



Hochschule für Angewandte  
Wissenschaften Hamburg  
*Hamburg University of Applied Sciences*

**Hochschule für Angewandte Wissenschaften Hamburg**

**Fakultät Life Sciences**

**Entwicklung und Etablierung einer TFF-Anlage zur  
Durchführung und Überwachung automatisierter  
Konzentrations- und Diafiltrationsprozesse**

Masterarbeit

im Studiengang *Pharmaceutical Biotechnology*

vorgelegt von

**Fabian Weirauch**



Hamburg

am 13.02.2022

1. **Gutachter:** Prof. Dr. Christian Kaiser (HAW Hamburg)
2. **Gutachterin:** Prof. Dr. Gesine Cornelissen (HAW Hamburg)



## Danksagung

An dieser Stelle möchte ich mich bei all denjenigen bedanken, die mich während der Anfertigung dieser Masterarbeit und meiner gesamten Studienzeit unterstützt und motiviert haben.

In erster Linie gebührt mein Dank meinem Betreuer, Herrn Prof. Dr. Christian Kaiser, der meine Masterarbeit betreut und begutachtet hat. Ich möchte mich für seine fachliche Unterstützung sowie für die Zeit und Mühe bedanken, die er in meine Arbeit investiert hat.

Ein großer Dank gebührt Prof. Dr. Gesine Cornelissen für ihre Bereitschaft, auch für diese Arbeit als Gutachterin tätig gewesen zu sein.

Ein besonderer Dank gilt Ulrich Scheffler, der mir bei der Anwendungsentwicklung mit viel Hilfsbereitschaft zur Seite stand, sowie Petra Derr und Florian Schiffel für ihre praktische wie auch ideenreiche Unterstützung und Hilfe im Labor.

Nicht zuletzt möchte ich mich bei meiner Familie bedanken, durch deren mentale sowie finanzielle Unterstützung das gesamte Studium erst möglich war. Ebenso möchte ich mich bei meiner Freundin Paula bedanken, die mich auf diesem Weg begleitet hat und stets ein offenes Ohr für mich hatte.

## Zusammenfassung

Tangentialflussfiltrationen, auch bekannt als Querstromfiltrationen, gelten in der Biotechnologie und der pharmazeutischen Wirkstoffproduktion als etabliertes Verfahren im *Downstream Processing*. Ob zum Pufferaustausch, bei der Abtrennung von Zellen und Zellbruchstücken oder bei der Aufkonzentrierung von gelösten Produkten, stellt die Tangentialflussfiltration eine attraktive Alternative zu herkömmlichen Abtrennungsverfahren wie die Zentrifugation oder Dead-End-Filtrationen dar.

Die vorliegende Arbeit befasst sich mit der Entwicklung einer Tangentialflussfiltrationsanlage inklusive einer Computer-Anwendung zur Planung, automatisierten Durchführung und Überwachung von Konzentrations- und Diafiltrationsprozessen. Die Entwicklung der graphischen Benutzeroberfläche erfolgte in der Entwicklungsumgebung Matlab App Designer<sup>®</sup>. Zur Verifikation der Funktionalität der Anlage und der Anwendung wurden Konzentrationsversuche mit *Saccharomyces cerevisiae* Zellsuspensionen durchgeführt und auf ihre Reproduzierbarkeit untersucht.

Die praktische Anwendung der Tangentialflussfiltrationsanlage hat die Fähigkeit der Anlage bestätigt, eine zellhaltige Suspension unter dem Gütekriterium der vollständigen Zellabtrennung bei gleichzeitiger Aufkonzentrierung zu filtrieren. Die Computer-Anwendung hat sich gemäß dem definierten Konzept bei der Planung, der Durchführung und der Live-Überwachung der Konzentrationsversuche als funktional und zuverlässig erwiesen. Die Reproduzierbarkeitsversuche haben gezeigt, dass die Filtrationen, insbesondere zum Filtrationsende, einer starken Variabilität unterliegen. Gründe für die hohe Variabilität konnten im Rahmen dieser Masterarbeit nicht exakt identifiziert werden.

## Abstract

Tangential flow filtration, also known as cross-flow filtration, is an established procedure in downstream processing in biotechnology and pharmaceutical active ingredient production. Whether for buffer exchange, when separating cells and cell fragments or for the recovery and purification of dissolved products, tangential flow filtration is considered to be an attractive alternative to conventional separation methods such as centrifugation or dead-end-filtration.

This thesis deals with the development of a tangential flow filtration plant including a computer application for planning, automated execution and monitoring of concentration and diafiltration processes. The graphical user interface was developed in the Matlab App Designer<sup>®</sup> development environment. Concentration experiments were carried out with *Saccharomyces cerevisiae* cell suspensions to verify the functionality of the plant and its application. In the course of this, the reproducibility of the concentration experiments was investigated.

The practical usage of the tangential flow filtration plant has confirmed the capability of the system to filter a cell containing suspension under the quality criterion of complete cell separation with simultaneous concentration. During the planning, execution and live monitoring of the concentration experiments, the computer application has proven to be fully functional and reliable. The reproducibility trials have indicated that the filtrations show a high degree of variability, especially at the end of the filtration process. Reasons for the high variability could not be precisely identified within the scope of this master thesis.



# Inhaltsverzeichnis

Abkürzungsverzeichnis.....	I
<b>1 Einleitung und Zielsetzung .....</b>	<b>1</b>
<b>2 Theoretischer Hintergrund .....</b>	<b>3</b>
2.1 Filtration und Filtrationsverfahren .....	3
2.1.1 Klassische Filtration.....	4
2.1.2 Membranverfahren und -filtration.....	7
2.2 Regelungstechnik.....	31
2.3 Matlab R2021a®.....	34
2.3.1 Matlab App Designer® .....	36
2.4 <i>Saccharomyces cerevisiae</i> .....	38
<b>3 Material und Methoden.....</b>	<b>39</b>
3.1 Materialien.....	39
3.1.1 Geräte .....	39
3.1.2 Verbrauchsmaterialien.....	40
3.1.3 Chemikalien.....	41
3.1.4 Lösungen.....	41
3.1.5 Verwendeter Organismus.....	41
3.2 Methoden .....	42
3.2.1 Vorversuche .....	42
3.2.2 Wasserwert-Versuche des Hohlfasermoduls.....	44
3.2.3 Konzentrationsversuche der <i>Saccharomyces cerevisiae</i> Zellsuspensionen .....	45
<b>4 Entwicklung der TFF-Anlage .....</b>	<b>49</b>
4.1 Aufbau der Anlage.....	49



4.1.1 Systemintegration.....	51
4.1.2 Gefäße und Schraubverschlüsse .....	52
4.1.3 Hohlfasermodule .....	54
4.1.4 Druckmonitor und mobile Drucksensoren.....	55
4.1.5 Pumpen und Schläuche .....	57
4.1.6 Waagen .....	59
4.1.7 Tangentialflussfiltrationsanlage .....	60
4.2 Entwicklung der graphischen Benutzeroberfläche.....	62
4.2.1 Konzept .....	62
4.2.2 Layout der Anwendung.....	62
4.2.3 Serielle Kommunikation mit der Hardware .....	79
4.2.4 Implementierung.....	81
<b>5 Praktische Anwendung der TFF-Anlage.....</b>	<b>130</b>
5.1 Vorversuche .....	130
5.1.1 Aktivierungszeit der suspendierten <i>Saccharomyces cerevisiae</i> .....	130
5.1.2 Ermittlung des Korrelationsfaktors zwischen Biotrockenmassekonzentration und OD <sub>600nm</sub> .....	131
5.1.3 Wasserwert-Versuche des Hohlfasermodule.....	133
5.2 Konzentrationsversuche der <i>Saccharomyces cerevisiae</i> Zellsuspensionen .	135
<b>6 Fazit und Ausblick.....</b>	<b>139</b>
<b>7 Literaturverzeichnis.....</b>	<b>141</b>
<b>8 Anhang .....</b>	<b>i</b>
8.1 Abbildungsverzeichnis.....	i
8.2 Tabellenverzeichnis.....	vii
8.3 Rohdaten.....	viii
8.3.1 Vorversuche .....	viii



