method in the OrbitSim-class was set to take five arguments; instructions for running a process, a string with the name of the parameter to be analyzed, a string with the name of the component for which the parameter was defined, a column-object indicating for which column the sensitivity analysis is to be conducted, and lastly a list with two factors. The factors indicate how many times smaller or larger than the original parameter the lower and upper bound will be. The sensitivity-method was set to plot a figure showcasing the simulated results for the lower bound, the original value, and the upper bound. See Appendix A.3 for syntax.

3.5 Comparison of LRM and LRMFT

A comparison between the LRM and the LRMFT modeling approaches was also made, while using a simplified SMA adsorption isotherm.

The parameter k_{kin} in the adsorption isotherm was set to $2 \cdot 10^{-11}\ m^3\ mol^{-1}\ s^{-1}$, to decrease the reaction speed, allowing for changes of the parameter k_{kin} to be more noticeable in the figures. A sensitivity analysis, using the developed sensitivity-method, with lower bound factor set to 1/10 and the upper bound factor to 10, for the parameter k_{kin} was then conducted. This was done for both the LRM and the LRMFT, to investigate how the reaction kinetics influenced the elution of the target during the two modeling approaches. The same procedure was then repeated for the parameters K and q_{max}.

3.6 Construction of a two column process

To test the ability to simulate multiple columns, a fake experiment with one IEX column and one SEC column was constructed, see Figure 3.3. These columns were modeled through a LRM modeling approach with a simplified SMA adsorption description for the IEX column. The design parameters presented in Table 3.2 were kept for the IEX column, while the column void was decreased to 0.2 for the SEC column.

Further on, three components were defined; "Salt" as modifier component, and two protein components, "Cyto" and "Zyto". For the proteins, model parameters were selected to achieve somewhat different behaviors of the proteins in the two columns, see Table 3.6.

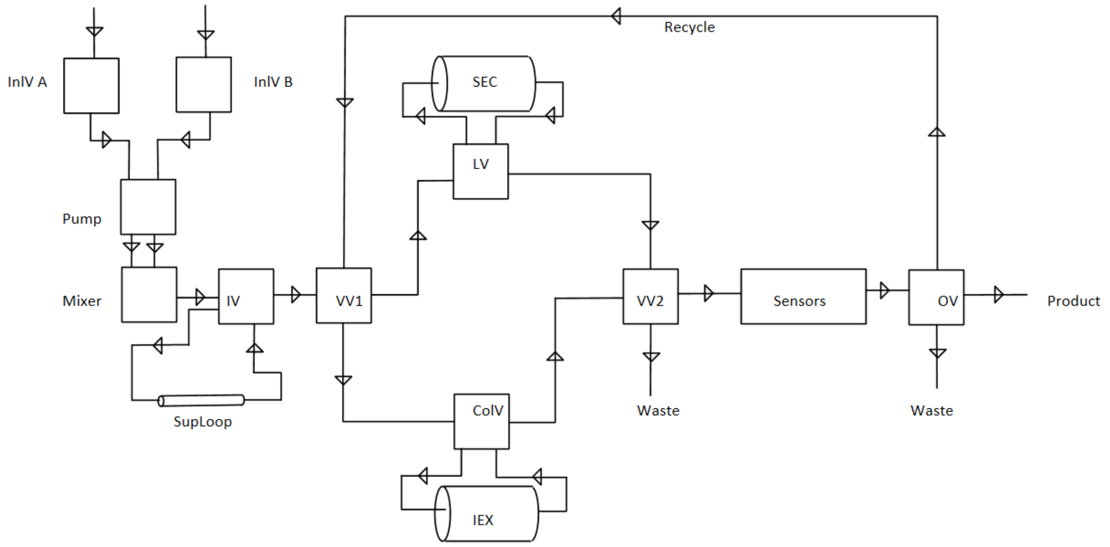


Figure 3.3 – Block diagram for the fake experiment with two columns. InlV A - Inlet valve A, InlV B - Inlet valve B, IV - Injection valve, SupLoop - Super loop, VV1 - Versatile valve 1, ColV - Column valve, IEX - Ion exchange chromatography column, VV2 - Versatile valve 2, OV - Outlet valve, LV - Loop valve, and SEC - Size exclusion chromatography column. Sensors is a series of sensors, including a UV-sensor, a conductivity sensor and a pH-sensor, in that order.

Table 3.6 – Parameters for Cyto and Zyto used in the simplified IEX-SMA model and the SEC model.

	IEX-SMA				**SEC**
	k_{kin} $[m^3\ mol^{-1}\ s^{-1}]$	K $[mol\ m^{-3}]$	q_{max} $[mol\ m^{-3}\ ]$	ν [-]	K_e [-]
Cyto	$1 \cdot 10^{-4}$	$5 \cdot 10^{10}$	51	4.1	0
Zyto	$1 \cdot 10^{-4}$	$7 \cdot 10^{10}$	51	4.1	1

Table 3.7 – Buffers defined for the fake experiment with two columns.

	Buffer 1 $[mol\ m^{-3}]$	**Buffer 2** $[mol\ m^{-3}]$	**Buffer 3** $[mol\ m^{-3}]$
Salt	0.5	500	0.5
Cyto	0	0	0.08
Zyto	0	0	0.08

Three different buffers where defined, the only difference from the fake experiment Cyto being the addition of concentrations for Zyto, see Table 3.7. These were for simplicity's sake set to be equal to that of Cyto in the same buffer. Buffer 1 was connected to inlet valve A and Buffer 2 to inlet valve B. The super loop was filled with Buffer 3.

The experiment was constructed with six phases; one equilibration phase called ”flush 1”, one injection phase called ”inject”, another wash phase called ”Wash11”, one gradient phase called ”Elution1”, a wash phase called ”Wash2”, and yet another wash phase called ”Wash12”. These are explained in Table 3.8. During the phases ”flush1”, ”inject”, and ”Wash11”, the outlet valve was set to waste, see Figure 3.4, and not changed until the target eluted during the phase ”Elution1”. The outlet valve was changed to recycle at 17 minutes into the phase ”Elution1”, causing the elution peak to be recycled, i.e. pooled, to the SEC column, see Figure 3.5. At 20 minutes into ”Elution1” the outlet valve was changed back to waste.

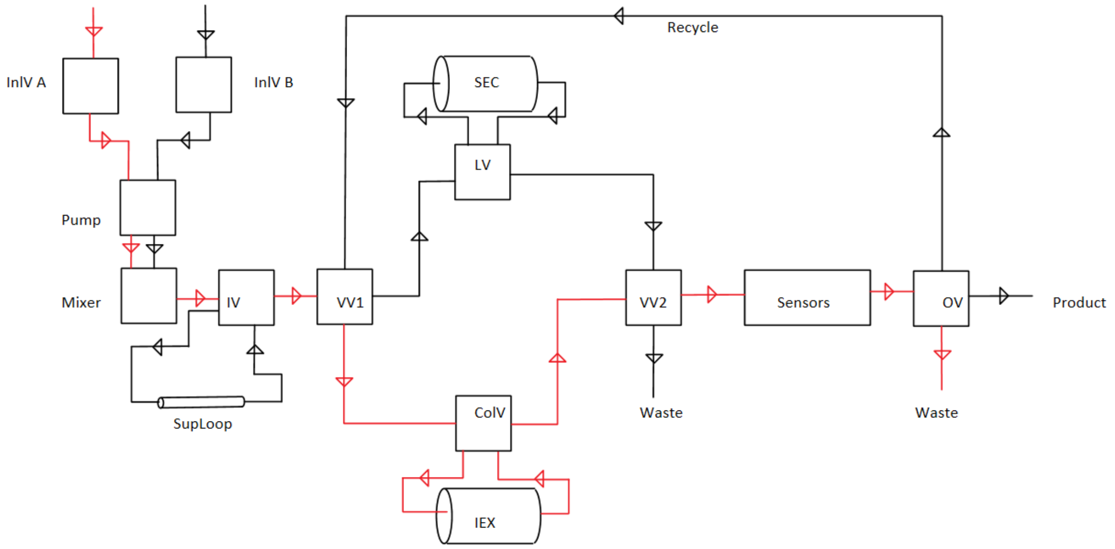


Figure 3.4 – Block diagram for the fake experiment with two columns. The path marked in red shows the flow path during ”flush1”, ”wash11” ,”wash12”, and the early stages of the elution phase. During the injection phase the super loop will also be included in the flow path. InlV A - Inlet valve A, InlV B - Inlet valve B, IV - Injection valve, SupLoop - Super loop, VV1 - Versatile valve 1, ColV - Column valve, IEX - Ion exchange chromatography column, VV2 - Versatile valve 2, OV - Outlet valve, LV - Loop valve, and SEC - Size exclusion chromatography column. Sensors is a series of sensors, including a UV-sensor, a conductivity sensor and a pH-sensor.

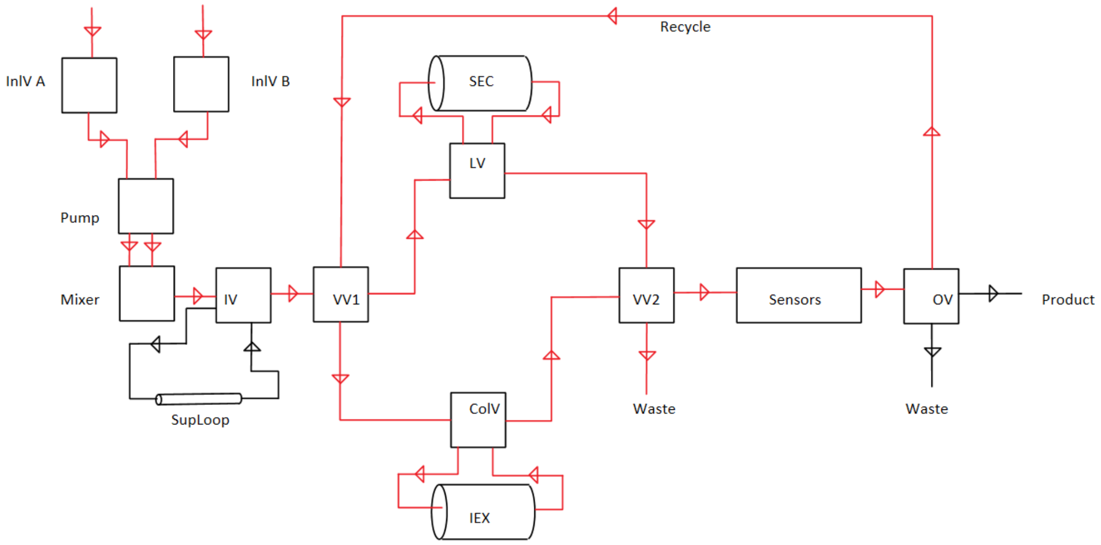


Figure 3.5 – Block diagram for the fake experiment with two columns. The path marked in red shows the flow path during the pooling of the elution peak from the IEX column. InlV A - Inlet valve A, InlV B - Inlet valve B, IV - Injection valve, SupLoop - Super loop, VV1 - Versatile valve 1, ColV - Column valve, IEX - Ion exchange chromatography column, VV2 - Versatile valve 2, OV - Outlet valve, LV - Loop valve, and SEC - Size exclusion chromatography column. Sensors is a series of sensors, including a UV-sensor, a conductivity sensor and a pH-sensor.

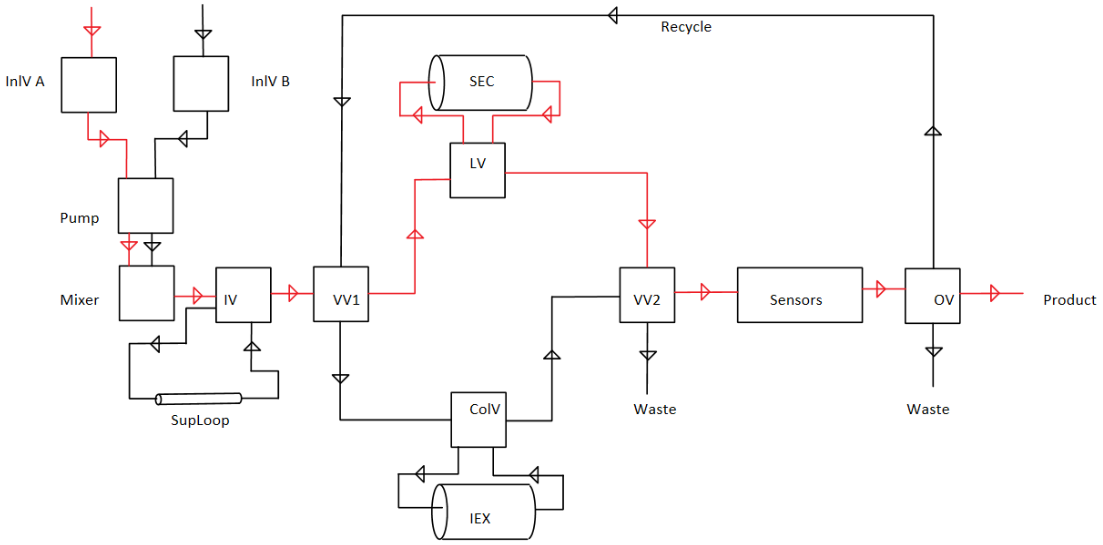


Figure 3.6 – Block diagram for the fake experiment with two columns. The path marked in red shows the flow path during the pooling of the elution peak from the SEC column. InlV A - Inlet valve A, InlV B - Inlet valve B, IV - Injection valve, SupLoop - Super loop, VV1 - Versatile valve 1, ColV - Column valve, IEX - Ion exchange chromatography column, VV2 - Versatile valve 2, OV - Outlet valve, LV - Loop valve, and SEC - Size exclusion chromatography column. Sensors is a series of sensors, including a UV-sensor, a conductivity sensor and a pH-sensor.

Table 3.8 – Defined phases in the fake experiment with two columns.

Phase	**Duration [min]**	**Description**
flush1	1	An equilibration phase, using Buffer 1, for the IEX column.
inject	1	An injection phase where Buffer 3 is injected into the IEX column.
Wash11	1	A wash phase where the IEX column is washed with Buffer 1.
Elution1	20	A gradient phase where a mixture of Buffer 1 and 2, starting at 0 % Buffer 2 and gradually increasing to 100 %, is pumped through the IEX column. As the target elutes, recirculation is activated and the sample is led to the SEC column, for further separation. Recirculation is deactivated at the end of the peak.
Wash2	5	Buffer 1 is pumped through the SEC column, washing out the sample.
Wash12	5	The IEX column is washed with Buffer 1, restoring it to low salt concentrations.

During the phase ”Wash2”, versatile valve 1 was changed so that the buffer was directly sent to the SEC column. At four minutes into ”Wash2”, versatile valve 2 was set to lead the sample to the outlet valve, i.e. pooling the elution peak. The outlet valve was at the same time set to fraction, i.e. to return a product, see Figure 3.6. At five minutes into ”Wash2”, the two valves were changed back to their previous settings. During the last wash phase, ”Wash12”, the flow path was changed back to that presented in Figure 3.4. See Appendix B.3 for syntax.

3.7 Development of a calibration-method

A calibration-method, called calibrate, was developed for the OrbitSim-class. But to enable for a calibration to be made, a layer function called layer, was first introduced to the simulator. The layer function was set to take six arguments, see Table 3.9, and four optional arguments.

Table 3.9 – Arguments for the added layer-function. The function also takes four additional optional arguments.

Argument	**Explanation**
t	An array of time data points.
parameters	A list of strings with names of the parameters to be calibrated.
instructions	Instructions for running a process.
orbsim	An OrbitSim-object.
comp_name	A string with the name of the component for which the parameters are to be calibrated.
item	An item-object for which item in the system the simulated data is to be measured.

The four optional arguments were set to be floats with parameter values. The layer function was designed to be able to handle anywhere between one up to four guessed parameters, but can easily be extended to handle more. The system-function sets the chosen parameters to their guessed values and calls the runsim-method on the OrbitSim-object, running a simulation according to the provided instructions.

To the runsim-method, a dictionary called calibration is also passed by the layer-function. This was an already existing optional argument for the runsim-method, and the layer-function was set to sort the provided item-object under the already implemented ”item”-key in the dictionary. The layer-function was, however, also set to provide a new key, called ”t_Data” to the dictionary. Sorted to this key was the provided array of time data points. These data points were set to be used as evaluation points in the numerical solver, used for solving the system of ODEs, in the runsim-method. See Appendix A.5 for syntax.