8.4 Hardware-Informationsblätter .....	X
8.4.1 Hohlfasermodul .....	X
8.5 Graphische Benutzeroberfläche .....	xi
8.5.1 Globale Variablen .....	xi
8.5.2 Kurzanleitung der Computer-Anwendung.....	xvi
8.6 Standardarbeitsanweisungen .....	xxvi
8.6.1 OD <sub>600nm</sub> - Messung .....	xxvi
8.6.2 BTM-Bestimmung.....	xxx
8.7 Erklärung zu verwendeten Hilfsmitteln .....	xxxiv

# Abkürzungsverzeichnis

ad.	lat. Präposition deu.: auf; hier.: auffüllen
AG	Aktiengesellschaft
AU	engl.: absorption units deu.: Absorptions-Einheiten
BTM	Biotrockenmassekonzentration
CF	engl.: <i>concentration factor</i> deu.: Konzentrationsfaktor
CIP	engl.: <i>cleaning in place</i> deu.: ortsgebundene Reinigung
COM-Port	engl.: <i>communication port</i> deu.: Kommunikationsanschluss
deu.	deutsch
DV	Diafiltrationsvolumen
e	Regelabweichung
engl.	englisch
<i>et al.</i>	lat.: <i>et aliis</i> deu.: und andere
GUI	engl.: <i>graphical user interface</i> deu.: graphische Benutzeroberfläche
G-Zahl	Relative Zentrifugalbeschleunigung
HAW	Hochschule für Angewandte Wissenschaften
ID	Innendurchmesser
K <sub>I</sub>	Integrationsfaktor
K <sub>P</sub>	Proportionalfaktor
lat.	latein
M	Mol, molar
MF	Mikrofiltration
mPES	modifiziertes Polyethersulfon
M <sub>r</sub>	engl.: <i>molecular weight</i> deu.: Molekulargewicht
MWCO	engl.: <i>molecular weight cut-off</i> deu.: Molekulargewichtsgrenzwert
OD	Optische Dichte
PC	engl.: <i>personal computer</i> deu.: persönlicher Rechner
pH	pH-Wert, lat.: <i>pondus hydrogenii</i> deu.: Potential des Wasserstoffs
RPM	engl.: <i>rounds per minute</i> deu.: Runden pro Minute



RS232	engl.: <i>recommended standard 232</i> deu.: Standard für eine serielle Schnittstelle
TFF	Tangentialflussfiltrationsanlage
TMP	Transmembrandruck
u	Stellgröße
UF	Ultrafiltration
USB	engl.: <i>universal serial bus</i> deu.: serielles Bussystem
UV	Ultraviolettstrahlung
VE-Wasser	Vollentsalztes Wasser
$V_F$	Feedvolumen
$V_L$	Flüssigvolumen
$V_P$	Permeatvolumen
$V_R$	Retentatvolumen
°C	Grad Celsius
.txt	Endung einer Text-Datei
$\dot{\gamma}$	Schergeschwindigkeit

# 1 Einleitung und Zielsetzung

Über drei Millionen Liter Grundwasser werden pro Tag in St. Maurice les Chateauf (Frankreich) als Teil der städtischen Wasserversorgung mit Hilfe von Ultrafiltrationen aufbereitet. Viele Lebensmittel, die wir täglich essen und trinken, wurden während ihrer Herstellung mit Membranen prozessiert (Howell, 1993). Filtrationen mit Membranen haben sich in den letzten 30 Jahren von einem Sonderverfahren zu einem Standardverfahren verschiedenster Industriezweige entwickelt. Neben der erwähnten Wasseraufbereitung und der Getränke- und Nahrungsmittelindustrie kommen in der pharmazeutischen Produktion und in der Biotechnologie vor allem Mikro- und Ultrafiltrationen zum Einsatz (Ripperger, 1992; Rautenbach und Melin, 2007). Bei der rekombinanten Wirkstoffherstellung in der pharmazeutischen Biotechnologie unterläuft das Zielmolekül einem komplexen Produktionsvorgang. Nach der molekularbiologischen Stammentwicklung, der Charakterisierung des Wachstums- und Produktionsverhaltens des Stammes und der Fermentation der rekombinanten Zellen inklusive der Synthese der Wirksubstanz (*Upstream Processing*) schließen sich aufwendige Aufarbeitungs- und Reinigungsprozesse (*Downstream Processing*) an (Kayser, 2002). Insbesondere dynamische Filtrationsprozesse wie die Tangentialflussfiltration (TFF) gewinnen durch die heutzutage immer größer werdende Konzentration an Biomasse und Produkten im *Downstream Processing* an großer Bedeutung. Durch die tangentielle Membranüberströmung von Tangentialflussfiltrationsmodulen werden Partikel aus der sich bildenden Deckschicht durch Scher- und Auftriebskräfte zurück in die Kernströmung geführt, wodurch eine konstante Deckschichtdicke und ein gleichbleibender Permeatfluss realisiert werden kann. Leichtes *Up-Scaling*, eine kontinuierliche Betriebsweise sowie eine einfache Integration in bestehende Prozesse stellen nur einige der Vorteile gegenüber klassischen Abtrennungsmöglichkeiten wie z. B. die Zentrifugation oder die Dead-End-Filtration dar. Die Tangentialflussfiltration findet im *Downstream Processing* vorwiegend bei der Abtrennung von Zellen und Zellbruchstücken, bei der Aufkonzentrierung von gelösten Produkten und beim Pufferaustausch zur Konditionierung und finalen Puffereinstellung (Diafiltration) Anwendung (Rautenbach und Melin, 2007).

Primäres Ziel der vorliegenden Masterarbeit ist die Entwicklung einer Tangentialflussfiltrationsanlage (TFF-Anlage) und einer graphischen Benutzeroberfläche (GUI), die

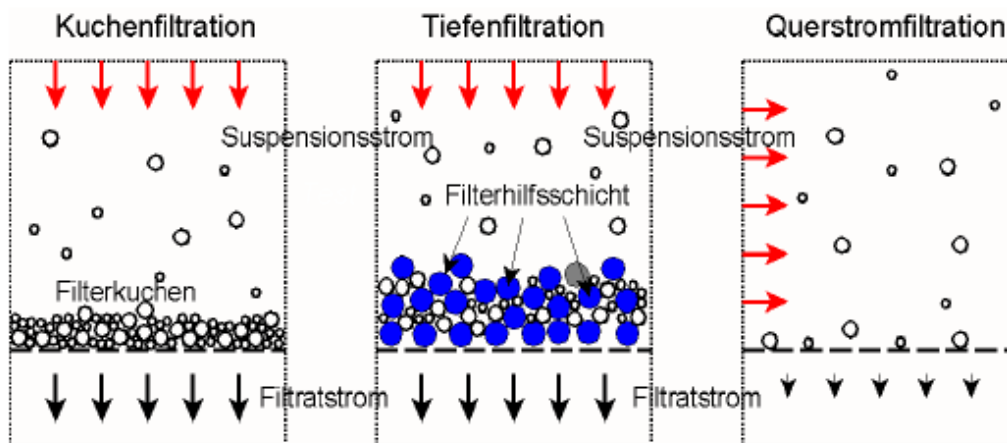
die Planung, Durchführung, Überwachung und Messdatendokumentation von Konzentrations- und Diafiltrationsprozessen über einen externen Rechner ermöglicht. Sekundäres Ziel ist die Etablierung eines Konzentrations- und Diafiltrationsprozesses einer *Saccharomyces cerevisiae* Suspension, um die Funktionalität der TFF-Anlage sowie der graphischen Benutzeroberfläche zu überprüfen. Für den Bau der TFF-Anlage werden zwei peristaltische Schlauchpumpen, ein Druckmonitor mit mobilen Drucksensoren, ein Hohlfasermodule, zwei Tischwaagen sowie drei Glasbehälter, Schläuche und Schraubverschlüsse verwendet. Die Programmierung der graphischen Benutzeroberfläche erfolgte über die Softwareentwicklungsumgebung "App Designer" des Programms Matlab® R2021a des Herstellers The MathWorks, Inc. Zum Erreichen des sekundären Ziels werden zunächst vier Konzentrationsversuche mit *Saccharomyces cerevisiae* Zellsuspensionen durchgeführt und auf ihre Reproduzierbarkeit untersucht.

## 2 Theoretischer Hintergrund

Im Folgenden werden die theoretischen Grundlagen verschiedener Filtrationsverfahren erläutert. Besonderes Augenmerk wird dabei auf die Membranverfahren Mikro- und Ultrafiltration gelegt, da die in dieser Masterthesis entwickelte Anlage diese Arten der Filtration nutzt. Neben den Erläuterungen zur Funktionsweise der Filtrationstechniken wird ein Einblick in die Regelungstechnik und in das Computer-Programm Matlab® gegeben. In dieser Softwareumgebung wurde die graphische Benutzeroberfläche der zur Filtrationsanlage gehörenden Anwendung designt und programmiert.

### 2.1 Filtration und Filtrationsverfahren

Die Filtration ist eine physikalische Trennmethode, bei dem Partikel aus einer Flüssigkeit oder einem Gas unter Zuhilfenahme eines Filtermediums abgetrennt werden (Stieß, 1997). Die sogenannte Trübe, zum Beispiel ein Fest-Flüssig Gemisch, kann durch verschiedene Kräfte durch ein gestütztes Filtermedium geführt werden (Tien, 2012). Dabei wird der enthaltene Feststoff durch das Filtermedium zurückgehalten, wohingegen die Flüssigkeit des Gemisches den Filter passieren kann (Anspach, 2018). Die treibende Kraft, welche für die Durchströmung des Filtermediums erforderlich ist, kann zum Beispiel die Gravitation, ein Druckgefälle oder die Zentrifugalkraft sein. Es werden grundsätzlich drei Arten der Filtration unterschieden: die Kuchenfiltration (Dead-End-Filtration), die Tiefenfiltration und die Tangentialflussfiltration (auch Querstromfiltration oder Cross-Flow-Filtration genannt) (Abbildung 2.1) (Stieß, 1997). Bei einem Filtrationsprozess entsteht die Klarflüssigkeit (auch Filtrat oder Permeat genannt), welche frei oder nahezu frei von Partikeln ist, und eine Festkörpermasse mit sehr geringem Anteil an mitgeführter Flüssigkeit. Bei einer Tangentialflussfiltration entsteht statt einer Festkörpermasse meist ein Fest-Flüssig Gemisch mit erhöhter Feststoffkonzentration (Tien, 2012). Kuchen- und Tiefenfiltration zählen zu den klassischen Filtrationsarten, wohingegen die Tangentialflussfiltration oft Anwendung in modernen Membranverfahren findet (Rautenbach und Melin, 2007).



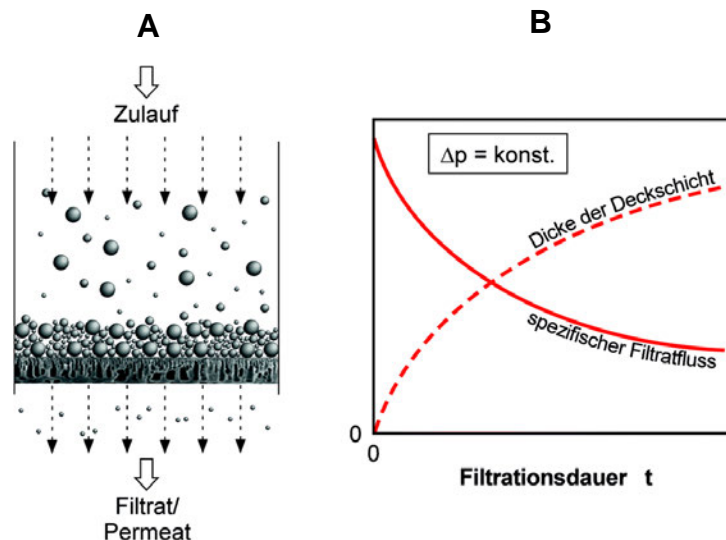
**Abbildung 2.1:** Darstellungen der drei grundsätzlichen Filtrationsarten **Kuchenfiltration**, **Tiefenfiltration** und **Tangentialflussfiltration (Querstromfiltration)**. Zu sehen sind die Richtungen der Suspensionsströme und der Filtratströme, das Filtermedium sowie die Filtermittelhilfsschicht (Anspach, 2018).

## 2.1.1 Klassische Filtration

### 2.1.1.1 Kuchen Filtration

Bei einer Dead-End-Kuchenfiltration fließt die zu filtrierende Flüssigkeit durch das Filtermedium, wobei die Zurückhaltung der Partikel auf der Oberfläche des Filtermediums stattfindet (Tien, 2012). Das gewünschte Produkt kann dabei sowohl im Filterkuchen enthalten als auch im Filtrat gelöst sein (Anspach, 2018). Die Poren des Filtermediums besitzen teilweise einen größeren Durchmesser als die zurückzuhaltenden Partikel, wodurch es zu Beginn der Filtration vorwiegend bei kleineren Partikeln zu einem Filterdurchbruch kommen kann. Durch den Rückhalt von größeren Partikeln und Partikeladhäsion wird jedoch eine Brückenschicht über den Poren des Filtermediums aufgebaut. Mit der Zeit entsteht dadurch ein Filterkuchen auf der Oberfläche des Filtermediums, welcher als zusätzliche Filterschicht mit steigender Effizienz weiteren Partikelrückhalt ermöglicht (Abbildung 2.2A) (Tien, 2012; Anspach, 2018). Der Filterkuchen erhöht zunächst die Filterleistung, doch mit fortschreitender Filtrationszeit nimmt auch die Schichtdicke des Filterkuchens zu, was einen Strömungswiderstand für das Fluid darstellt. Durch diesen Druckverlust ergibt sich bei zunehmender Kuchenhöhe ein verminderter Filtratfluss durch den Filter (Abbildung 2.2B) (Stieß, 1997).

Erreicht der Filtratfluss dadurch einen Wert unter einem akzeptablen Level wird die Filtration gestoppt und die Oberfläche des Filtermediums vom Filterkuchen befreit sowie gereinigt und ggf. ein neuer Filtrationszyklus durchgeführt (Sutherland und Chase, 2008). Die Kuchenfiltration findet oft Anwendung bei der Behandlung von Suspensionen mit hoher Partikelkonzentration (Tien, 2012).



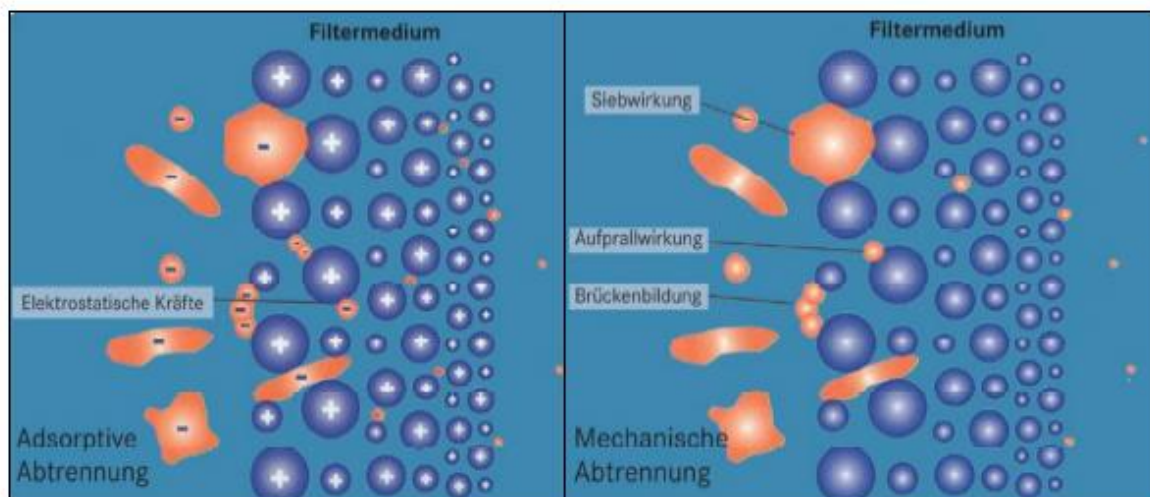
**Abbildung 2.2:** (A) Darstellung einer Kuchenfiltration inklusive Aufbau des Filterkuchens. (B) Abhängigkeit der Schichtdicke des Filterkuchens auf den spezifischen Filtratfluss (verändert nach Kraume, 2020).