This enabled for the calibrate-method to be run on an OrbitSim-object. The calibrate-method was set to take five arguments, see Table 3.10, and one optional argument containing bounds for the calibration.

Table 3.10 – Arguments for the calibration-method. The method also takes one additional optional argument.

Argument	**Explanation**
file_name	A string with the name of the csv-file from which the data, to which the parameters are to be calibrated against, is read.
parameters	A list of lists containing three elements; two strings, one with the name of the column type for which the parameter is defined and one with the name of the parameter, and a guessed value.
instructions	Instructions for running a process.
comp_name	A string with the name of the component for which the parameters are to be calibrated.
item	An item-object for which item in the system the simulated data is to be measured.

The function curve_fit, available in scipy.optimize, was used in the calibrate-method for fitting the layer-function to the read data. curve_fit uses non-linear least squares with a Levenberg-Marquardt algorithm to fit a function to data. (The Scipy Community 2022b) This work will not go into any further detail on the algorithm and the interested reader is instead referred to the documentation concerning curve_fit. When provided with one parameter, the parameter, regardless of which parameter, was scaled by the tenth logarithm, before passed to curve_fit, and then scaled up within the layer-function. Here, the step length, in curve_fit named epsfcn, and the desired relative error, in curve_fit named ftol, were adjusted to $1 \cdot 10^{-8}$ and $1 \cdot 10^{-12}$ respectively. This was made to avoid too large steps to be taken, causing calculation errors, and for the solution to be more accurate. When provided with multiple parameters, the parameters were scaled by dividing with the guessed value, before passed to curve_fit, and then scaled up within the layer-function. No adjustment of step length nor desired relative error was made this time. The calibrate-method was then set to plot the provided data, the simulated data for the guessed parameter values, and the simulated data for the calibrated parameters. See Appendix A.6 for syntax.

To be able to read csv-files, a function called readLog was developed and added to the simulator. This function takes a string containing the name of the csv-file and returns a dictionary with the data in the csv-file. See Appendix A.4 for syntax.

3.8 Parameter estimation

For the parameter estimation, only parameters in the simplified SMA adsorption description and the exclusion factor will be estimated. Design parameters of the chromatography columns, such as column volume, packing porosity, column length etc, will not be considered during the parameter estimation in this work.

3.8.1 Parameter estimation in a single chromatography column

To test the calibration method, a data set was generated by simulating the fake experiment Cyto using a simplified SMA adsorption description, with parameter values according to Table 3.3, for the IEX column. This data set was logged to a csv-file using the preexisting logItemsolution method, on the UV-sensor. A simulated data set was generated since no experimental data was available for the fake experiment Cyto.

The calibration method was then tested by providing a number of guessed values for different parameters in the simplified SMA adsorption description and calibrating them to the simulated data set, for the Cyto component. Firstly, the calibration method was tested for parameter estimation of lone parameters. Here, a guess was provided for a single parameter while keeping the others to the values presented in Table 3.3. This was conducted for all parameters except ν since the simplified SMA adsorption description is too sensitive to changes of ν, often causing errors to be generated within the simulator if ν is changed. Also, a change of the value of ν would cause a change in units for the other parameters in the adsorption description.

For each parameter a parameter estimation was conducted for four different guessed values. The guessed values can be seen in Table 3.11. The guessed values were chosen to represent two values below the parameter value used for data generation and two values above the parameter value used for data generation.

Table 3.11 – Guessed values of the parameters during the lone parameter estimation test of the calibration method.

	Guess 1	**Guess 2**	**Guess 3**	**Guess 4**
k_{kin} $[m^3\ mol^{-1}\ s^{-1}]$	$1 \cdot 10^{-12}$	$1 \cdot 10^{-10}$	$1 \cdot 10^{-3}$	$1 \cdot 10^{-2}$
K $[mol\ m^{-3}]$	$5 \cdot 10^{8}$	$4 \cdot 10^{10}$	$6 \cdot 10^{10}$	$5 \cdot 10^{12}$
q_{max} $[mol\ m^{-3}]$	10	45	55	100

Then, the number of parameters were successively increased to test the calibration methods capability to estimate multiple parameters. The guessed values and combination of estimated parameters can be seen in Table 3.12.

Table 3.12 – Guessed values of the parameters during the multiple parameter estimation test of the calibration method.

	Estimated parameters	**Guessed values**
Estimation 1	k_{kin} $[m^3\ mol^{-1}\ s^{-1}]$ K $[mol\ m^{-3}]$	$1 \cdot 10^{-12}$ $4 \cdot 10^{9}$
Estimation 2	k_{kin} $[m^3\ mol^{-1}\ s^{-1}]$ q_{max} $[mol\ m^{-3}]$	$1 \cdot 10^{-12}$ 10
Estimation 3	K $[mol\ m^{-3}]$ q_{max} $[mol\ m^{-3}]$	$4 \cdot 10^{10}$ 10
Estimation 4	k_{kin} $[m^3\ mol^{-1}\ s^{-1}]$ q_{max} $[mol\ m^{-3}]$ K $[mol\ m^{-3}]$	$1 \cdot 10^{-12}$ 10 $4 \cdot 10^{10}$

3.8.2 Parameter estimation in multiple chromatography columns

The calibration methods capability to estimate multiple parameters for multiple columns was also tested. For this test the fake experiment with two columns was used to generate a data set, with parameters set according to the values presented in Table 3.13. The data set was logged to a csv-file using the preexisting logItemSolution method, on the UV-sensor. A simulated data set was generated since no experimental data set was available for the fake experiment with two columns.

Then, a parameter estimation of multiple parameters in different chromatography columns for the Zyto component was made. First, a parameter estimation of k_{kin} in the simplified SMA adsorption description used for modeling of the IEX column, and the exclusion factor K_e in the SEC column was made. Then, k_{kin} was exchanged by K and another parameter estimation conducted. Following that, K was exchanged by q_{max}, and yet another estimation was made. Lastly, a parameter estimation for k_{kin}, K, q_{max}, and K_e was made.

Table 3.13 – Guessed values of the parameters during the multiple parameter estimation test in different columns during a multi column setup.

	Estimated parameters	**Guessed values**
Estimation 1	k_{kin} $[m^3\ mol^{-1}\ s^{-1}]$	$1 \cdot 10^{-12}$
	K_e [-]	0.2
Estimation 2	K $[mol\ m^{-3}]$	$5 \cdot 10^{10}$
	K_e [-]	0.2
Estimation 3	q_{max} $[mol\ m^{-3}]$	10
	K_e [-]	0.2
Estimation 4	k_{kin} $[m^3\ mol^{-1}\ s^{-1}]$	$1 \cdot 10^{-12}$
	q_{max} $[mol\ m^{-3}]$	10
	K $[mol\ m^{-3}]$	$5 \cdot 10^{10}$
	K_e [-]	0.2

4 Result and Discussion

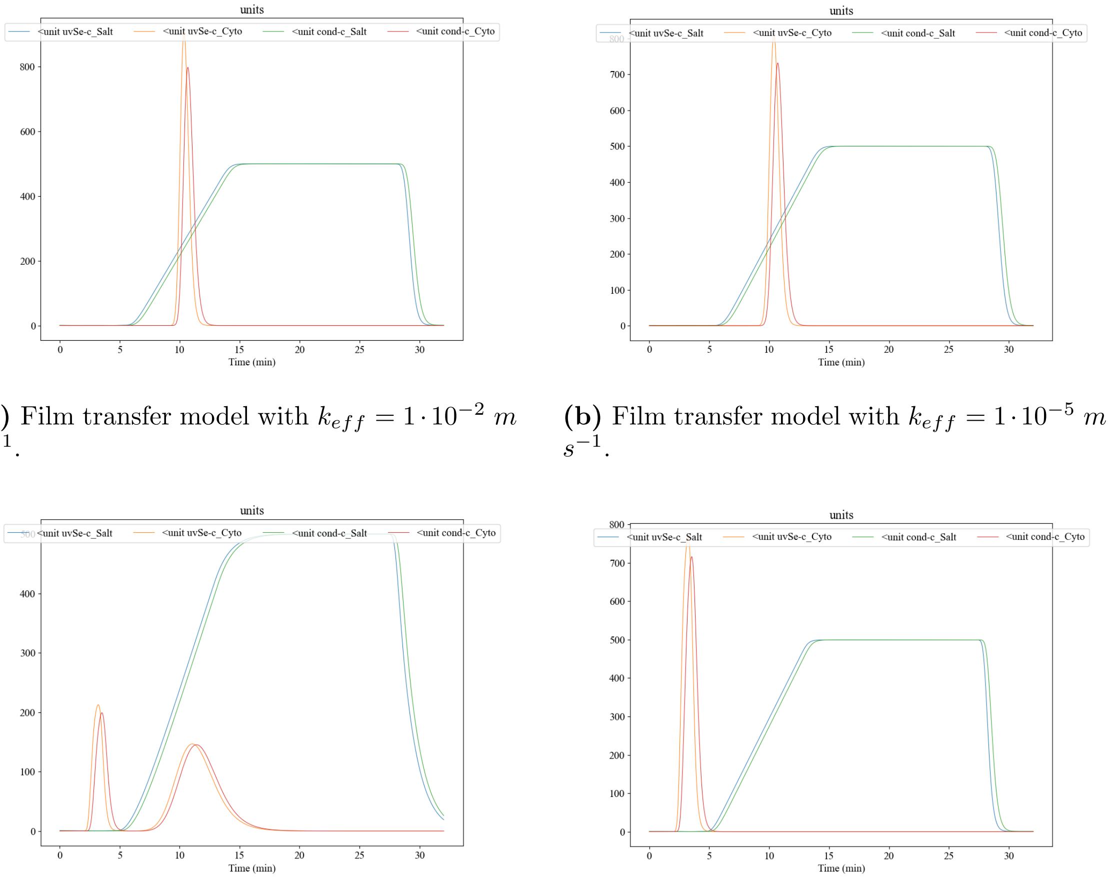
In this section the results from the implemented LRMFT modeling approach, the different models for IEX, SEC, HIC, and AC, and the parameter estimation are presented. The results presented for the implemented adsorption descriptions in the following sections are based on a LRM modeling approach for all chromatography types.

4.1 Modeling approaches and adsorption descriptions

An integration of a film transfer model into the simulator was made and it is now possible to choose between a LRM and a LRMFT, when simulating a chromatography column, potentially allowing for more physio-chemical behaviors to be captured, see Figure 4.1. However, usage of the LRMFT is not advised and the reason for this is further explained in Section 4.2.

In Figure 4.1a the mass transfer coefficient in the film transfer model was set to a high value, $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$, resulting in a high mass transfer between the bulk phase and the particle phase, thus being limited only by the concentration difference between the bulk phase and the particle phase. This explains the high peak and late elution of the target.

In Figure 4.1b, the mass transfer coefficient was lowered to $k_{eff} = 1 \cdot 10^{-5}\ m\ s^{-1}$, resulting in a lower and broader peak compared to $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$. This is reasonable since a lower mass transfer coefficient will have a ”retarding” effect on the transport of the components. This will lead to a broader peak as the retention time for some molecules will be longer.



(a) Film transfer model with $k_{eff} = 1 \cdot 10^{-2}$ m s^{-1}.

(b) Film transfer model with $k_{eff} = 1 \cdot 10^{-5}$ m s^{-1}.

(c) Film transfer model with $k_{eff} = 1 \cdot 10^{-7}$ m s^{-1}.

(d) Film transfer model with $k_{eff} = 1 \cdot 10^{-10}$ m s^{-1}.

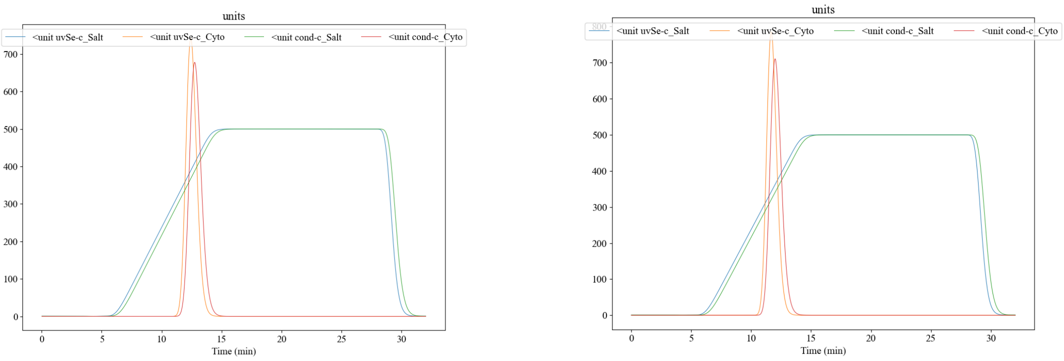
Figure 4.1 – Film transfer models for IEX with a simplified SMA adsorption description and varying mass transfer coefficients, k_{eff}. The unit for the y-axis is mol m^{-3} but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

In Figure 4.1c, the mass transfer coefficient was lowered to $k_{eff} = 1 \cdot 10^{-7}$ m s^{-1}, resulting in two low peaks at different elution times. This is due to the mass transfer being low between the particles and the bulk, causing some of the target protein to be flushed through the column without entering the particles, thus not allowing for it to be adsorbed to the stationary phase. This explains the first elution peak. Some protein will however enter the particles, and adsorb to the stationary phase, and desorb at a certain modifier concentration, explaining the second elution peak. The ”retarding” effect is as expected even greater with this mass transfer coefficient, notice the salt profile in Figure 4.1c.

In 4.1d, the mass transfer coefficient was lowered even further, to $k_{eff} = 1 \cdot 10^{-10}$ m s^{-1}, resulting in almost no mass transfer between the bulk and the particle. This causes the entirety of the target to be flushed through the column without entering the particles, leading to no adsorption, thus explaining the lone early elution peak. However, this time no ”retarding” effect is seen. This is due to only a very limited amount of the components being able to enter the particles before being flushed straight through the column.

4.1.1 Comparing an IEX-SMA to an IEX-MPM

The added IEX-MPM adsorption description is able to capture the behavior of an IEX system. Compared to the already implemented IEX-SMA adsorption description no significant difference is seen. The peaks in Figure 4.2 appear to display differing elutions but this is due to the selection of adsorption parameters in the two adsorption descriptions. The current parameters have been chosen in such a manner that the peaks will elute at reasonable salt concentrations, and at somewhat similar times but is not verified by actual experiments. For any future comparisons, a calibration to experimental data should be made, to allow for a more just comparison.



(a) LRM IEX SMA.

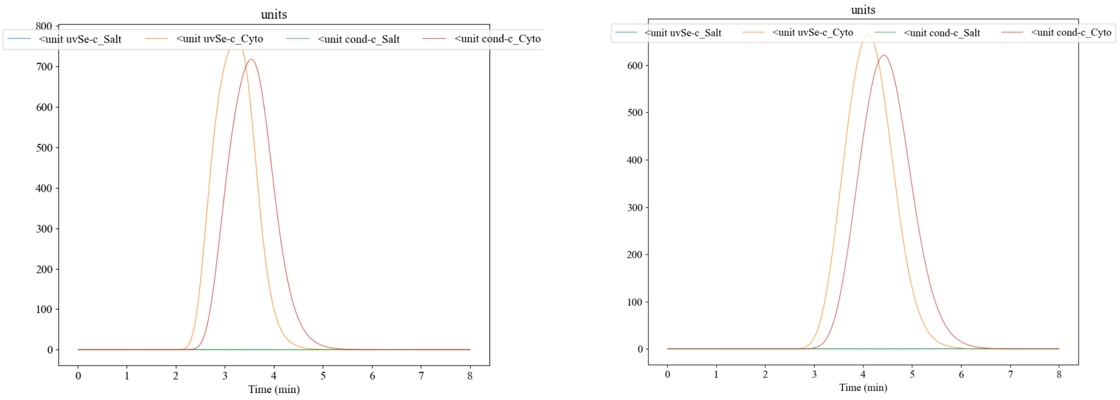
(b) LRM IEX MPM.