### 2.1.1.2 Tiefenfiltration

Das Filtermedium bei einer Tiefenfiltration besitzt im Vergleich zur Kuchenfiltration eine relativ dicke Filterschicht und ähnelt einem engmaschigen Sieb mit zahllos verästelten Kanälen (Anspach, 2018). Die Suspension fließt durch das Filtermedium, wobei die Porengrößen des Filtermediums meist größer als die abzufiltrierenden Partikel sind. Dies führt dazu, dass die Partikel zum Teil tief in die Filterschicht eindringen können (Stieß, 1997). Im Inneren des Filtermediums findet eine Ablagerung der Partikel an der Oberfläche der Filtermittelbestandteile statt (Tien, 2012). Die Zurückhaltung erfolgt dabei durch mechanische Abtrennungsmechanismen, welche durch zusätzliche Adsorption (durch elektrostatische oder andere Wechselwirkungen) unterstützt wird

(Abbildung 2.3). Um eine lange Kontaktzeit für diese Wechselwirkungen zu ermöglichen, durchströmt die filtrierende Flüssigkeit den Filter relativ langsam (Stieß, 1997; Sutherland und Chase, 2008; Anspach, 2018). Die Tiefenfiltration besitzt ebenso eine begrenzte Kapazität. Eine Sättigungsbeladung oder ein maximaler Druckverlust kann aufgrund von verblockten Poren des Filtermediums mit abzufiltrierenden Partikeln erreicht werden. Daraufhin muss der Filter gewechselt oder durch Rückspülen von den Partikeln befreit werden (Sutherland und Chase, 2008; Stieß, 1997).

Ebenso zu erwähnen ist, dass die Mechanismen der Kuchen- und Tiefenfiltration nicht ausschließlich unabhängig voneinander auftreten. Der während einer Kuchenfiltration aufgebaute Filterkuchen kann unter gegebenen Umständen als Tiefenfilter agieren. Andersherum können sich nach einer länger andauernden Tiefenfiltration Partikel am Eingang des Tiefenfilters ablagern und schließlich zu einem Aufbau eines Filterkuchens führen. Die Tiefenfiltration wird im Vergleich zur Kuchenfiltration zum Klären oder Sterilfiltrieren von Suspensionen mit geringer Partikelkonzentration eingesetzt (Tien, 2012). Sie kommt auch zur Sterilfiltration von Gasen zum Einsatz, zum Beispiel in der Zu- und Ableitung der Begasung von Bioreaktoren (Anspach, 2018).



**Abbildung 2.3: Illustration einer Kuchenfiltration.** Abgebildet sind die mechanischen (rechts) und adsorptiven (links) Abtrennungsmechanismen des Filtermediums (mts Apic, 2020).

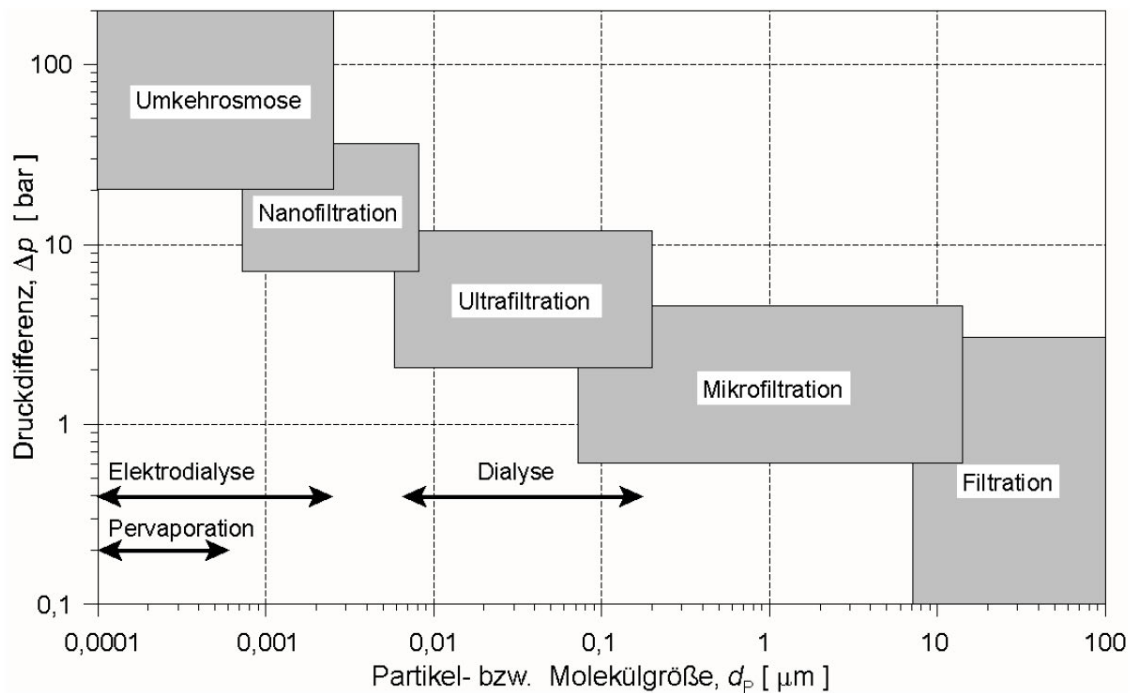
## 2.1.2 Membranverfahren und -filtration

Membranen sind flächenhafte Strukturen, die zwei Phasen voneinander trennen und dabei für mindestens eine Komponente einer sie berührenden Phase permeabel sind, andere dahingegen zurückhalten (Rautenbach und Melin, 2007; Ripperger, 1992). Demnach ist jede Membran im Allgemeinen auch ein Filter, wobei das Filtermedium in die Membran integriert ist. Dabei erlauben Membranen im Gegensatz zu klassischen Filtern die Trennung bis in den molekularen Bereich. Für die Biotechnologie bedeutende Membranverfahren, bei denen eine Druckdifferenz über die Membran die Triebkraft zur Durchströmung der Membran darstellt, werden hinsichtlich der Größe der abzutrennenden Teilchen in die folgenden Membranfiltrationsarten unterteilt (Ripperger, 1992; Anspach, 2018):

- Mikrofiltration (Abtrennung von Mikroorganismen und Zellen)
- Ultrafiltration (Trennung bzw. Konzentrierung von Proteinen)
- Nanofiltration (Entfernung von Substanzen mit  $M_r < 1000$  aus Wasser, z. B. zur Enthärtung)
- Umkehrosmose (Entfernung niedermolekularer Substanzen mit  $M_r < 100$  aus Wasser, z. B. zur Entsalzung)

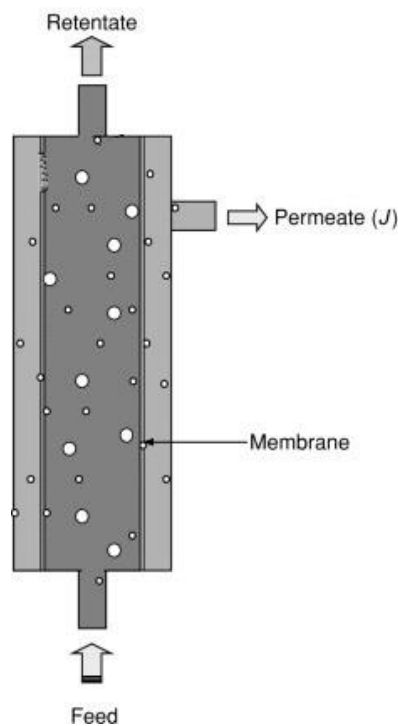
Bei einer Membranfiltration wird die zu filtrierende Flüssigkeit oft durch Konvektion an die Oberfläche einer Membran gebracht, wobei ein Teil des Gemisches die Membran durchdringt und ein anderer Teil zurückgehalten wird (Köly, 2010). Die Verfahren der Mikro- und Ultrafiltration werden in Kapitel 2.1.2.2 genauer erklärt. Ebenso relevante Membranverfahren mit anderen Triebkräften (z. B. Konzentrationsdifferenzen oder elektrische Potenzialdifferenzen) und ggf. anderen Trennprinzipien sind die Dialyse (Anpassung der Salzkonzentration und pH-Wert), Pervaporation (Abtrennung verdampfbarer Substanzen) oder die Elektrodialyse (Konzentrierung von Ionen und Entsalzung von Wasser) (Ripperger, 1992; Anspach, 2018). Der Abbildung 2.4 können die Arbeitsbereiche der genannten Membranverfahren entnommen werden.