Figure 4.2 – SMA and MPM adsorption description for an IEX-column. The unit for the y-axis is mol m^{-3} but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

4.1.2 Results of the SEC model

The SEC model was also successfully implemented into the simulator, and the results for SEC with full exclusion, ($K_e = 0$), and no exclusion, ($K_e = 1$), can be seen in

Figure 4.3. ($K_e = 1$) results in slightly later elution of the target, than ($K_e = 0$), as expected since the available volume is increased to that of $\epsilon_a \cdot V_{col}$. It is however recommended to compare and to calibrate against experimental data to validate the SEC model. No experimental data was available for an experiment comparable to the modified fake experiment Cyto, used for the testing of the SEC model. Thus no validation of the SEC model could be made in this work.



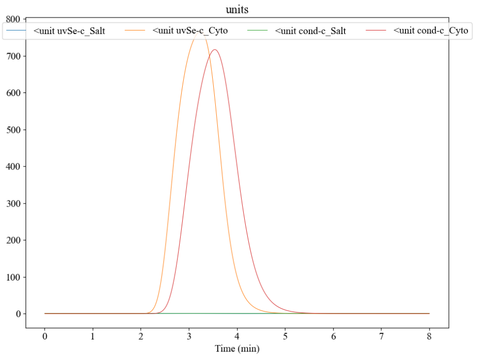
(a) LRM SEC with $K_e = 0$.

(b) LRM SEC with $K_e = 1$.

Figure 4.3 – SEC at different exclusion factors during a LRM modeling approach. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

For the SEC model during a LRMFT modeling approach with $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$, with the same degrees of exclusion as in the LRM modeling approach, the results can be seen in Figure 4.4. The results are identical between the two modeling approaches, suggesting that the SEC model works in both modeling approaches.

For $k_{eff} = 1 \cdot 10^{-7}\ m\ s^{-1}$ and no exclusion, the results can be seen in Figure 4.5. $k_{eff} = 1 \cdot 10^{-7}\ m\ s^{-1}$ results in a "tailing" effect on the peak, see Figure 4.5a. This could be explained by the retarding effect caused by the lowered mass transfer through the film, resulting in even longer retention times. Also, the peak is starting to appear at the same time as with the case with full exclusion, ($K_e = 0$). This is as expected since at this mass transfer rate, some of the protein will be flushed straight through the column, see Figure 4.1c, similar to what is seen at full exclusion.

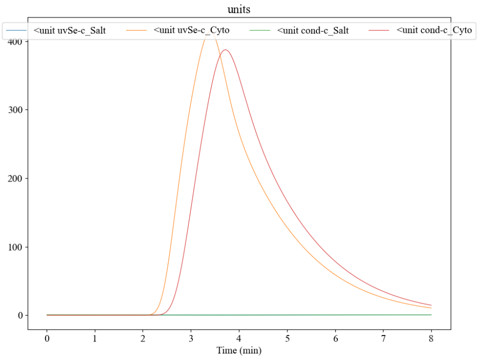


(a) LRMFT SEC with $K_e = 0$ and $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$.

(b) LRMFT SEC with $K_e = 1$ and $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$.

Figure 4.4 – SEC at different exclusion factors during a LRMFT modeling approach with $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

For $k_{eff} = 1 \cdot 10^{-10}\ m\ s^{-1}$ and no exclusion, the results can be seen in Figure 4.5. As expected, the peak is very similar to that achieved with full exclusion, $(K_e = 0)$. This can be explained by the mass transfer being low enough for almost no molecules to enter the particles, much like what is achieved with full exclusion.



(a) LRMFT SEC with $K_e = 1$ and $k_{eff} = 1 \cdot 10^{-7}\ m\ s^{-1}$.

(b) LRMFT SEC with $K_e = 1$ and $k_{eff} = 1 \cdot 10^{-10}\ m\ s^{-1}$.

Figure 4.5 – SEC at no exclusion during a LRMFT modeling approach with different k_{eff}. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

4.1.3 Results of the HIC model

The HIC model was also successfully implemented into the simulator, and the results for a simulation of a HIC column can be seen in Figure 4.6. Note the reversed gradient profile, as the target protein will elute at low salt concentrations instead of high during HIC. It is however, once again, recommended to compare and to calibrate against experimental data to validate the HIC model. No experimental data was available for an experiment comparable to the modified fake experiment Cyto, used for the testing of the HIC model. Thus, no validation of the model could be made in this work. The results still indicate, though, that it is possible to capture elution caused by a reversed salt gradient.

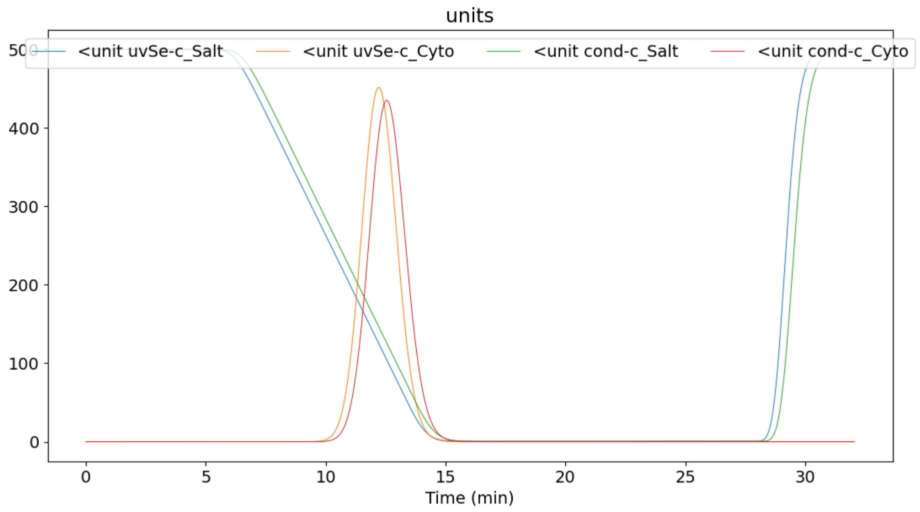


Figure 4.6 – HIC column modeled with the LRM modeling approach. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

4.1.4 Results of the AC model

An AC model for pH-dependent adsorption was also successfully implemented to the simulator, and the results for a simulation of an AC column can be seen in Figure 4.7. The elution peak is rather small only reaching slightly above 8 $mmol\ m^{-3}$. This could be due to how the modified fake experiment is designed, but most likely it is due to how the model parameters were chosen. These parameters were adjusted only for a peak to appear, and have no experimental basis. It is therefore possible, that a set of parameters were selected, resulting in a very weak elution.

This chromatogram also displays a completely different peak compared to previous simulations using the same adsorption description, presented in the literature (Nilsson and Andersson 2017; Ng et al. 2012). However, such a comparison is not meaningful to make, since the chromatogram generated by the Orbit-simulator is based on an entirely different, imaginary process, with different process steps, components, design parameters etc. For a just comparison to be made, an entirely new process script should be designed, mimicking the processes presented by Nilsson and Andersson, and Ng et.al. (Nilsson and Andersson 2017; Ng et al. 2012). Alternatively, a physical experiment resembling the fake experiment used for testing of the AC model, should be made. This is beyond the scope of this work, which sadly also makes it impossible to evaluate the validity of the implemented AC model. Nonetheless, the model is implemented and the results suggest that it is possible to capture elution due to a decrease in pH.

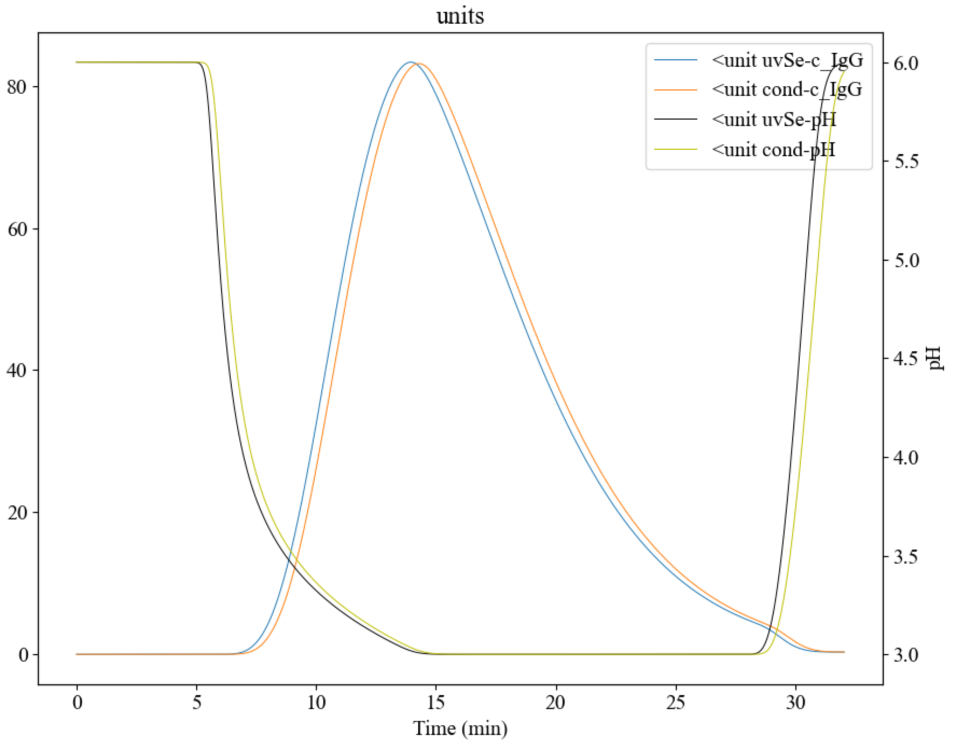
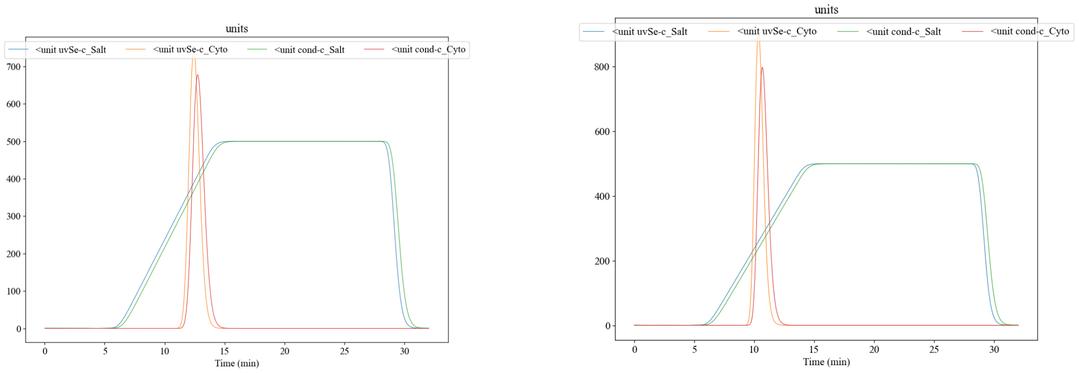


Figure 4.7 – AC column modeled with the LRM modeling approach. The unit for the left-hand y-axis is $mol\ m^{-3}$ but the signal for IgG is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.



(a) LRM for IEX with a SMA adsorption description.

(b) Film transfer model for IEX with a SMA adsorption description and $k_{eff} = 1 \cdot 10^{-2}$ m s^{-1} for all components.

Figure 4.8 – Comparison of the LRM modeling approach and the LRMFT modeling approach with a mass transfer coefficient of $k_{eff} = 1 \cdot 10^{-2}$ m s^{-1}. The unit for the y-axis is mol m^{-3} but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

4.2 Comparing the LRMFT to the LRM

The chromatograms differ between the LRM and the LRMFT for an IEX column with a simplified SMA adsorption isotherm and a mass transfer coefficient $k_{eff} = 1 \cdot 10^{-2}$ m s^{-1} for all components, see Figure 4.8. Even though the mass transfer coefficient is set to a high value, the elution peak for the film transfer model is seen at an earlier time than that for the LRM. The highest peak value during LRMFT appears at 11.31 minutes and a salt concentration of 261.28 mol m^{-3}, and measures 689.02 mol m^{-3} when scaled. The highest peak value during LRM appears at 12.43 minutes and a salt concentration of 389.47 mol m^{-3}, and measures 742.54 mol m^{-3} when scaled. This means that the target elutes at a salt concentration 33 % less during LRMFT compared to LRM, while the peak only decreases 6 %. This is questionable, since a high mass transfer coefficient would cause the intrapraticulate concentration to be very close to the bulk concentration, suggesting that a similar elution is to be expected.

If the bulk concentration is compared to the intraparticulate concentration, in the last finite volume of the column, for a mass transfer coefficient $k_{eff} = 1 \cdot 10^{-2}$ m s^{-1}, no difference is observed, see Figure 4.9. Therefore, the earlier elution can not be explained by a difference between bulk concentration and intraparticulate

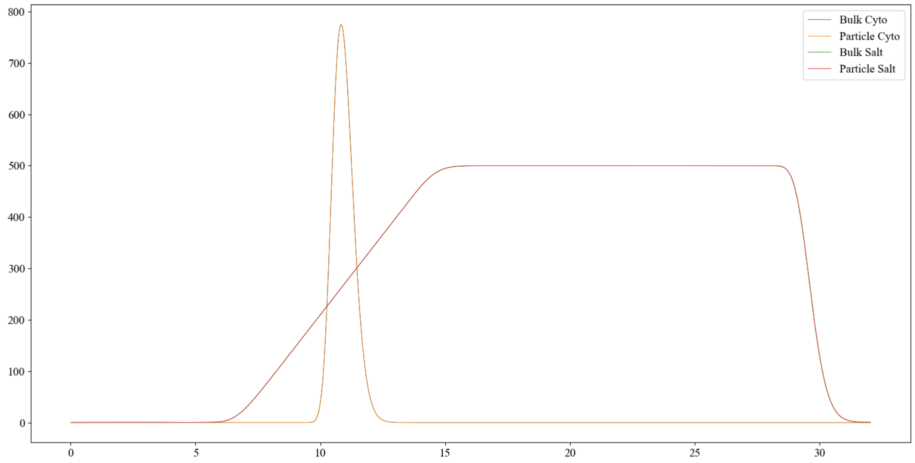
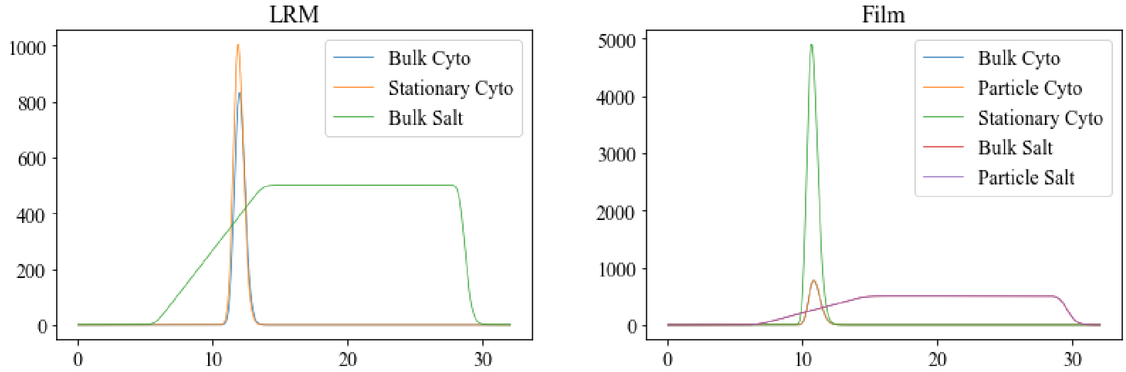


Figure 4.9 – Bulk concentration and intraparticulate concentration with mass transfer coefficient $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$, in the last finite volume of the simulated IEX column. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

concentration, neither for salt nor Cyto.

If instead the bulk concentration, the intraparticulate concentration, and the amount of adsorbed Cyto during a LRMFT modeling approach in the last finite volume of the column are compared to the bulk concentration, and the amount of adsorbed Cyto during a LRM modeling appproach in the last finite volume of the column, a difference in adsorbed amount of Cyto is observed, see Figure 4.10. Here, the amount of adsorbed Cyto appears to be five times greater during a LRMFT modeling approach compared to a LRM modeling approach. This explains the earlier elution during a LRMFT modeling approach compared to a LRM modeling approach, if one considers Equation 3.1. A larger amount of adsorbed Cyto will cause the expression to change sign, i.e. desorption of the protein, at a lower salt concentration. As for the reason to why a five times greater adsorption is seen during a LRMFT modeling approach, no explanation could be found within the time frame of this master thesis. An earlier hypothesis of incorrect particle porosity or column void was tested, but this was found not to be the problem. Another hypothesis was the possibility of any of the parameters being defined per volume, causing a mismatch in units, resulting in calculation errors. However, no such mismatch could be identified within the given time frame.



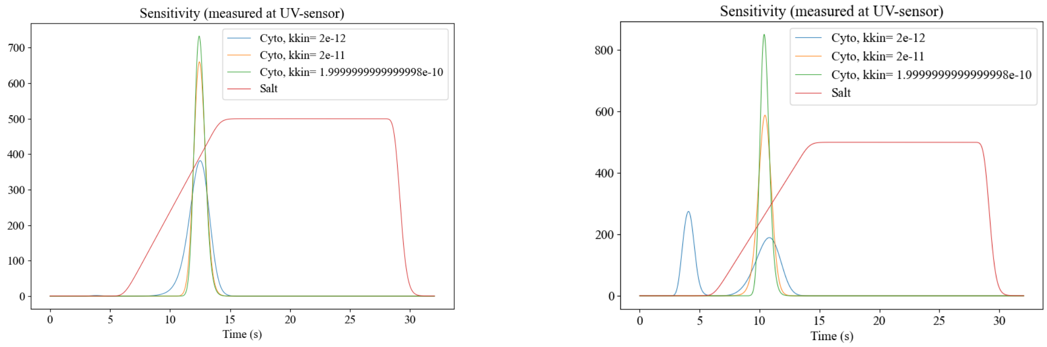
(a) The bulk concentration of salt and Cyto, and the amount of adsorbed Cyto during a LRM modeling approach in the last finite volume of the column.

(b) The bulk concentration of salt and Cyto, the intraparticulate concentration of salt and Cyto, and the amount of adsorbed Cyto during a LRMFT modeling approach in the last finite volume of the column.

Figure 4.10 – Comparison of the bulk concentrations, the intraparticulate concentration, and the amount of adsorbed Cyto during a LRMFT modeling approach and a LRM modeling approach. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

The results for the sensitivity analysis conducted for the parameter k_{kin} in a simplified SMA adsorption isotherm for a LRM modeling approach and a LRMFT modeling approach with a mass transfer coefficient of $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$ can be seen in Figure 4.11. The results of the sensitivity analysis would suggest that the LRMFT is more sensitive to changes in reaction rate. This is as expected, since the LRMFT overestimates the amount of adsorption, thus amplifying any changes made to k_{kin} in the adsorption description.

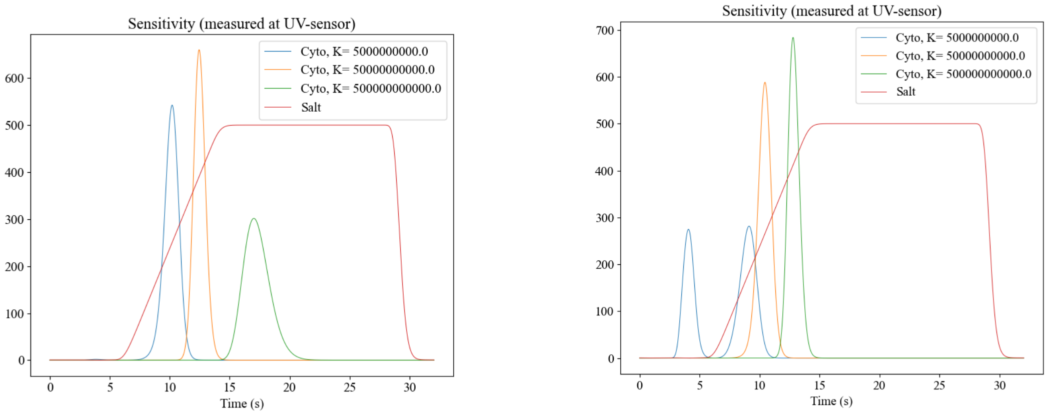
The results for the sensitivity analysis conducted for the parameter K in a simplified SMA adsorption isotherm for a LRM modeling approach and a LRMFT modeling approach with a mass transfer coefficient of $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$ can be seen in Figure 4.12. Due to the elution peak occurring at a lower salt concentration, two completely different behaviors are observed in the two modeling approaches, as changes are made to K. The general behavior for the LRMFT is an earlier elution, for all values of K when compared to the LRM. This is probably due to the adsorption being overestimated in the LRMFT. Because of this, it is difficult to make a fair comparison between the two modeling approaches as the results from the LRMFT builds on a faulty model. The sensitivity analysis does, however, further demonstrate the effects of the overestimated adsorption in the LRMFT.



(a) LRM modeling approach.

(b) LRMFT modeling approach

Figure 4.11 – Sensitivity analysis for the parameter k_{kin} in a simplified SMA adsorption isotherm for a LRM modeling approach and a LRMFT modeling approach with a mass transfer coefficient of $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

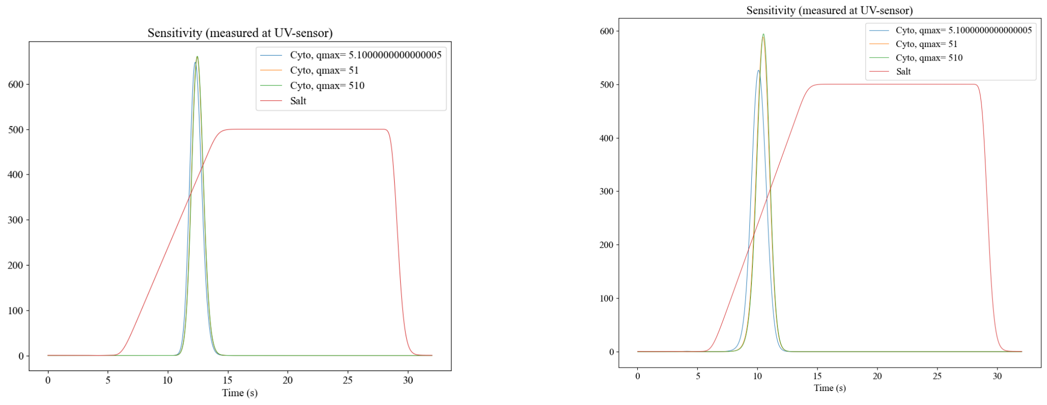


(a) LRM modeling approach.

(b) LRMFT modeling approach

Figure 4.12 – Sensitivity analysis for the parameter K in a simplified SMA adsorption isotherm for a LRM modeling approach and a LRMFT modeling approach with a mass transfer coefficient of $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

The results for the sensitivity analysis conducted for the parameter q_{max} in a simplified SMA adsorption isotherm for a LRM modeling approach and a LRMFT modeling approach with a mass transfer coefficient of $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$ can be seen in Figure 4.12. The LRM shows very little sensitivity towards changes of q_{max}, whereas the LRMFT shows greater sensitivity. The greater sensitivity in the LRMFT, however, is, as mentioned before, most likely due to an overestimated adsorption. The insensitivity is due to the maximum loading capacity of the column, never being reached in the fake experiment Cyto.



(a) LRM modeling approach.

(b) LRMFT modeling approach

Figure 4.13 – Sensitivity analysis for the parameter q_{max} in a simplified SMA adsorption isotherm for a LRM modeling approach and a LRMFT modeling approach with a mass transfer coefficient of $k_{eff} = 1 \cdot 10^{-2}\ m\ s^{-1}$. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Cyto is scaled by $1 \cdot 10^{4}$ to be able to show the elution peak and salt gradient in the same figure.

4.3 Simulating membranes

It was not possible to simulate membranes within the Orbit-simulator, due to every stream connected to the membrane being interpreted by the simulator, as of now, as outgoing streams. This disrupted the flowpath and no simulation could be made. Attempts were made to identify why this was the case, but no solution was found within the time frame of this master thesis. The implemented simple membrane model could therefore not be tested, thus no progression towards more complex models, like the gel model, could be made. Membrane simulation within the Orbit-simulator therefore still remains an interesting development potential in any future work with the simulator.

4.4 Simulation of a process with two columns

It was possible to simulate a process with two chromatography columns, see Figure 4.14. The proteins are first separated somewhat in the IEX column and then even more separated in the SEC column, resulting in a quite pure Zyto product, containing almost only salt and Zyto. The last salt peaks are due to washing of the IEX column, flushing out the remaining salt, originating from the end of the gradient phase. A similar rise in salt concentrations can be seen at the end of the washing of the SEC column. This salt originates from how the pooling was chosen to be made, more precisely when it was ended. In this experiment, the pooling was ended at the end of the gradient phase, allowing for more salt to be recycled to the SEC column. This was a conscious choice, as it first caused the Cyto peak to be flushed out and sent to waste, before the pooling from the IEX-column was turned off and the Zyto peak lead to the sensors.

As can be seen in Figure 4.14, the gradient phase starts a little bit after five minutes, even though the three predescending phases are set to run for only three minutes. This is most probably due to the volume of the columns, causing a delay to occur before a signal is read at the sensors. The flow rate was set to 0.5 mL/min and the column volume to 1 mL, which would result in a delay of 2 minutes. The delay observed in Figure 4.14 is, however, a little bit greater than 2 minutes. The majority of this delay is therefore explained by the volume of the columns and the flow rate, while the remaining delay is most probably due to the lengths of the tubes.

The second Zyto peak also appears somewhat asymmetrical, a behavior not seen when simulating a lone SEC column, see Figure 4.3. This is most probably due to how the pooling cuts are made. Some of the Zyto protein resulting in the second peak will be flushed to waste, before versatile valve 2 is redirected and instead sending the sample to the sensors.

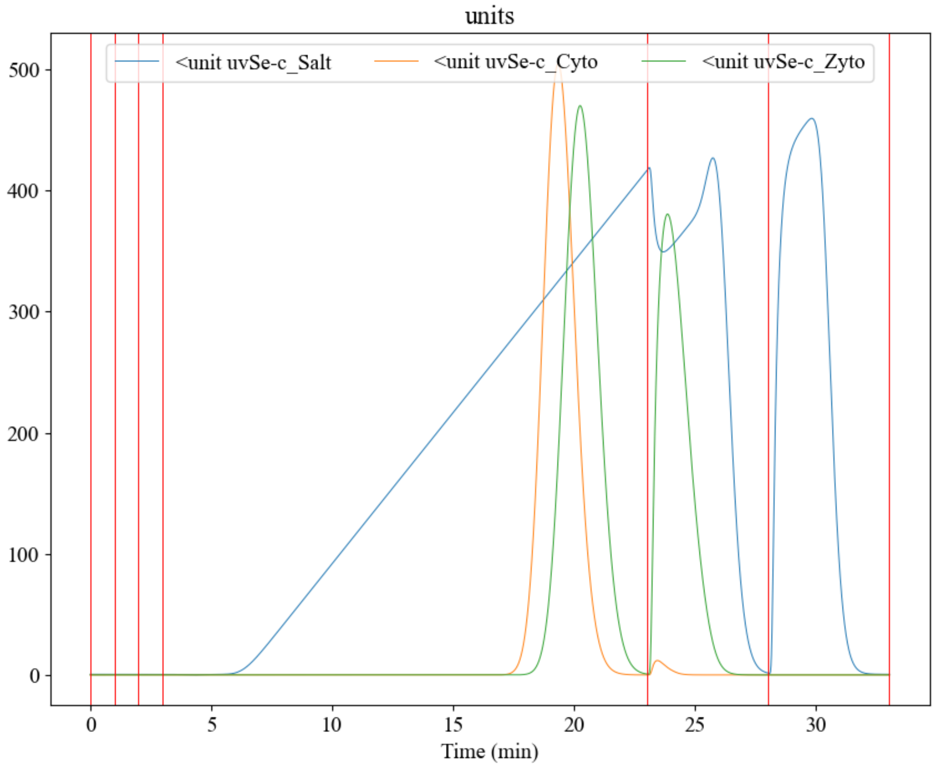


Figure 4.14 – Chromatogram for the fake experiment with two columns. The vertical red lines indicate the start and stop of the different phases. Phases from the left; flush1, inject, Wash11, Elution1, Wash2, and Wash12. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Cyto and Zyto is scaled by $1 \cdot 10^4$ to be able to show the elution peaks and salt gradient in the same figure.

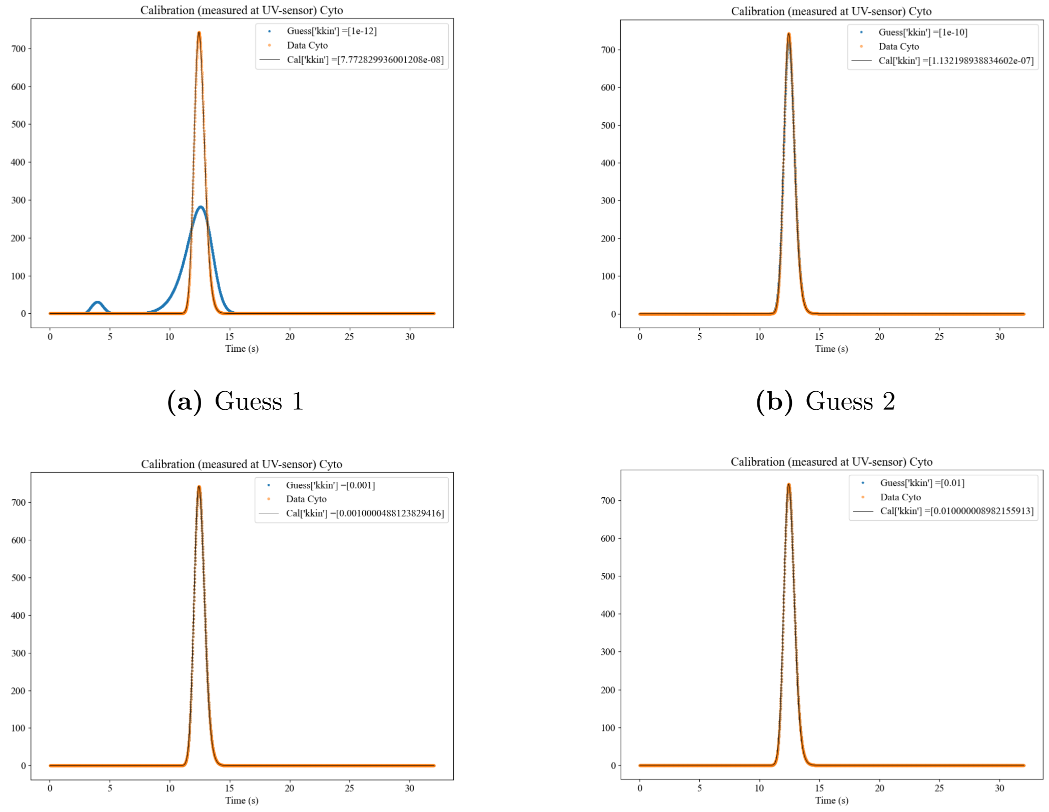
4.5 Parameter estimation

In this section, the results of the parameter estimations are presented.

4.5.1 Parameter estimation in a single chromatography column

The results of the lone parameter estimation of k_{kin} can be seen in Figure 4.15. The calibration method is capable of estimating a value of k_{kin} resulting in a similar peak as the value used for data generation. However, the estimated value is not the same as the parameter value used for data generation, $k_{kin} = 1 \cdot 10^{-4}\ m\ s^{-1}$. This is due to a insensitivity to changes of k_{kin} in the simplified SMA adsorption description when close to the parameter value used for data generation. Due to the k_{kin} used for data generation being set to an already high value, resulting in a fast adsorption, no major change in the results will be seen once the estimated parameter value approaches the value used for data generation. The adsorption is not being limited by k_{kin}, at these values, and will therefore approach a maximum limit. The curve_fit function will therefore have a difficult time in estimating a value closer to the value used for data generation, since the residuals will be very close to zero. Nonetheless, the calibration is able to estimate the parameter good enough for it to provide accurate results. For some applications, this might be sufficient, as long as the process designer is aware of the limitations of the estimated parameters, and the errors and uncertainties this might introduce to the simulations.