**Abbildung 2.4: Arbeitsbereiche der Membranverfahren.** Zu erkennen sind die genannten Membranverfahren in Abhängigkeit von der Partikel- bzw. Molekülgröße der abzutrennenden Substanzen sowie bei druckgetriebenen Membranfiltrationen (graue Kästchen) von den typischen Druckdifferenzen. Die mit Pfeil dargestellten Membranverfahren sind durch andere Potentiale getrieben (Anspach, 2018).

Die Membran befindet sich während einer Membranfiltration meist innerhalb einer Modulkonstruktion. Da technische Membrananlagen normalerweise aus einer großen Anzahl an modular verbundenen identischen Bausteinen bestehen, wird hier der Begriff Modul verwendet (Rautenbach und Melin, 2007). In Kapitel 2.1.2.2.2 wird näher auf verfügbare Modulkonstruktionen eingegangen. Auch bei Membranfiltrationen können zwei grundsätzliche Betriebsarten unterschieden werden. Die statische Filtration (Dead-End) und die dynamische Filtration (Tangentialflussfiltration) (Ripperger, 1992). Im dynamischen Betrieb verfügt das Modul über mindestens einen Eingang für das zu filtrierende Fluid, wobei der eintretende Stoffstrom als „Feed“ bezeichnet wird (s. Abbildung 2.5). Zudem besitzt das Modul zwei Ausgänge, einen für die Membran passierenden Komponenten, das „Permeat“ und einen für die zurückgehaltenen Komponenten, das „Retentat“. Besitzt das Membranmodul dahingegen nur einen Ausgang, wird von einer Dead-End oder statischen Filtration gesprochen (Rautenbach und Melin, 2007; Anspach, 2018). Weitere Unterschiede dieser Betriebsarten werden im folgenden Kapitel genauer erläutert.



**Abbildung 2.5: Modulkonstruktion einer dynamischen Membranfiltration.** Zu sehen sind die Ein- bzw. Ausgänge des Moduls für Feed, Retentat und Permeat (verändert nach Lozano, 2003).

Der Volumenfluss, welcher die Membran durchströmt, wird meist auf eine Zeiteinheit und die durchströmte Membranfläche bezogen und als Flux (oder spezifische Permeatfluss) mit der Einheit  $\frac{L}{m^2 \cdot h}$  (auch LMH genannt) definiert (Anspach, 2018). Der Flux-Wert kann mit der folgenden Formel berechnet werden (Nikolay *et al.*, 2020).

$$J = \frac{\dot{V}_P}{A} \quad \text{mit} \quad \dot{V}_P = \frac{V_{P2} - V_{P1}}{t_2 - t_1} \quad (2.1)$$

Dabei stellt  $J$  den spezifischen Permeatfluss,  $\dot{V}_P$  den Permeat-Volumenstrom in  $\frac{L}{h}$  und  $A$  die Filterfläche der verwendeten Membran in  $m^2$  dar. Der Permeat-Volumenstrom wird durch die Ermittlung der Permeat-Volumendifferenz über einen Zeitabschnitt berechnet.

Wie bereits beschrieben stellt bei den Membranfiltrationen die Druckdifferenz über die Membran die Triebkraft für die Durchströmung der Membran dar. Dieser sogenannte Transmembrandruck (TMP) kann nach Ohlrogge und Ebert (2006) durch die folgende Formel berechnet werden.

$$TMP = \left( \frac{p_{Feed} + p_{Retentat}}{2} \right) - p_{Permeat} \quad (2.2)$$

Dabei beschreibt  $p_{Feed}$  den Druck am Moduleingang,  $p_{Retentat}$  den Druck am retentatseitigen Modulausgang und  $p_{Permeat}$  den Druck am permeatseitigem Modulausgang.

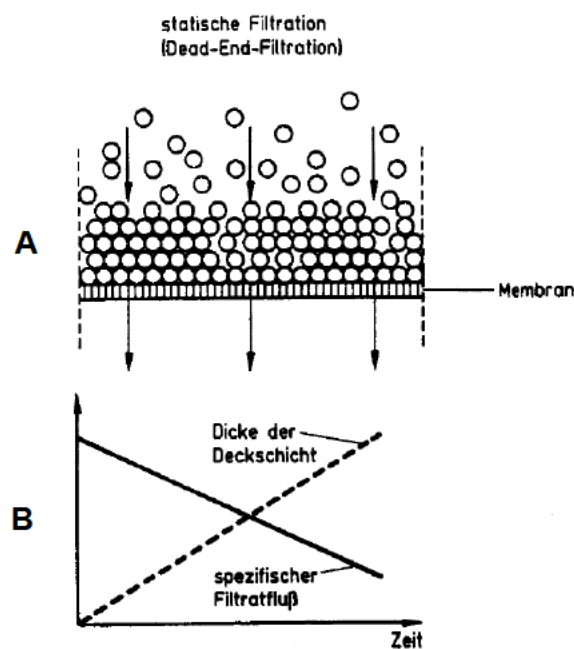
### 2.1.2.1 Arten der Prozessführung

Wie bereits erwähnt, wird grundsätzlich zwischen einer Dead-End-Filtration (statisch) und einer Tangentialflussfiltration (dynamisch) unterschieden werden.

#### 2.1.2.1.1 Statische Filtration (Dead-End-Filtration)

Im statischen Betrieb wird die Membran orthogonal von der zu filtrierenden Flüssigkeit durchströmt. Dabei gibt es nur einen Ausgang für die Membran passierenden Komponenten. Wie bei der Kuchenfiltration lagern sich alle zurückgehaltenen Teilchen auf der Oberfläche der Membran ab, sodass mit der Zeit die Dicke der abgetrennten Schicht anwächst (Abbildung 2.6A) (Ripperger, 1992; Rautenbach und Melin, 2007; Anspach, 2018). Bei Membranfiltrationen wird diese Schicht nicht Filterkuchen, sondern Deckschicht genannt. Diese Deckschicht bildet einen zunehmenden Strömungswiderstand, sodass auch bei hohen Drücken die Deckschicht nur noch von einer geringen Flüssigkeitsmenge durchströmt wird, es kommt zur Abnahme des Permeatflusses (Abbildung 2.6B). Wie schon bei der Kuchenfiltration muss dann die Filtration nach solch einem Filtrationsintervall abgebrochen werden und die auf der Membran abgelagerte Deckschicht entfernt oder die Membran gewechselt werden

(Ripperger, 1992; Rautenbach und Melin, 2007). Eine Dead-End-Filtration zählt demnach zu einem diskontinuierlichen Verfahren (Rautenbach und Melin, 2007). Aus diesem Grund finden Dead-End Membranfiltrationen meist nur Verwendung mit Suspensionen mit niedriger Partikelkonzentration, um dennoch wirtschaftliche Filtrationsintervalle zu ermöglichen. Überwiegend wird dieses Verfahren deshalb für End- oder Sicherheitsfiltrationen (Sterilfiltration) verwendet (Ripperger, 1992).