The results of the lone parameter estimation of K can be seen in Figure 4.16. Here, the simple calibration method manages to achieve an accurate estimation of the parameter for some guesses. Only for Guess 4 is an accurate parameter estimation not found. Olsson stated in his master thesis, that if a too extreme guess, either too low or too high, of the value of K is made, the adsorption will either cause the component to be flushed straight through the column or not to be eluted at all. This will make it impossible for the minimization method to find a minimum to the objective function since there is no sensitivity in regards to changes in K. (Olsson 2019) This is however only true for an objective function considering, in some way, retention time. When only considering concentration in the objective function, very little sensitivity is to be had if the elution peaks of the guessed parameter value and the data do not overlap. During no overlap, the sum of squares will be very similar for each step taken by the algorithm, causing insensitivity. This could be the case in



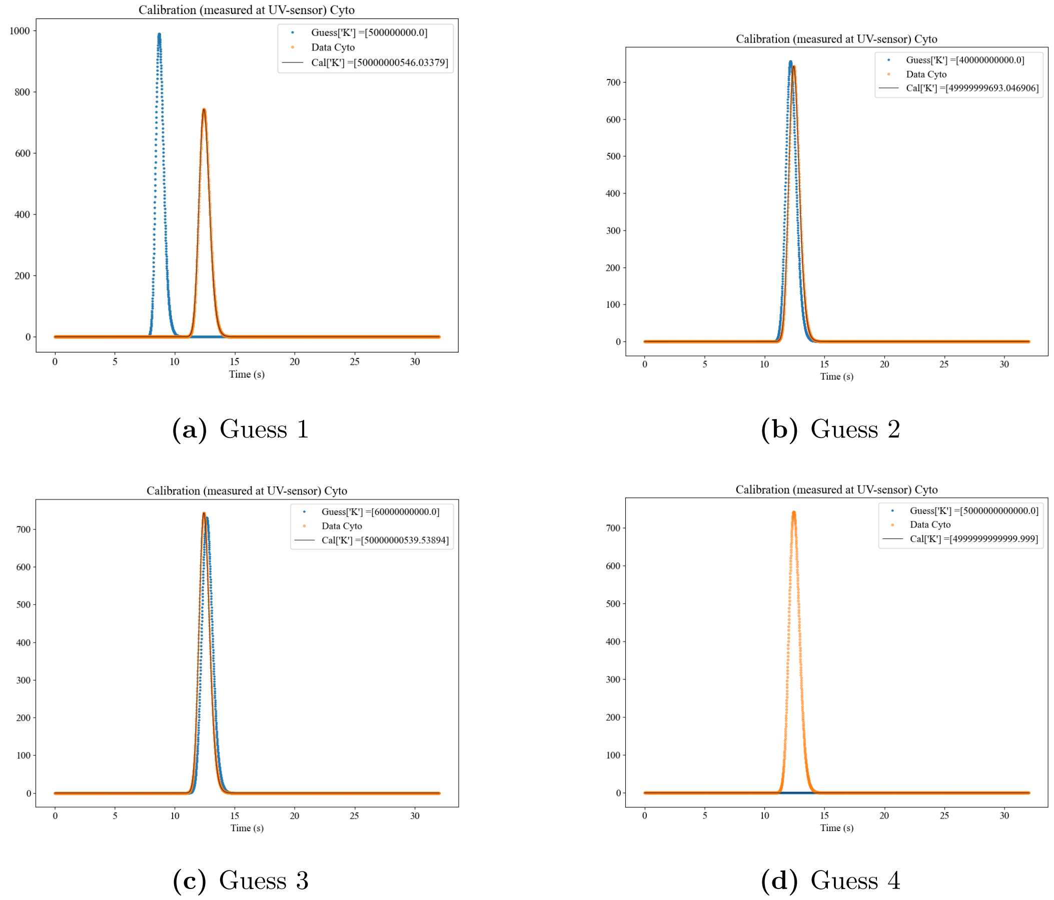
(a) Guess 1

(b) Guess 2

(c) Guess 3

(d) Guess 4

Figure 4.15 – Lone parameter estimation of k_{kin} for a simplified SMA adsorption description in the fake experiment Cyto. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.



(a) Guess 1

(b) Guess 2

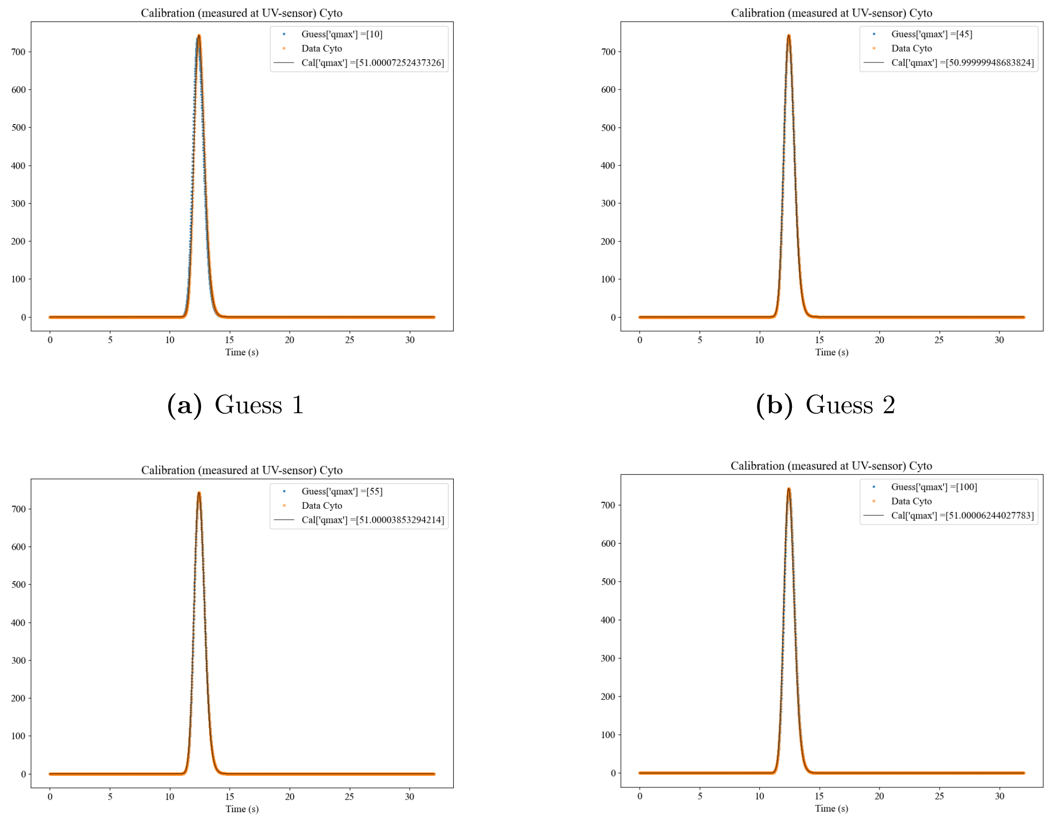
(c) Guess 3

(d) Guess 4

Figure 4.16 – Lone parameter estimation of K for a simplified SMA adsorption description in the fake experiment Cyto. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

Figure 4.16d, since the guessed value results in no elution peak. When provided with a guess close to the actual parameter value, however, the simple calibration method manages to accurately estimate the parameter, even without apparent overlap, see Figures 4.16a-4.16c. This is remarkable, but most probably a result of the adjusted, low tolerance in the curve_fit-function and the chosen step-length. A difference, be it small, in the sum of squares will thus be seen for the adjustments of the parameter made by the algorithm, leading to sensitivity.

The results of the lone parameter estimation of q_{max} can be seen in Figure 4.16. Here, the simple calibration method is capable of achieving an accurate estimation of the parameter for all guesses. This is remarkable, since in the sensitivity analysis, the LRM modeling approach showed very little sensitivity towards changes in q_{max}.



(a) Guess 1

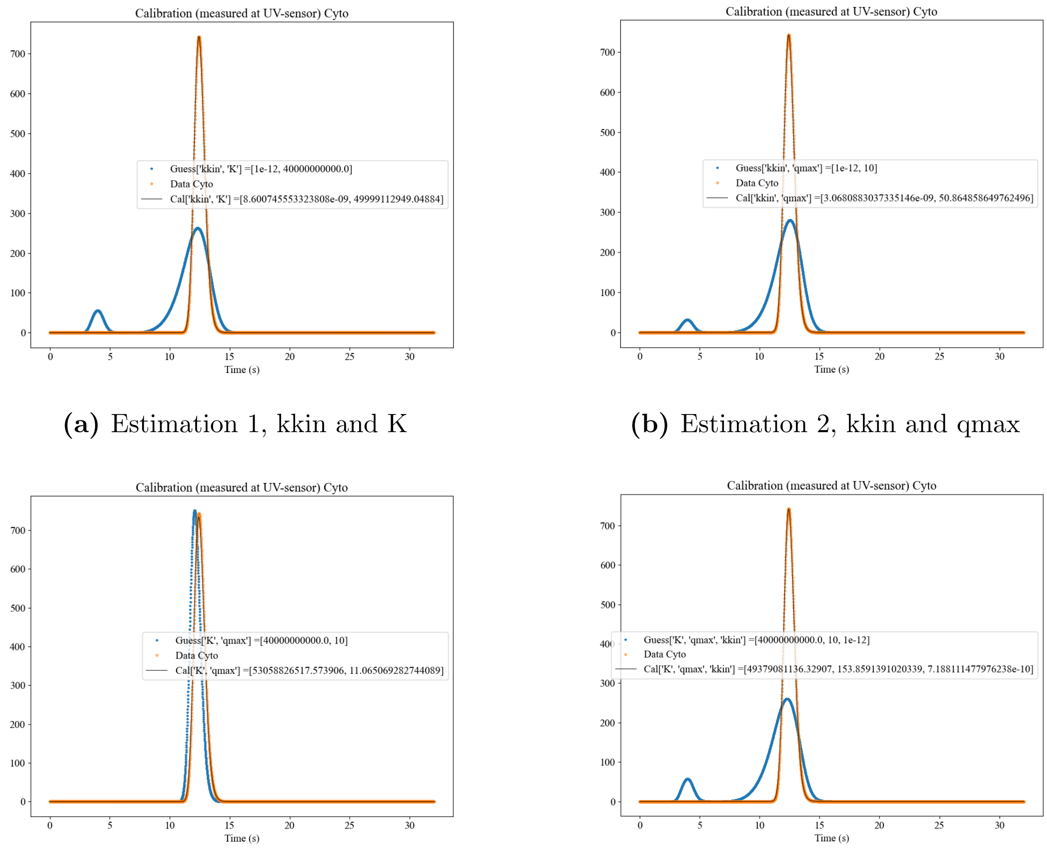
(b) Guess 2

(c) Guess 3

(d) Guess 4

Figure 4.17 – Lone parameter estimation of q_{max} for a simplified SMA adsorption description in the fake experiment Cyto. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

One could therefore expect it to be difficult for curve_fit to estimate the parameters since very little change to the results is seen when adjusting q_{max}. However, by adjusting the step length and the desired relative error in curve_fit, and with the scaled parameters, it is possible that this aids curve_fit in the parameter estimation of q_{max}, either by having a very low error tolerance or by making the step length finer reducing the risk of missing the solution.



(a) Estimation 1, kkin and K

(b) Estimation 2, kkin and qmax

(c) Estimation 3, K and qmax

(d) Estimation 4, all

Figure 4.18 – Parameter estimation of k_{kin}, K, and q_{max} for a simplified SMA adsorption description in the fake experiment Cyto. The unit for the y-axis is *mol* m^{-3} but the signal for Cyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

The results of the multiple parameter estimation for a single chromatography column can be seen in Figure 4.18. The calibration method is capable of estimating the parameters for an accurate solution to be achieved, however the estimated parameter values are different from the values used for data generation. For the estimations containing k_{kin}, i.e. Estimation 1, 2, and 4, this is most probably due to the same reason as discussed earlier, the value of k_{kin} being set to a high value for the data generation. However, for Estimation 4 the number of estimated parameters could also be a contributing factor to the inaccuracy of the estimated parameters. Schmidt-Traub et.al. are in *Preperative chromatography* recommending not to estimate more than two to three parameters as this increases the risk of getting trapped in a local minima or finding more than one set of parameters resulting in an accurate solution

(Schmidt-Traub, Schulte, and Seidel-Morgenstern 2020). Estimation 4 is a prime example of the latter, since an accurate solution is found but the parameters are incorrectly estimated. For a calibration to experimental data, it would be impossible, as long as the parameters hold physically or chemically reasonable values, to know if the parameters are incorrect if an accurate solution is obtained. By using a simulated data set, with known parameters, this problem is identified, making it clear that an increased robustness of the calibration method or a decreased number of parameters to be estimated is needed.

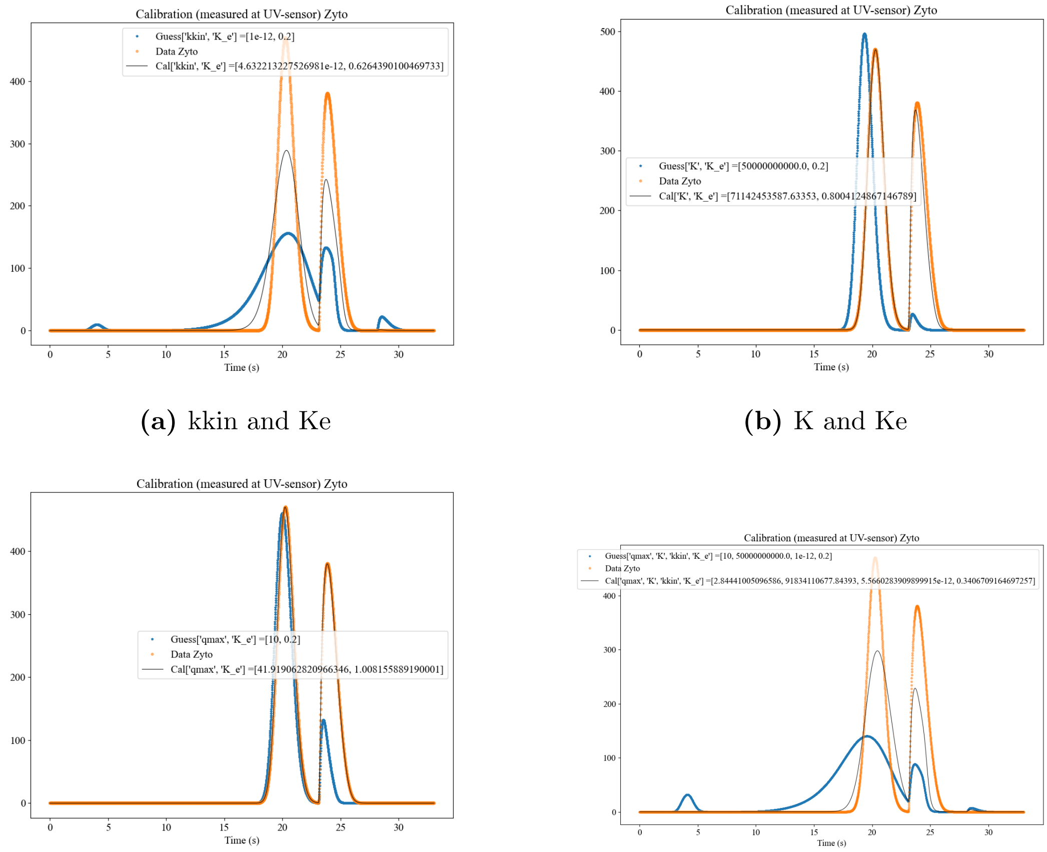
From the results presented above it is clear that the robustness of the calibration method has to be increased to allow for a reliable parameter estimation. Some tests in scaling the parameters by the tenth logarithm also during the multiple parameter estimation were made, but were only found to cause calculation errors within the simulator. It could, however, be possible that a thought-through scaling of the parameters could increase the reliability of the calibration method.