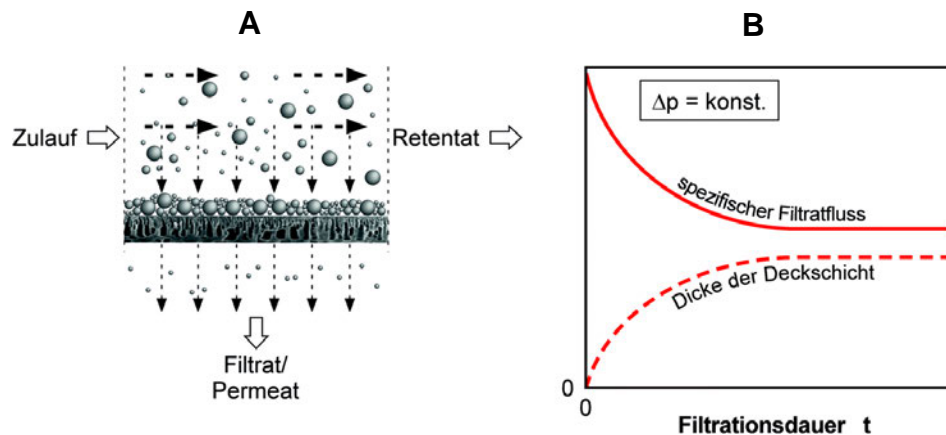


**Abbildung 2.6:** (A) Darstellung einer statischen Membranfiltration inklusive Aufbau der Deckschicht. (B) Abhängigkeit der Schichtdicke auf den spezifischen Permeatfluss (verändert nach Ripperger, 1992).

### **2.1.2.1.2 Dynamische Filtration (Tangentialflussfiltration)**

Die dynamischen Filtrationsverfahren realisieren im Gegensatz zur Dead-End-Filtration eine ständige tangentiale Überströmung der Membran auf der Feedseite (Rautenbach und Melin, 2007). Deshalb wird diese Prozessführung auch Tangentialflussfiltration (oder auch Querstromfiltration und Cross-Flow-Filtration) genannt. In solch einem Modul können zwei Hauptströme unterschieden werden, deren Richtungen orthogonal zueinanderstehen: die eben genannte Überströmung tangential zur Membran und den Permeatstrom durch die Membran (Abbildung 2.7A) (Ripperger, 1992). Auch bei der Tangentialflussfiltration kommt es während des Filtrationsprozesses zur Partikel-Ablagerung an der Membran. Diese an der Membran abgelagerten Partikel werden zum Teil durch die Membranüberströmung aufgrund von Scher- und Auftriebskräften zurück in die Kernströmung geführt oder bereits am Abscheiden gehindert (Ripperger, 1992; Rautenbach und Melin, 2007). Diese Funktionsweise erlaubt nach einer Einlaufphase ein stationäres Verhalten des Systems, da sich das Ablagern und Zurückführen der Partikel daraufhin in einem Gleichgewicht befinden. Dies ermöglicht eine konstante Deckschichtdicke und damit einen höheren und stationären spezifischen Permeatfluss (Abbildung 2.7B) (Rautenbach und Melin, 2007).

Mit dieser Prozessführung lassen sich im Vergleich zur Dead-End-Filtration Filtrationsverfahren länger und mit weniger Unterbrechungen betreiben. Dies ermöglicht das Ersetzen von mehrstufigen Filtrationsprozessen sowie die Membranen länger nutzbar zu halten (Ripperger, 1992). Ein weiterer Vorteil stellt die geschlossene Apparateausführung mit aseptischer Betriebsweise des Prozesses dar, insbesondere bei Verfahren mit rekombinanten oder pathogenen Mikroorganismen (Ripperger, 1992; Anspach, 2018). Ein Nachteil der dynamischen Fahrweise ist im Vergleich zur Dead-End-Filtration der zusätzliche Energieaufwand für die ständige tangentiale Überströmung der Membran (Rautenbach und Melin, 2007).



**Abbildung 2.7:** (A) Darstellung einer dynamischen Membranfiltration inklusive Aufbau der Deckschicht. (B) Abhängigkeit der Schichtdicke auf den spezifischen Permeatfluss (Kraume, 2020).

In dieser Masterthesis wird eine Tangentialflussfiltrationsanlage entwickelt, mit der Mikro- und Ultrafiltrationen in dynamischer Prozessführung durchgeführt werden können. Diese Filtrationsverfahren werden im folgenden Kapitel näher erläutert.

### 2.1.2.2 Mikrofiltration und Ultrafiltration

Bei der Mikro- und Ultrafiltration wird das Zurückhalten der Substanzen nach dem Prinzip des Siebeffektes und somit nach der Teilchengröße ermöglicht (Singh und Purkait, 2019). Dabei bestimmt größtenteils die mittlere Porengröße der Membran die Trennwirkung der Filtration (Gasper *et al.*, 2000). Neben den klassischen Filtrationen (2.1.1 Klassische Filtration) haben insbesondere die Mikrofiltration und die Ultrafiltration große Bedeutung in der Biotechnologie erlangt (Anspach, 2018). Bezogen auf die Größe der abzutrennenden Partikel bzw. Moleküle schließen diese Verfahren die Lücke zwischen der Umkehrosmose und der Nanofiltration auf der einen Seite und der klassischen Filtration auf der anderen Seite (s. Abbildung 2.4). Die Trennbereiche der beiden Verfahren können sich zum Teil überschneiden, welches eine klare Trennung nicht immer ermöglicht (Rautenbach und Melin, 2007).

Im Allgemeinen liegt die Trenngrenze, welche approximativ durch die abzutrennenden Partikel angegeben wird, bei der Mikrofiltration etwa bei 0,08 bis 10  $\mu\text{m}$ . Der Arbeitsbereich liegt bei einer transmembranen Druckdifferenz von 0,3 bis 3 bar (Ripperger, 1992; Rautenbach und Melin, 2007). Die Mikrofiltration realisiert die Abtrennung suspendierter kolloidaler Partikel aus Flüssigkeiten oder Gasen, wobei es sich dabei in der Biotechnologie vornehmlich um Mikroorganismen, Zellen oder Zellbruchstücken handelt (Ripperger, 1992). In den folgenden Gebieten kommt die Mikrofiltration zum Einsatz: Steril- und Entkeimungsfiltration, Zellernte, Zellabtrennung, Waschen von Zellen, Abtrennung von Zelltrümmern nach einem Zellaufschluss, Abtrennen von Proteinniederschlägen nach der Fällung (Anspach, 2018). Dabei kann sich das gewünschte Produkt im Permeat aber auch im Retentat befinden. Ebenso kann das Aufkonzentrieren einer Zellsuspension das Ziel einer Mikrofiltration darstellen.

Dahingegen liegt die Trenngrenze der Ultrafiltration im Allgemeinen bei etwa 0,001 bis 0,1  $\mu\text{m}$  und der Arbeitsbereich bei einer transmembranen Druckdifferenz von 0,5 bis 10 bar. In Ultrafiltrationsanlagen können ebenso kleine Partikel und Kolloide, zu dem aber auch gelöste niedermolekulare Stoffe und Makromoleküle mit Durchmessern zwischen 5 und 500 nm abgetrennt werden (Gasper *et al.*, 2000; Rautenbach und Melin, 2007). Der Übergang zur Mikrofiltration ist dabei fließend (s. Abbildung 2.4). Zum einen können mit Hilfe einer Ultrafiltration, die eben genannten abzutrennenden Stoffe durch Wahl einer geeigneten Porengrößen der UF-Membran fraktioniert werden. Das Molekulargewicht der zu fraktionierenden Stoffe muss sich dabei mindestens um eine Größenordnung unterscheiden, um ein gutes Aufteilungsergebnis zu erzielen (Ripperger, 1992; Rautenbach und Melin, 2007; Anspach, 2018). Zum anderen kann durch Entfernen des Lösungsmittels bei Zurückhalten des gelösten Stoffes eine Konzentrierung der Lösung stattfinden (Ripperger, 1992; Anspach, 2018). Einsatzgebiete in der Biotechnologie stellen damit z. B. die Isolierung und Konzentrierung von Enzymen oder auch die Reinigung und Konzentrierung von therapeutischen Produkten dar (Gasper *et al.*, 2000).

In der Biotechnologie wird die Tangentialflussfiltration (Querstrom) vielfältig als Alternative für Fest-Flüssig Trennungen angewandt. Die Mikrofiltration steht z. B. im Bereich der Zellernte und Zellabtrennung in Konkurrenz zur Zentrifugation. Die Ultrafiltration dahingegen vor allem zur Fällung und Eindampfung (Ripperger, 1992; Rautenbach und Melin, 2007; Anspach, 2018). Ein grundlegendes Problem dieser

druckgetriebenen Membranfiltrationen stellt die Reduktion des spezifischen Permeatflusses im Verlauf der Filtrationszeit dar. Dieses Problem wird im Kapitel „Deckschichtbildung und Fouling“ beschrieben und erklärt.