Olsson presented in his master thesis a robust method for the estimation of parameters in the simplified SMA adsorption description, for the in-built simulator in Orbit. This method contains six different steps in the estimation of the parameters, where curve_fit is used only as a ”polishing” step. The calculation time for this method was estimated to be between 2-4 hours. (Olsson 2019) This is to compare with a calculation time of a couple of minutes to approximately half an hour for the simple calibration method presented in this work. The focus of this work was not to develop a robust calibration method for automated parameter estimation, but rather to extend the amount of models present in the simulator. Thus, the short calculation time of the simple calibration method developed in this work is desirable, since it is still able to estimate the parameters if a close guess is made. If the parameters are completely unknown, i.e. an experimental data set is used, the sensitivity-method can be used to determine a guess resulting in a peak close to the experimental data, and then a calibration can be run. For any future development of the simulator, it is however recommended to increase the robustness of the calibration method if an automated parameter estimation is pursued.

4.5.2 Parameter estimation in multiple chromatography columns

The results from the parameter estimation in multiple chromatography columns can be seen in Figure 4.19. The simple calibration method presented in this work is capable of estimating the parameters in two different chromatography columns if the guessed parameter values are close to the actual value and if the amount of estimated parameters are kept to a minimum in each column. This is as expected since the lack of robustness in the calibration method remains also for the parameter estimation in multiple chromatography columns, since no changes to the calibration method has been made. As the amount of estimated parameters is increased, the calibration methods capability to accurately estimate the parameters is lost, as suggested by Schmidt-Traub et.al (Schmidt-Traub, Schulte, and Seidel-Morgenstern 2020). The same holds true for guesses far from the actual parameter value, see Figure 4.19a, but this time the accuracy is most likely lost due to a lack of robustness. However, the parameter estimation conducted in this work shows that it is possible to, within the in-built simulator in Orbit, estimate parameters in multiple columns. If the robustness is increased this would present a powerful tool for process design and evaluation. This is, however, beyond the limits of this master thesis and is only proposed as a future improvement of the simulator.

When estimating parameters in multiple chromatography columns, the calculation time is greatly increased compared to the single chromatography setup. It is, however, still a matter of minutes. If the robustness is increased, very long calculation times should therefore be expected, if a parameter estimation in multiple columns is made. For an automated parameter estimation this might be acceptable, otherwise a transition to a faster programming language would be beneficial or the parameter estimation could of course be run on a more powerful computer.



(a) kkin and Ke

(b) K and Ke

(c) qmax and Ke

(d) All

Figure 4.19 – Parameter estimation of k_{kin}, K, q_{max}, and K_e for a simplified SMA adsorption description and a SEC model in the fake experiment with two columns. The unit for the y-axis is $mol\ m^{-3}$ but the signal for Zyto is scaled by $1 \cdot 10^4$ to be able to show the elution peak and salt gradient in the same figure.

5 Future Work

In this section future work prospects, identified throughout the work on this master thesis, are proposed.

5.1 Communication with CADET

CADET is a highly developed, open source chromatography simulator based on C++ but with a python front-end (Leweke and Lieres 2018). This would make it possible and potentially quite easy to integrate this simulator into Orbit, by letting Orbit send design parameters to CADET allowing for CADET to simulate the column and return the results to Orbit, for further calculations. By doing this the entire library of models and the state of art numerical solvers used in CADET, would possibly be able to capture the behavior of the columns more accurately and more efficiently than Orbit, while preserving the capability presented by Orbit to automatically generate lab setup simulations.

5.2 Enabling for simulation of membranes

One of the aims of this work was to implement a model for UFDF into the simulator, but it was never reached due to how the simulator interprets the streams connected to the membrane. This problem must first be solved before any simulation of membranes can take place, and integrating the capability to simulate membranes is therefore proposed as a future work.

5.3 Improving the calibration method

The simple calibration method presented in this work is capable of estimating parameters as long as the guessed parameter values are close to the actual value. This is no problem if the actual parameters are known, but if a calibration to experimental data is to be made the robustness of the calibration method has to be increased. ”Bad” guesses could therefore be handled, and an automated parameter estimation in multiple columns could possibly be achieved.

5.4 Experimental data

None of the models presented in this work could be validated against experimental data, since no experimental data was available. Therefore, experimental data should be obtained to enable for a validation of the models to be made.

6 Conclusions

An IEX-MPM, a HIC, and a pH-dependent AC adsorption description were implemented to the Orbit-simulator making it now possible to simulate, not only IEX columns through a new isotherm, but also HIC columns and AC columns. A model for SEC was also implemented to the simulator. This proves that it is possible to extend the number of available adsorption descriptions and to simulate SEC in the simulator. This increases its usefulness and allows for even more down stream processes to be simulated. Experimental data should however be obtained, so that validation of the models can be made.

A LRMFT modeling approach was also added to the simulator, providing the possibility to choose another model complexity level while simulating. The modeling approach was found to overestimate the adsorption, however, and no reason to why could be found within the time limits of this master thesis. Further work is thus needed before an evaluation of the impact of increased model complexity can be made.

An attempt was also made to implement a simple model for UFDF, treating the membrane as a splitter, into the simulator, but without any success. The simulator interprets every stream connected to the membrane as an outgoing stream, disrupting the flow path, halting the simulation. No solution to this problem was found within the time frame of this master thesis, and therefore no progression towards a gel model could be made. Membranes will thus remain an interesting work prospect within the simulator.

A two column process, containing an IEX column and a SEC column, was also simulated with the Orbit-simulator. The elution peak from the IEX column was pooled and sent to a SEC column for further separation. The result was a mixture of target protein and salt, with minimal traces of contamination. This proves that it is possible to simulate processes with multiple chromatography columns with the Orbit-simulator.

A simple calibration method was also developed and a parameter estimation was conducted for both a single column process and a multiple column process. The calibration method lacked robustness, but was in some cases able to accurately estimate the parameters for a single component. The parameter estimation for the multiple column process suggests that it is possible to accurately estimate parameters for a single component in multiple columns at the same time, within the simulator.

To conclude, through the extended amount of models within the simulator, its usefulness is increased and a digitilized pharmaceutical industry is brought even closer.

Acronyms

AC Affinity Chromatography. iv, v, x, xi, 5, 24, 27, 28, 42, 47, 48, 66

DAE Differential Algebraic Equations. 21

HIC Hydrophobic Interaction Chromatography. iv, v, x, xi, 5, 24, 26, 27, 42, 47, 66

IEX Ion Exchange Chromatography. iv, v, x, xi, 4, 17, 22, 24, 26, 32–36, 39, 41–44, 49, 54, 66

LRM Lumped Rate Model. x, xi, 8, 10, 25, 28, 29, 31, 32, 42, 44, 45, 47–53

LRMFT Lumped Rate Model with Film Transfer. x, xi, 10, 25, 28, 29, 31, 42, 45, 46, 49–53, 66, 70

MPM Mobile Phase Modulator. x, xi, 24, 26, 44, 66

NFF Normal Flow Filtration. 10

ODE Ordinary Differential Equations. 21, 37

PDE Partial Differential Equations. 21

SEC Size Exclusion Chromatography. iv, v, x, xi, 3, 24, 28, 29, 32–36, 41, 42, 44–46, 54, 63, 66

SMA Steric Mass Action. xi, 22, 24–26, 31–33, 39, 41, 43, 44, 49, 51–53, 56–61, 63

TFF Tangential Flow Filtration. x, 10, 11, 29

TMP Transmembrane pressure. 11, 13

UFDF Ultrafiltration Diafiltration. iv, v, x, 10–12, 30, 66

List of Symbols

β -	Characteristic charge.	[-]
c_0 -	Concentration before UF.	$[\frac{mol}{m^3}]$
c_1 -	Concentration after UF.	$[\frac{mol}{m^3}]$
c_i -	Concentration of component i.	$[\frac{mol}{m^3}]$
c_{IgG} -	Concentration of IgG.	$[\frac{mol}{m^3}]$
c_g -	Gel point concentration.	$[\frac{mol}{m^3}]$
$c_{p,i}$ -	Intraparticulate concentration of component i.	$[\frac{mol}{m^3}]$
$D_{ax,i}$ -	Axial dispersion of component i.	$[\frac{m^2}{s}]$
d_p -	Membrane pore diameter.	[m]
Δx -	Length of membrane channel.	[m]
ϵ -	Membrane surface porosity.	[-]
ϵ_a -	Total porosity.	[-]
ϵ_c -	Column void.	[-]
ϵ_p -	Packing porosity.	[-]
γ_i -	Characteristic hydrophobicity coefficient of component i.	$[\frac{m^3}{mol}]$
J -	Membrane flux.	$[\frac{m}{s}]$
K_i -	Equilibrium Henry coefficient of component i.	$[\frac{mol}{m^3}]$

$K_{0,i}$ -	Modified equilibrium coefficient of component i.	$[\frac{mol}{m^3}]$
$K_{0,IgG}$ -	Modified equilibrium coefficient in the pH-dependent AC adsorption description.	$[\frac{mol}{m^3}]$
$K_{0M,i}$ -	Modified equilibrium coefficient of component i in the HIC adsorption description.	$[\frac{m^3}{mol}]$
K_e -	Exclusion factor.	[-]
k -	Mass transfer coefficient in the gel model.	$[\frac{m}{s}]$
k_{eff} -	Mass transfer coefficient in the LRMFT.	$[\frac{m}{s}]$
k_f -	Internal mass transfer coefficient.	$[\frac{1}{s}]$
k_{kin} -	Kinetic coefficient.	$[\frac{m^3}{mol \cdot s}]$
$k_{kinM,i}$ -	Kinetic coefficient of component i in the HIC adsorption description.	$[\frac{1}{s}]$
n -	Exponent parameter for the pH dependency.	[-]
ν_i -	Characteristic charge of component i.	[-]
μ -	Dynamic viscosity.	$[\frac{N \cdot s}{m^2}]$
pH_{ref} -	Reference pH coefficient.	[-]
q_i -	Adsorbed amount of component i.	$[\frac{mol}{m^3}]$
q_{IgG} -	Adsorbed amount of IgG.	$[\frac{mol}{m^3}]$
q_{max} -	Maximum loading capacity.	$[\frac{mol}{m^3}]$
$r_{ads,i}$ -	Adsorption rate for component i.	$[\frac{mol}{m^3 \cdot s}]$
R_G -	Retention factor increase due to polarization and fouling.	[-]
$r_{HIC,i}$ -	Adsorption rate of component i according to the HIC adsorption description.	$[\frac{mol}{m^3 \cdot s}]$

$r_{IgG,f}$ -	Adsorption rate of IgG.	$[\frac{mol}{m^3 \cdot s}]$
R_M -	Retention factor.	[-]
$r_{MPM,i}$ -	Adsorption rate of component i according to the MPM adsorption description.	$[\frac{mol}{m^3 \cdot s}]$
r_p -	Particle radius.	$[m]$
$r_{SIM,i}$ -	Adsorption rate of component i according to the simplified SMA adsorption description.	$[\frac{mol}{m^3 \cdot s}]$
s -	Salt concentration.	$[\frac{mol}{m^3}]$
t -	Time.	$[s]$
TMP -	Transmembrane pressure.	$[Pa]$
V_0 -	Volume before UF.	$[m^3]$
v_{sup} -	Superstitial velocity.	$[\frac{m}{s}]$
V_{UF} -	Volume after UF.	$[m^3]$
z -	Position.	$[m]$

References

Andersson, Niklas (2018). *The Orbit Controller.*

AspenTech (2022). *Simulate Chromatography While Improving Process Design.* URL: https://www.aspentech.com/en/products/pages/aspen-chromatography (visited on 01/24/2022).

Bellot, JC and JS Condoret (1993). “Modelling of liquid chromatography equilibria”. In: *Process Biochemistry* 28.6, pp. 365–376.

Broadus, Calvin Cordozar Jr (2018). *Hollywood Walk of Fame acceptance speech.* Acceptance speech by Calvin Cordozar Broadus Jr, also known as Snoop Dogg, at Hollywood Walk of Fame. URL: https://www.youtube.com/watch?v=GvpZ2mcUJWc (visited on 02/18/2022).

Challener, Cynthia A (2018). “Evolving UF/DF capabilities”. In: *BioPharm International* 31, pp. 24–27.

Cheryan, Munir (1998). *Ultrafiltration and microfiltration handbook.* CRC press.

Cytiva (2020). *Cytiva and GoSilico agree to co-market ChromX mechanistic modeling software.* URL: https://www.cytivalifesciences.com/en/us/news-center/ge-healthcare-life-sciences-and-gosilico-enter-comarketing-agreement-10001 (visited on 01/24/2022).

— (2021). *ÄKTA™ pure - Chromatography systems.* URL: https://cdn.cytivalifesciences.com/dmm3bwsv3/AssetStream.aspx?mediaformatid=10061&destinationid=10016&assetid=16276.

Gomis-Fons, Joaquín et al. (2019). “Integration of a complete downstream process for the automated lab-scale production of a recombinant protein”. In: *Journal of Biotechnology* 301, pp. 45–51. ISSN: 0168-1656. DOI: https://doi.org/10.1016/j.jbiotec.2019.05.013. URL: https://www.sciencedirect.com/science/article/pii/S0168165619301774.

GoSilico (2021). *Fluid dynamics - Fluid flow effects in a chromatography column.* URL: https://gosilico.com/technology/fluid-dynamics/ (visited on 12/01/2021).

Gu, Tingyue (2015). “Literature Review”. In: *Mathematical Modeling and Scale-Up of Liquid Chromatography: With Application Examples.* Cham: Springer International Publishing, pp. 7–14. ISBN: 978-3-319-16145-7. DOI: 10.1007/978-3-319-16145-7_2. URL: https://doi.org/10.1007/978-3-319-16145-7_2.

— (2022). *Chromulator-IEX.* URL: https://people.ohio.edu/gu/CHROM/IEX/Chromulator-IEX.html (visited on 01/24/2022).

Hage, David S. and Ryan Matsuda (2015). “Affinity Chromatography: A Historical Perspective”. In: *Affinity Chromatography: Methods and Protocols.* Ed. by Senta Reichelt. New York, NY: Springer New York, pp. 1–19. ISBN: 978-1-4939-2447-9.

DOI: 10.1007/978-1-4939-2447-9_1. URL: https://doi.org/10.1007/978-1-4939-2447-9_1.

Hahn, T, V Heuveline, and J Hubbuch (2012). "ChromX–A Powerful and User-Friendly Software Package for Modeling Liquid Chromatography Processes". In: *Chemie Ingenieur Technik* 84.8, pp. 1342–1342.

Izmaylov, A, A Saraev, and Z Barinova (2021). "The Development of the Domestic Pharmaceutical Industry in the Context of Digitalization". In: *Current Achievements, Challenges and Digital Chances of Knowledge Based Economy.* Springer, pp. 181–188.

Jungbauer, Alois and Rainer Hahn (2009). "Ion-exchange chromatography". In: *Methods in enzymology* 463, pp. 349–371.

Klutz, Stephan et al. (2015). "Developing the biofacility of the future based on continuous processing and single-use technology". In: *Journal of Biotechnology* 213, pp. 120–130.

Knox, John H. and Hugh P. Scott (1984). "Theoretical models for size-exclusion chromatography and calculation of pore size distribution from size-exclusion chromatography data". In: *Journal of Chromatography A* 316. Eight International Symposium on Column Liquid Chromatography, pp. 311–332. ISSN: 0021-9673. DOI: https://doi.org/10.1016/S0021-9673(00)96162-4. URL: https://www.sciencedirect.com/science/article/pii/S0021967300961624.

Leweke, Samuel and Eric von Lieres (2018). "Chromatography analysis and design toolkit (CADET)". In: *Computers & Chemical Engineering* 113, pp. 274–294.

Linde (2021). *Liquid chromatography.* Last accessed 01 December 2021. URL: http://hiq.linde-gas.com/en/analytical_methods/liquid_chromatography/index.html.

Lund University (2018). *Preperative Chromatography Simulation.* URL: https://www.control.lth.se/research-old/piclu-process-industrial-center/piclu-2008-2014/competence-development/industrial-courses/preparative-chromatography-simulation/ (visited on 01/24/2022).

"1 - Fundamentals" (2015). In: *Ultrafiltration for Bioprocessing.* Ed. by Herb Lutz. Woodhead Publisching, pp. 1–6. ISBN: 9781907568466. DOI: https://doi.org/10.1016/B978-1-907568-46-6.00001-X..

Ng, Candy KS et al. (2012). "Design of high productivity antibody capture by protein A chromatography using an integrated experimental and modeling approach". In: *Journal of Chromatography B* 899, pp. 116–126.

Nilsson, Bernt and Niklas Andersson (2017). "Simulation of process chromatography". In: *Preprative chromatography for separation of proteins. Wiley, New Jersey*, pp. 81–108.

Olsson, Johan (2019). "Automated Simulation and Parameter Estimation of Chromatography Processes". MA thesis. Chemical Engineering.

Queiroz, J.A., C.T. Tomaz, and J.M.S. Cabral (2001). "Hydrophobic interaction chromatography of proteins". In: *Journal of Biotechnology* 87.2, pp. 143–159. ISSN: 0168-1656. DOI: https://doi.org/10.1016/S0168-1656(01)

00237-1. URL: https://www.sciencedirect.com/science/article/pii/S0168165601002371.

Saxena, Arunima et al. (2009). “Membrane-based techniques for the separation and purification of proteins: An overview”. In: *Advances in colloid and interface science* 145.1-2, pp. 1–22.

Schmidt-Traub, Henner, Michael Schulte, and Andreas Seidel-Morgenstern (2020). *Preparative chromatography*. John Wiley & Sons.

The Scipy Community (2022a). *scipy.integrate.solve_ivp*. URL: https://docs.scipy.org/doc/scipy-1.8.0/html-scipyorg/reference/generated/scipy.integrate.solve_ivp.html?highlight=solve%5C%20ivp#scipy.integrate.solve_ivp (visited on 02/08/2022).

— (2022b). *scipy.optimize.curve_fit*. URL: https://docs.scipy.org/doc/scipy-1.8.0/html-scipyorg/reference/generated/scipy.optimize.curve_fit.html#scipy.optimize.curve_fit (visited on 02/08/2022).

ThermoFischer (2022). *Virtual IC Column Separation Simulator*. URL: https://www.thermofisher.com/se/en/home/industrial/chromatography/ion-chromatography-ic/ion-chromatography-columns/virtual-ic-column-separation-simulator.html (visited on 01/24/2022).

Ultrafiltration for Bioprocessing (2018). APE Workshop 2018 at Lund University.

Ypso-Facto (2022). *ChromWorks™, the chromatographic simulator made by users for users*. URL: https://www.ypsofacto.com/services-chemical-software-chromworks (visited on 01/24/2022).

A Appendix 1 - The Simulator

A.1 The LRMFT

```
        #------FILM MODEL------
        if item.model == 'Film':
            for ic in range(Nc):
                cs = item.c[0,:]
                cps = item.c[Nc,:] # Modifier concentration in particle
                c = item.c[ic,:]
                cp = item.c[Nc+ic,:]
                q = item.c[Nc+Nc+ic,:]

                if item.columnType in components[ic].ads:

                    if item.columnType == "SEC":
                        epsp = item.porosity*components[ic].ads["SEC"]["K_e"]
                        eps=epsc+(1-epsc)*epsp#epsilon
                        absorption = 0

                    elif item.columnType == "IEX_SMA":
                            abs1= cp* (1-sumq) * np.abs(1-sumq)**(components[ic].ads["IEX_SMA"]["nu"]-1)
                            abs2= q * cps * np.abs(cps)**(components[ic].ads["IEX_SMA"]["nu"]-1)
                            absorption = components[ic].ads["IEX_SMA"]["kkin"]*(components[ic].ads["IEX_SMA"]["K"]*abs1-abs2)

                    elif item.columnType == "IEX_MPM":
                        abs1 = components[ic].ads["IEX_MPM"]["qmax"]*cp*(1-q/components[ic].ads["IEX_MPM"]["qmax"])
                        abs2 = q*cps*np.abs(cps)**(components[ic].ads[col_type]["beta"]-1)
                        absorption = components[ic].ads["IEX_MPM"]["kkin"]*(components[ic].ads["IEX_MPM"]["H_0"]*abs1-abs2)


                    elif item.columnType == "HIC":
                        absorption = components[ic].ads["HIC"]["kkin"]*(components[ic].ads["HIC"]["K_0"]*np.exp(components[ic].ads["HIC"]["gamma"]*cps)*components[ic].ads["HIC"]["qmax"]*cp*(1-sumq)-q)

                    elif item.columnType == "AC":
                            pH = -np.log10(cps)
                            abs1 = components[ic].ads["AC"]["K_0"]*(pH/components[ic].ads["AC"]["pH_ref"])**components[ic].ads["AC"]["n"]*components[ic].ads["AC"]["qmax"]*cp
                            abs2 = (1 + components[ic].ads["AC"]["K_0"]*(pH/components[ic].ads["AC"]["pH_ref"])**components[ic].ads["AC"]["n"]*cp)
                            absorption = components[ic].ads["AC"]["k_f"]*(abs1/abs2 - q)

                else: # Default column model, i.e. simplified IEX-SMA
                    abs1= cp* (1-sumq) * np.abs(1-sumq)**(components[ic].ads["Default"]["beta"]-1)
                    abs2= q * cps * np.abs(cps)**(components[ic].ads["Default"]["beta"]-1)
                    absorption = components[ic].ads["Default"]["kkin"]*(components[ic].ads["Default"]["H_0"]*abs1-abs2)

                if np.any((1-sumq)<0.):
                    bug

                dispersion = A2tot.dot(item.c[ic,:]) + B2tot*cin[0][ic]
                convection = v/epsc*(A1tot.dot(item.c[ic,:]) + B1tot*cin[0][ic])
                film_transfer = components[ic].k_eff*3/(item.diameterPar/2)*(c-cp)
                sumOfAll = dispersion-convection-(1-epsc)/(epsc)*film_transfer
                dc[ic,:] = sumOfAll
                if item.columnType == "SEC" and epsp == 0:
                    sumOfAll = dispersion-convection
                    dc[ic,:] = sumOfAll
```

```
                        dc[Nc+ic,:] = 0
                    else:
                        dc[Nc+ic,:] = film_transfer/epsp - (1-epsp)/epsp * absorption
                    dc[Nc+Nc+ic,:] = absorption
                dc = dc.reshape((N*(Nc+Nc+Nc),))
                #-----END FILM MODEL-----
```

A.2 The getParameter- and setParameter methods

```
def getParameter(self, param, col_type):
        if param in self.ads[col_type]:
            return self.ads[col_type][param]

        elif param == "k_eff":
            return self.k_eff

        else:
            raise Exception("There is no parameter called "+ param +" in the adsorption dictionary. Check column model.")

    def setParameter(self, col_type, param, value):
        if param in self.ads[col_type]:
            self.ads[col_type][param] = value

        elif param == "k_eff":
            self.k_eff = value

        else:
            raise Exception("There is no parameter called " + param +" in the adsorption dictionary. Check column model.")
```

A.3 The sensitivity method

```
def sensitivity(self, instructions, parameter, comp_name, column, factor):

        fig2 = plt.figure(2)
        plt.xlabel('Time (s)')

        for comp in self.S._sim['compObjects']:
            if comp.name == comp_name:
                component = comp
                index = self.S._sim['compObjects'].index(comp)


        org = component.getParameter(parameter, column.columnType)

        for i in [(factor[0])*org, org, (factor[1])*org]:
            component.setParameter(column.columnType, parameter, i)

            blockPrint()
            self.runsim(instructions)
            enablePrint()

            tsol = self.getSolutionTime()/60
            solutions = self.getItemSolution(self.S.uvSens)
            plt.plot(tsol, solutions[index]*component.K_uv, label=component.name +", "+ parameter +"= "+str(component.getParameter(parameter, column.columnType)))
        plt.plot(tsol, solutions[0], label="Salt")
        plt.title("Sensitivity (measured at UV-sensor)")
        plt.legend()
        plt.show(block = False)
```

A.4 The csv-reader function

```
def readLog(file_name):
    data = {}
    with open(file_name) as csvfile:
        reader = csv.reader(csvfile)

        first_row = next(reader)
        for name in first_row:
            data[str(first_row.index(name))] = []
            data["Index " + str(name)] = first_row.index(name)

        for row in reader:
            n = 0
            for value in row:
                data[str(n)].append(float(value))
                n+=1
    return data
```

A.5 The layer function

```
def layer(t, parameters, instructions, orbsim, comp_name, item, value1 = None, value2 = None, value3 = None, value4 = None):

    for comp in orbsim.S._sim['compObjects']:
        if comp.name == comp_name:
            component = comp
            index = orbsim.S._sim['compObjects'].index(comp)

    if len(parameters) == 1:
        component.setParameter(parameters[0][0], parameters[0][1], 10**(value1))

    elif len(parameters) == 2:
        component.setParameter(parameters[0][0], parameters[0][1], value1*parameters[0][2])
        component.setParameter(parameters[1][0], parameters[1][1], value2*parameters[1][2])

    elif len(parameters) == 3:
        component.setParameter(parameters[0][0], parameters[0][1], value1*parameters[0][2])
        component.setParameter(parameters[1][0], parameters[1][1], value2*parameters[1][2])
        component.setParameter(parameters[2][0], parameters[2][1], value3*parameters[2][2])

    elif len(parameters) == 4:
        component.setParameter(parameters[0][0], parameters[0][1], value1*parameters[0][2])
        component.setParameter(parameters[1][0], parameters[1][1], value2*parameters[1][2])
        component.setParameter(parameters[2][0], parameters[2][1], value3*parameters[2][2])
        component.setParameter(parameters[3][0], parameters[3][1], value4*parameters[3][2])

    tsol, sol = orbsim.runsim(instructions,
                              calibration = {'item': item,
                                             't_Data': t})



    return sol[index]
```

A.6 The calibration method

```
    def calibrate(self, file_name, parameters, instructions, comp_name, item, pre = None, bounds = (0, np.inf)):
        if len(parameters) == 1:

            for comp in self.S._sim['compObjects']:
                if comp.name == comp_name:
                    component = comp
                    index = self.S._sim['compObjects'].index(comp)

            # Reads csv-file
            data = readLog(file_name)

            t_data = np.array(data['0'])
            c_data = np.array(data[str(self.S._sim['compObjects'].index(component)+1)]) # Plus one since time takes up first spot

            parameter_guesses_scaled = []
            parameter_names = []

            for i in range(len(parameters)):
                # Set parameter to guesses
                component.setParameter(parameters[i][0], parameters[i][1], parameters[i][2])

                # Scale with log10
                parameter_guesses_scaled.append(np.log10(parameters[i][2]))
                parameter_names.append(parameters[i][1])

            # Generate guess solutions
            blockPrint()
            self.verbosity = 0
            self.runsim(instructions)

            tsol_guess = self.getSolutionTime()/60
            solutions_guess = self.getItemSolution(item)

            if 0:
                raise Exception(c_data)

            # Precalibration
            if pre != None:
                fun_pre = lambda t, value1: layer(t, pre, instructions, self,
                                                  comp_name, item, value1 = value1)


                pre_opt1, pcov1 = curve_fit(fun_pre, t_data, c_data, np.log10(pre[0][2]), bounds = bounds, ftol = 1e-8)

                component.setParameter(pre[0][0], pre[0][1], 10**pre_opt1[0])


            fun1 = lambda t, value1: layer(t, parameters, instructions, self,
                                              comp_name, item, value1 = value1)


            param_opt1, pcov1 = curve_fit(fun1, t_data, c_data, parameter_guesses_scaled, epsfcn = 1e-8, ftol = 1e-12) #, bounds = bounds,

            parameter_guesses = []
            parameter_cal = []

            for i in range(len(parameters)):
                if len(parameters)==1:
                    component.setParameter(parameters[i][0], parameters[i][1], 10**(param_opt1[i]))
                    parameter_guesses.append(parameters[i][2])
```

```
                    parameter_cal.append(10**(param_opt1[i]))
                    printc(param_opt1[i])
                else:
                    component.setParameter(parameters[i][0], parameters[i][1], param_opt1[i]*parameters[i][2])
                    parameter_guesses.append(parameters[i][2])
                    parameter_cal.append(param_opt1[i]*parameters[i][2])


            self.runsim(instructions)
            enablePrint()


            tsol = self.getSolutionTime()/60
            solutions = self.getItemSolution(item)
            plt.figure(3)
            plt.xlabel('Time (s)')
            plt.plot(tsol_guess, solutions_guess[index]*component.K_uv, '*', label= "Guess" + str(parameter_names) + " =" + str(parameter_guesses))
            plt.plot(t_data/60, c_data*component.K_uv, 'o', label= "Data " + component.name, alpha = 0.5)
            plt.plot(tsol, solutions[index]*component.K_uv, 'k', label= "Cal" + str(parameter_names) + " =" + str(parameter_cal))
            plt.title("Calibration (measured at UV-sensor) " + component.name)
            plt.legend()
            plt.show(block = False)

        else:
            for comp in self.S._sim['compObjects']:
                if comp.name == comp_name:
                    component = comp
                    index = self.S._sim['compObjects'].index(comp)

            # Reads csv-file
            data = readLog(file_name)

            t_data = np.array(data['0'])
            c_data = np.array(data[str(self.S._sim['compObjects'].index(component)+1)]) # Plus one since time takes up first spot
```

```python


        # Set parameter to guesses
        for i in range(len(parameters)):
            component.setParameter(parameters[i][0], parameters[i
][1], parameters[i][2])

        # Generate guess solutions
        blockPrint()
        self.verbosity = 0
        self.runsim(instructions)

        tsol_guess = self.getSolutionTime()/60
        solutions_guess = self.getItemSolution(item)

        if 0:
            raise Exception(c_data)

        # Precalibration
        if pre != None:
            fun_pre = lambda t, value1: layer(t, pre, instructions,
 self,
                                           comp_name, item, value1 =
 value1)


            pre_opt1, pcov1 = curve_fit(fun_pre, t_data, c_data,
pre[0][2]/pre[0][2], bounds = bounds, ftol = 1e-8)

            component.setParameter(pre[0][0], pre[0][1], pre_opt1
[0]*pre[0][2])


        # Decide how many parameters
        if len(parameters) == 1:
            fun1 = lambda t, value1: layer(t, parameters,
instructions, self,
                                           comp_name, item, value1 =
 value1)

        elif len(parameters) == 2:
            fun1 = lambda t, value1, value2: layer(t, parameters,
instructions, self,
                                                 comp_name, item,
value1 = value1, value2 = value2)

        elif len(parameters) == 3:
            fun1 = lambda t, value1, value2, value3: layer(t,
parameters, instructions, self,
                                                 comp_name, item,
value1 = value1, value2 = value2, value3 = value3)
        elif len(parameters) == 4:
            fun1 = lambda t, value1, value2, value3, value4: layer(
t, parameters, instructions, self,comp_name, item, value1 = value1,
 value2 = value2, value3 = value3, value4 = value4)

        parameter_guesses = []
        parameter_names = []
        for i in parameters:
            parameter_guesses.append(i[2]/i[2])
            parameter_names.append(i[1])

        param_opt1, pcov1 = curve_fit(fun1, t_data, c_data,
parameter_guesses, bounds = bounds) #

        parameter_guesses = []
        parameter_cal = []
        for i in range(len(parameters)):
            component.setParameter(parameters[i][0], parameters[i
][1], param_opt1[i]*parameters[i][2])
            parameter_guesses.append(parameters[i][2])
            parameter_cal.append(param_opt1[i]*parameters[i][2])



        self.runsim(instructions)
        enablePrint()

        tsol = self.getSolutionTime()/60
        solutions = self.getItemSolution(item)
        plt.figure(3)
        plt.xlabel('Time (s)')
        plt.plot(tsol_guess, solutions_guess[index]*component.K_uv,
 '*', label= "Guess" + str(parameter_names) + " =" + str(
parameter_guesses))
```

```
            plt.plot(t_data/60, c_data*component.K_uv, 'o', label= ”Data ” + component.name, alpha = 0.5)
            plt.plot(tsol, solutions[index]*component.K_uv, 'k', label=”Cal” + str(parameter_names) + ” =” + str(parameter_cal))
            plt.title(”Calibration (measured at UV−sensor) ” + component.name)
            plt.legend()
            plt.show(block = False)
```