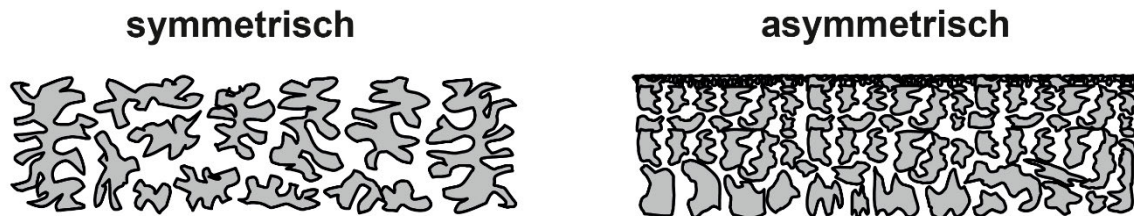
### **2.1.2.2.1 Membranen**

In der Membranfiltration angewendete Membranen werden hinsichtlich ihrer Herkunft, Morphologie und Struktur klassifiziert. Zunächst werden natürlich vorkommende biologische Membranen von den vermehrt für technische Zwecke eingesetzten synthetischen Membranen unterschieden. Synthetische Membranen werden weiterhin in feste und flüssige Membranen unterteilt. Feste synthetische Membranen können organisch aus Polymeren oder anorganisch aus zum Beispiel Keramik bestehen. Letztere wurden erst in den letzten Jahren vermehrt angeboten und besitzen noch hohe Materialpreise, weshalb organische Polymermembranen trotz niedrigerer chemischer und Temperaturbeständigkeit im Allgemeinen größere Bedeutung erlangt haben. Flüssige Membranen haben sich bislang nur zu Forschungszwecken etabliert, da diese sich zurzeit noch in der Entwicklung befinden.

Die Morphologie und Struktur der Membranen besitzt großen Einfluss auf den Trennmechanismus und dementsprechend auch auf die Anwendung der Membran. Es wird dabei zwischen porösen und nicht porösen „dichte“ Membranen mit einer aktiven Schicht unterschieden. Bei porösen Membranen (auch Porenmembranen genannt) ist meist der konvektive Transport durch die Poren der Membran dominierend, wobei hier oft ein Druckgradient zwischen Feed- und Permeatseite die Triebkraft darstellt. Bei dichten Membranen (auch Lösungs-Diffusions-Membranen genannt) kommt es auf die intrinsische Eigenschaft des Membranmaterials an, da hier der diffusive Transport von in der Membran löslichen Komponenten als Transportmechanismus dominiert. Dabei stellen meist elektrische oder chemische Potenzialdifferenzen die Triebkraft der Filtration dar (Rautenbach und Melin, 2007). Beide Membranarten werden zusätzlich aufgrund ihrer Herstellungsart zwischen symmetrischen und asymmetrischen Membranen unterschieden. Asymmetrische Strukturen besitzen den Vorteil eines geringeren spezifischen Strömungswiderstandes der Membran und damit eines höheren Permeatflusses, da hier die tatsächliche selektive Schicht sehr dünn gehalten wird und



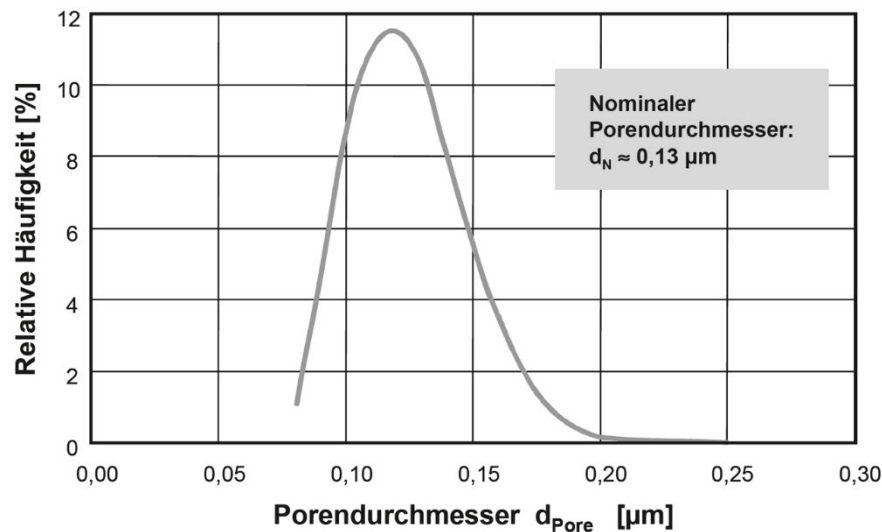
von einer grobporösen Stützschrift getragen wird (Abbildung 2.8) (Rautenbach und Melin, 2007; Anspach, 2018).



**Abbildung 2.8:** Schematische Darstellung einer symmetrischen und einer asymmetrischen Membran (verändert nach Rautenbach und Melin, 2007).

Wohingegen bei Membranverfahren wie die Elektrodialyse, die Umkehrosmose oder die Pervaporation nicht poröse dichte Membranen dominieren, finden poröse Membranen meist ihre Anwendung in den hier näher erklärten Verfahren der Mikrofiltration und Ultrafiltration (Rautenbach und Melin, 2007).

Die Porengröße von Mikrofiltrationsmembranen liegt in dem Bereich von ungefähr  $0,08\ \mu\text{m}$  bis  $10\ \mu\text{m}$ . Dabei ist es wichtig zu erwähnen, dass es keine absolute Trenngrenze gibt, da die Poren einer Membran nicht eine definierte Größe besitzen und sich so eine Porengrößenverteilung (Abbildung 2.9) mit einem mehr oder weniger breitem Spektrum ergibt. Als Eigenschaft wird ein nominaler Porendurchmesser einer Membran angegeben. Dieser entspricht dem Durchmesser eines charakteristischen Partikels, welcher durch die Membran zu 95-98 % zurückgehalten wird. Bei der Mikrofiltration werden oft symmetrische Membranstrukturen verwendet. Grund hierfür ist, dass aufgrund der Porengröße der Strömungswiderstand der Mikrofiltrationsmembran bereits relativ gering ist. Gegenüber dem Widerstand der sich auszubildenden Deckschicht auf der Membranoberfläche ist dieser zu vernachlässigen, weshalb von asymmetrischen Membranstrukturen abgesehen wird. Werkstoffe wie Polypropylen und PTFE (Polytetrafluorethylen), aber auch Polyamid oder Polysulfon werden häufig zur Mikrofiltrationsmembranherstellung genutzt (Rautenbach und Melin, 2007).



**Abbildung 2.9: Porengrößenverteilung einer Mikrofiltrationsmembran mit einem nominalen Porendurchmesser von 0,13  $\mu\text{m}$**  (Rautenbach und Melin, 2007).


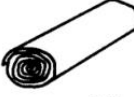


Auch Ultrafiltrationsmembranen besitzen keine einheitliche Porengröße und weisen somit eine Porengrößenverteilung auf. Da deshalb keine absolute Trenngrenze genannt werden kann, wird als charakteristische Größe das Molekulargewicht einer zu 90 % zurückgehaltenen Komponente angegeben. Dieser Wert wird *Molecular Weight Cut-off* oder kurz MWCO genannt (Rautenbach und Melin, 2007). Um während einer Ultrafiltration Produktverluste zu verringern und so eine möglichst quantitative Abtrennung zu realisieren, sollte der MWCO der Membran deutlich unter dem Molekulargewicht des aufzukonzentrierenden Produktes liegen (Anspach, 2018). Ultrafiltrationsmembranen besitzen typische molekulare Trenngrenzen im Bereich von ca. 1000 bis 100.000 Da, wobei als Membranmaterial Polyvinylidenfluorid (PVDF) und Polyethersulfon (PES) zu betonen sind. Letzteres besitzt eine deutlich engere Porengrößenverteilung als das bei Mikrofiltrationsmembranen erwähnte Polysulfon (Rautenbach und Melin, 2007). Im Gegensatz zu Mikrofiltrationsmembranen werden für Ultrafiltrationen asymmetrische Membranen mit einer grobporösen Stützschiicht und einer feinporösen trennaktiven Schicht verwendet. Dies realisiert einen möglichst hohen Permeatfluss und gewährleistet auch bei den hohen Drücken der Ultrafiltration Stabilität (Rautenbach und Melin, 2007).