B Appendix 2 - Fake Experiments

B.1 Fake Experiment Cyto

```
from system_pure_sim import *
from process_sim import Phase, Process
from sim.simulator import Buffer,Component, OrbitSim
import matplotlib.pyplot as plt
import os

def method(S,gradientTime):

    CV = 1 # column volume
    flowrateCol = 1 # flowrate in the column

    ''' ************ Define buffers and components ************* '''
    # --- Define different adsorption parameters
    SEC = {"K_e": 1}
    IEX_SMA = {"kkin":1e-4, "K": 5e10, "qmax": 51, "nu": 4.1}
    IEX_MPM = {"kkin":1e-4, "H_0": 5e8, "qmax": 51, "beta": 4.1}
    HIC = {"kkin":1e-0, "K_0": 50, "qmax": 51, "gamma": 0.04} #9.7,
    0.0415

    ads = {"SEC": SEC, "IEX_SMA": IEX_SMA, "IEX_MPM": IEX_MPM, "HIC":
    HIC}

    # --- Define the components in the system
    c1 = Component(name = 'Salt',
                    S=S,
                    Mw = 18.01,
                    modifier = True,
                    k_eff = 1e-10, # Only used if film-model is
    selected
                    K_uv=1) # to match the conductivity

    c2 = Component(name = 'Cyto',
                    S=S,
                    Mw = 12.4e3,
                    K_uv = 10000,
                    ads = ads,
                    k_eff = 1e-10
                  )

    # -- Define the buffers in the system
    b1 = Buffer(conc = {'Salt':0.0005e3, 'Cyto':0}) #
    b2 = Buffer(conc = {'Salt':0.5e3}) #
    b3 = Buffer(conc = {'Salt':0, 'Cyto':1/12.4e3*1e3}) #1 g/L of Lyzosyme / molecular weigt / 100 so that unit becomes mol/m3

    # -- Fill system with buffer
    b1.fillSystem(S)
    b2.fill(S.pump.p['OutB'].tube)
    b2.fill(S.inlValveB)

    b2.fill(S.inlValveB.p['Out'].tube)
    #b2.fill(S.pump.p['OutB'])

    b3.fill(S.supLoop)
    b3.fill(S.supLoop.p['Out'].tube)

    # -- Connect buffers to inlets
    b1.connect(S.inlValveA.p['5'])
    b2.connect(S.inlValveB.p['1'])

     '''PumpMap for Simulation trails on Massimo'''
    #
    #   S1:
    #   S2:
    #   S3: y
    #   S4:
    #   S5:
    #   S6: Buffer A 20mM Soudium Phosphate
    #   S7:
    #   S0
    #
    #   A1:
    #   A2:
    #   A3:
    #   A4:
    #   A5: Buffer A 20mM Sodium Phosphate
    #    A6:
    #    A7:
    #
    #   B1: Buffer B 20mM Sodium Phosphate, 500mM NaCl
    #   B2:
    #   B3:
    #   B4:
    #   B5:
    #   B6:
    #   B7: 20% EtOh
    #
    #    Change inlets to match the AKTA setup
    #
    ..............................................................
    pumpwMap = {'BufferA':             {'%B':0,      'A':5,   'B':1},
                'BufferB':             {'%B':100,    'A':5,   'B':1},
                'SetUp':               {'%B':0,      'A':5,   'B':1},
                'Eluation':            {'%B':40,     'A':5,   'B':1}}

    ''' ************ Define phases ************* '''

    flush1 =          Phase(name = 'flush1',
                            SetInlet = pumpwMap['BufferA'],
                            Column = col1,
                            FlowRate = 0.5,
                            Time = 1*60)

    inject =         Phase(name = 'inject',
                            Column = col1,
                            Inject = True ,
                            Time = 1*60)

    flush2 =          Phase(name = 'flush2',
                            SetInlet = pumpwMap['BufferA'],
                            Column = col1,
                            Time = 1*60)

    eluate =         Phase(name = 'eluate',
                            Inject = False,
                            Column = col1,
                            Gradient = [0,100],
                            Time = gradientTime*60)

    finalEluation = Phase(name = 'finalEluation',
                          SetInlet = pumpwMap['BufferB'],
                          Column = col1,
                          Time = 15*60)

    clear =             Phase(name = 'Clear',
                          SetInlet = pumpwMap['BufferA'],
                          Column = col1,
                          Time = 6*60)

    phases = [flush1,inject,flush2,eluate,finalEluation,clear]


    P = Process(name='FAKE_EXPERIMENT_Cyto',
                phases=phases,
                S=S)

    options = {'mode':'test',
               'timeFactor':100.,
               'sampleSignals':['uv1'],
               'sampleTime':1.0,
               'loops':[],
               'logData':True,
               'logRun':True,
               'simulation':{'calibrate':False,
                             'simulate': True}}

    # Set initial settings (optional)
    P.I0 = [(0, S.inlValveA.set , {'position': 5})]

    #P.run(options)


    return P



if __name__ == '__main__':

    S1 = SystemPure(mode='test')
    P1 = method(S1,8)

    orbsim = OrbitSim(S=S1)
    orbsim.runsim(P1.I, plotFlag=True)

    if 0:
        S2 = SystemPure(mode='test')
        P2 = method(S2,20)
        P2.orbsim.logItemSolution(S2.uvSens,P2.folder, nameSuffix = '
   Cyto_20minGrad')
```

B.1.1 Tube connections

```
'''     source,              destination,          info'''
Tube( inlValveA.p['Out'], pump.p['InA'], length=30.,diameter=0.29)
Tube( inlValveB.p['Out'], pump.p['InB'], length=30.,diameter=0.29)
Tube( pump.p['OutA'],   mixer.p['InA'], length=34.+11.5+40,diameter=0.1)
    #via pressure monitor
Tube( pump.p['OutB'],   mixer.p['InB'], length=34.+11.5+40,diameter=0.1)
     #via pressure monitor

Tube( mixer.p['Out'],    injValve.p['SyP'],   color='green', length=43.5)
    #
Tube( injValve.p['Col'],   colValve.p['In'],    color='green', length
   =20.2) #

Tube( colValve.p['Out'],   uvSens.p['In'],    color='green', length=16.7)
    #

Tube( uvSens.p['Out'],     condSens.p['In'],    color='green', length
   =17.0) #
Tube( condSens.p['Out'],   phSens.p['In'],      color='green', length
   =30.5/2) #
Tube( phSens.p['Out'],     outlValve.p['In'],   color='green', length
   =30.5/2) #
```

B.2 Fake Experiment Membrane

```
from system_Membrane_sim import *
from Andreas_process import Phase, Process
from sim.simulator import Buffer,Component, OrbitSim
import matplotlib.pyplot as plt
```

```
#from sim.calibrator import orbitCalibrate

def method(S,gradientTime):

    CV = 1 # column volume
    flowrateCol = 1 # flowrate in the column

    ''' ************ Define buffers and components ************* '''
    # —— Define different adsorption parameters
    SEC = {"K_e": 1}
    IEX = {"SMA":{"kkin":1e-4, "K": 5e10, "qmax": 51, "nu": 4.1},
           "MPM":{"kkin":1e-4, "H_0": 5e8, "qmax": 51, "beta": 4.1}}
    HIC = {"kkin":1e-0, "K_0": 50, "qmax": 51, "gamma": 0.04} #9.7, 0.0415


    ads = {"SEC": SEC, "IEX": IEX, "HIC": HIC}

    # —— Define the components in the system
    c1 = Component(name = 'Salt',
                   S=S,
                   Mw = 18.01,
                   modifier = True,
                   k_eff = 1e-7, # Only used if film-model is selected
                   K_uv=1) # to match the conductivity

    c2 = Component(name = 'Cyto',
                   S=S,
                   Mw = 12.4e3,
                   K_uv = 10000,
                   ads = ads,
                   k_eff = 1e-7
                  )

    # —— Define the buffers in the system
    b1 = Buffer(conc = {'Salt':0.0005e3, 'Cyto':0})
    b2 = Buffer(conc = {'Salt':0.5e3})
    b3 = Buffer(conc = {'Salt':0.0005e3, 'Cyto':1/12.4e3*1e3}) #1 g/L of Lyzosyme / molecular weigt / 100 so that unit becomes mol/m3

    # —— Fill system with buffer
    b1.fillSystem(S)
    #b2.fill(S.pump.p['OutB'].tube)

    # Connect sample
    #b3.connect(S.pump.p['InA'])
    b3.connect(S.inlValveA.p['5'])


    '''PumpMap for Simulation trails on Massimo'''
    #
    #   S1:
    #   S2:
    #   S3: y
    #   S4:
    #   S5:
    #   S6: Buffer A 20mM Soudium Phosphate
    #   S7:
    #   S0
    #
    #   A1:
    #   A2:
    #   A3:
    #   A4:
    #   A5: Buffer A 20mM Sodium Phosphate
    #   A6:
    #   A7:
    #
    #  B1: Buffer B 20mM Sodium Phosphate, 500mM NaCl
    #  B2:
    #  B3:
    #  B4:
    #  B5:
    #  B6:
    #  B7: 20% EtOh
    #
    #   Change inlets to match the AKTA setup
    #
    ..............................................................
    pumpwMap = {'BufferA':             {'%B':0,      'A':5,   'B':1},
                'BufferB':             {'%B':100,    'A':5,   'B':1}
                }

    ''' ************ Define phases ************* '''
    load =          Phase(name = 'load',
                          SetInlet = pumpwMap['BufferA'],
                          Membrane = membrane,
                          Gradient = [0, 50],
                          FlowRate = 0.5,
                          Time = 0)

    phases = [load]


    P = Process(name='FAKE_EXPERIMENT_Membrane',
                phases=phases,
                S=S)

    options = {'mode':'test',
               'timeFactor':100.,
               'sampleSignals':['uv1'],
               'sampleTime':1.0,
               'loops':[],
               'logData':True,
               'logRun':True,
               'simulation':{'calibrate':False,
                             'simulate': True}}

    # Set initial settings (optional)
    P.I0 = [(0, S.inlValveA.set , {'position': 5})]

    #P.run(options)


    return P



if __name__ == '__main__':

    S1 = SystemPure(mode='test')
    P1 = method(S1,8)

    orbsim = OrbitSim(S=S1)
    orbsim.runsim(P1.I, plotFlag=True)
```

B.2.1 Tube connections

```
'''    source,          destination,         info'''
Tube( inlValveA.p['Out'], pump.p['InA'], length=30.,diameter=0.29)
Tube( pump.p['OutA'], colValve.p['In'], length=30., diameter=0.29)
Tube(colValve.p['Out'], pump.p['InB'], length=34.+11.5+40,diameter=0.1)
```

B.3 Fake Experiment With Two Columns

```
from SIM_system_pure_RECIRC import *
import numpy as np
from  process_sim_RECIRC import Phase, Process
from sim.simulator import Buffer,Component, OrbitSim
import os


def method(S):
#%%
    ''' ************ Define buffers and components ************* '''
    # --- Define different adsorption parameters
    SEC = {"K_e": 0}
    IEX_SMA = {"kkin":1e-4, "K": 5e10, "qmax": 51, "nu": 4.1}
    IEX_SMA_Zyto = {"kkin":1e-4, "K": 7e10, "qmax": 51, "nu": 4.1}
    IEX_MPM = {"kkin":1e-4, "H_0": 5e8, "qmax": 51, "beta": 4.1}



    ads1 = {"SEC": SEC, "IEX_SMA": IEX_SMA, "IEX_MPM": IEX_MPM}
    ads2 = {"SEC": {"K_e":1}}
    ads3 = {"SEC": {"K_e":1}, 'IEX_SMA':IEX_SMA_Zyto}
    # --- Define the components in the system


    c1 = Component(name = 'Salt',
                   S=S,
                   Mw = 18.01,
                   modifier = True,
                   k_eff = 1e-2, # Only used if film-model is selected
                   ads = ads2,
                   K_uv=1) # to match the conductivity

    c2 = Component(name = 'Cyto',
                   S=S,
                   Mw = 12.4e3,
                   K_uv = 10000,
                   ads = ads1,
                   k_eff = 1e-2
                  )

    c3 = Component(name = 'Zyto',
                   S=S,
                   Mw = 12.4e3,
                   K_uv = 10000,
                   ads = ads3,
                   k_eff = 1e-2
                  )

    # -- Define the buffers in the system
    b1 = Buffer(conc = {'Salt':0.0005e3, 'Cyto':0}) # , 'Zyto': 1/12.4e3*1e3
    b2 = Buffer(conc = {'Salt':0.5e3}) #e3
    b3 = Buffer(conc = {'Salt':0, 'Cyto':1/12.4e3*1e3, 'Zyto': 1/12.4e3*1e3}) #1 g/L of Lyzosyme / molecular weigt / 100 so that unit becomes mol/m3

    # -- Fill system with buffer
    b1.fillSystem(S)
    b2.fill(S.pump.p['OutB'].tube)
    b2.fill(S.inlValveB)

    b2.fill(S.inlValveB.p['Out'].tube)
    #b2.fill(S.pump.p['OutB'])

    b3.fill(S.supLoop)
    b3.fill(S.supLoop.p['Out'].tube)

    # -- Connect buffers to inlets
    b1.connect(S.inlValveA.p['5'])
    b2.connect(S.inlValveB.p['1'])



    # ..............................................................
    pumpwMap = {'BufferA':              {'%B':0,      'A':5,   'B':1},
                'BufferB':              {'%B':100,    'A':5,   'B':1},
                'SetUp':                {'%B':0,      'A':5,   'B':1},
                'Eluation':             {'%B':40,     'A':5,   'B':1}}


    flush1 =          Phase(name = 'flush1',
                       SetInlet = pumpwMap['BufferA'],
                       Column = col1,
                       FlowRate = 0.5,
                       Time = 1*60)

    inject =         Phase(name = 'inject',
                       Column = col1,
                       Inject = True  ,
                       Time = 1*60)

    Wash11 =Phase(name='Wash11',
                 Time = 1*60,
                 Column =col1,
                 SetInlet = pumpwMap['BufferA']
                 )

    Elution1 =         Phase(name = 'Elution1',
                       Inject = False,
                       Column = col1,
                       Gradient = [0,100],
                       Time = 20*60.,
                       Recirculation = [17*60, 20*60]
                       )



    Wash2 =Phase(name='Wash2',
                 Time = 5*60,
                 Column =col2,
                 SetInlet =pumpwMap['BufferA'],
                 PoolingTime = [4*60, 5*60]
                 )

    Wash12 =Phase(name='Wash12',
                 Time = 5*60,
                    Column =col1,
                    SetInlet =pumpwMap['BufferA']
                    )


    phases = [flush1, inject, Wash11, Elution1, Wash2, Wash12]

    P = Process('Double', phases, S)

    # Set initial settings
    P.I0 = [(0., S.phSens.pH_ON, {'pH_pos':1,'restrictor_pos':1})]

    return P
#%%

if __name__ == '__main__':

    S = SystemPureNics(mode='test')
    P = method(S)

    orbsim = OrbitSim(S=S)
    orbsim.runsim(P.I, plotFlag=True)
```

B.3.1 Tube connections

```
'''     source,             destination,          info'''
Tube( inlValveA.p['Out'], pump.p['InA'], length=30.,diameter=0.29)
Tube( inlValveB.p['Out'], pump.p['InB'], length=30.,diameter=0.29)
Tube( pump.p['OutA'],   mixer.p['InA'], length=34.+11.5+40,diameter=0.1)     #via pressure monitor
Tube( pump.p['OutB'],   mixer.p['InB'], length=34.+11.5+40,diameter=0.1)      #via pressure monitor

Tube( mixer.p['Out'],     injValve.p['SyP'],   color='green', length=43.5)     #
Tube( injValve.p['Col'], versValve1.p["1"],     color='green', length=20.2) #
Tube( versValve1.p["4"], colValve.p["In"], color='green', length=16.7)

Tube( colValve.p["Out"], versValve2.p["4"])
Tube( versValve2.p["1"], uvSens.p['In'],     color='green', length=16.7)

Tube( uvSens.p['Out'],      condSens.p['In'],    color='green', length
    =17.0) #
Tube( condSens.p['Out'],  phSens.p['In'],       color='green', length
    =30.5/2) #
Tube( phSens.p['Out'],      outlValve.p['In'],  color='green', length
    =30.5/2)

Tube( outlValve.p['Out1'], versValve1.p['3'], color='green', length
    =16.7)
Tube( versValve1.p['2'], loopValve.p["In"], color='green', length=16.7)
Tube( loopValve.p["Out"], versValve2.p["2"], color='green', length
    =16.7)
```