### 2.1.2.2 Membrankonstruktionen

In jeder Membrananlage befindet sich die Membran in einer Modulkonstruktion. Hier findet mit Hilfe der Membran die Aufteilung des Feedstromes in den Retentat- und Permeatstrom statt. Module müssen gewisse Anforderungen erfüllen, um wirtschaftliche Filtrationsprozesse zu ermöglichen. Dabei sind die Schwerpunkte der Anforderungen abhängig von der Anwendung. Im Allgemeinen sollten folgende Anforderungen bei der Modulentwicklung berücksichtigt werden (Rautenbach und Melin, 2007):

- gleichmäßige Überströmung der Membran
- mechanische, chemische und thermische Stabilität
- hohe Packungsdichte
- gute Reinigungsmöglichkeit
- geringe Druckverluste
- hohe Feststoffbeladbarkeit

Membranmodule für die Tangentialflussfiltration können in 2 Bauklassen mit jeweils 3 Bauarten unterschieden werden (Rautenbach und Melin, 2007). Module mit Flachmembranen beinhalten Plattenmodule, Wickelmodule oder Module mit plissierter Membran. Rohrförmige Membranmodule werden in Rohrmodule, Kapillarmodule und Hohlfasermodule unterteilt (Anspach, 2018). In Abbildung 2.10 sind die Modularten zusammen mit einer schematischen Darstellung und ihrer typischen Membranfläche pro Bauvolumen aufgelistet.

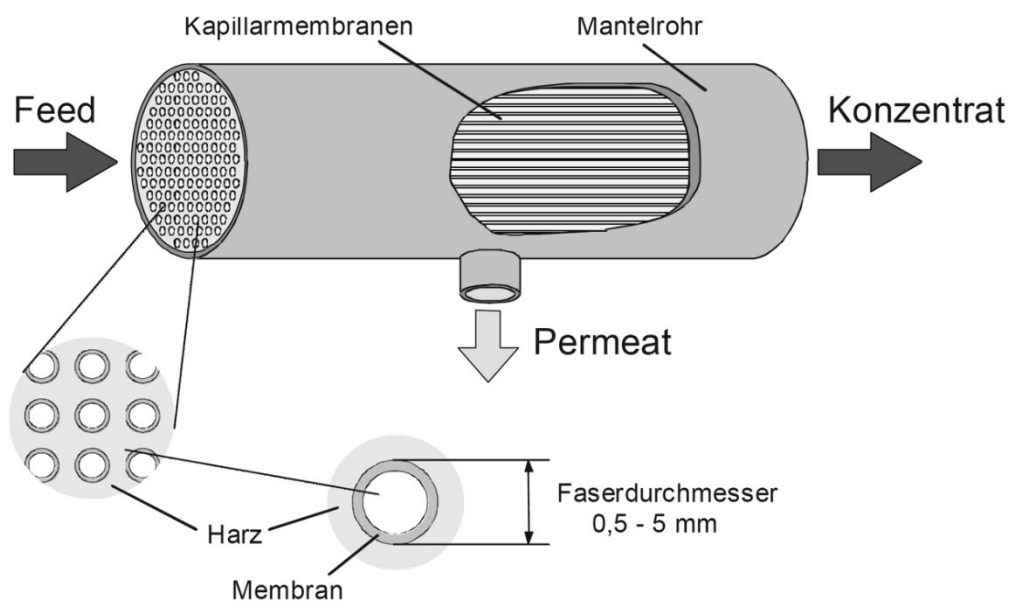
Module mit Flachmembranen		Membranflächen pro Bauvolumen [ $\text{m}^2 \text{m}^{-3}$ ]
Plattenmodul		200 bis 800
Wickelmodul		600 bis 1000
Modul mit plissierter Membran		400 bis 1400
Module mit rohrförmigen Membranen		
Rohrmodul ( $d_i > 4 \text{ mm}$ )		80 bis 500
Kapillarmodul ( $d_i = 0,5\text{-}4 \text{ mm}$ )		500 bis 4000
Hohlfasermodul ( $d_i < 0,5 \text{ mm}$ )		> 4000

**Abbildung 2.10: Bauklassen und -arten von Membrankonstruktionen.** Zusätzlich abgebildet sind schematische Darstellungen der Module sowie typische Packungsdichten (Anspach, 2018).

Hohlfasermodule besitzen noch kleinere Faserinnendurchmesser als Kapillarmodule, welche ebenso deutlich geringere Innendurchmesser aufweisen als Rohrmodule. In der Literatur zur Mikro- und Ultrafiltration werden die Begriffe Kapillar- und Hohlfasermodul jedoch meist austauschbar verwendet, weshalb auch in dieser Masterthesis im weiteren Verlauf ausschließlich von Hohlfasermodulen gesprochen wird. Innerhalb der Konzentrationsversuche aus Kapitel 3.2.3 wird ein Hohlfasermodul verwendet, weshalb diese Modulart nun näher erklärt wird.

Hohlfasermodule bestehen aus vielen Membranschläuchen, auch Hohlfasern genannt, welche parallel innerhalb eines Mantelrohrs (Gehäuse) aus Kunststoff angeordnet sind und an beiden Enden zur Abdichtung in einer Kopfplatte mit Harz verklebt sind (Abbildung 2.11) (Rautenbach und Melin, 2007; Anspach, 2018). Die Parallelschaltung dieses Aufbaus ermöglicht eine einfache Skalierung der Module. Meist werden die Module feedseitig innen angeströmt, d.h. die zu filtrierende Flüssigkeit wird durch den Moduleingang in die Hohlfasern geleitet. Das Permeat durchdringt dabei die Membran von innen nach außen (Anspach, 2018). Das Retentat wird durch die Hohlfasern zum Modulausgang geführt. Das anstehende Permeat sammelt sich im Außenraum des Membranbündels und verlässt das Modul über einen Permeatauslass (Rautenbach und Melin, 2007).

Im Gegensatz zu einem Rohrmodul werden die Hohlfasern nicht mechanisch gestützt, die Membran ist demnach selbsttragend (Anspach, 2018). Darüber hinaus zeichnen sich Hohlfasermodule durch ihre hohen Packungsdichten aus, welche aufgrund der kleinen Durchmesser der Hohlfasern zu erreichen sind (Goedecke, 2006).



**Abbildung 2.11: Struktureller Aufbau und Betriebsweise von Kapillar- und Hohlfasermodulen.** Dargestellt sind die Flussrichtungen der Stoffströme sowie die einzelnen Bestandteile des Moduls (Rautenbach und Melin, 2007).

### 2.1.2.2.3 Deckschichtbildung und Fouling

Deckschichten, speziell kompakte oder sehr dicke Deckschichten, können die Filterleistung und den Permeatfluss erheblich verringern, welches das zentrale Problem dieser Filtrationsarten darstellt. Die Prozessführung in der Tangentialflussfiltration wirkt, wie in Kapitel 2.1.2.1.2 beschrieben, durch eine tangentiale Membranüberströmung dagegen. Wie auch bereits erwähnt bildet sich aber auch unter dynamischen Prozessbedingungen eine gewisse Deckschicht auf der Membranoberfläche aus zurückgehaltenen Partikeln aus (Rautenbach und Melin, 2007; Anspach, 2018). Kann solch eine Deckschichtbildung durch zum Beispiel Erhöhung der Überströmungsgeschwindigkeit wieder abgetragen werden, spricht man von einer reversiblen Deckschichtbildung (Rautenbach und Melin, 2007).

Bei der Mikrofiltration kommt es meist in einer Einlaufphase zur Ausbildung einer reversiblen Deckschicht durch z. B. Zellen oder Zellbruchstücken, wobei durch den dadurch zusätzlichen Strömungswiderstand der spezifische Permeatfluss zunächst deutlich sinkt. Durch Erreichen der stationären Betriebsphase wird idealerweise die Deckschicht und somit der Permeatfluss konstant gehalten (Anspach, 2018; Rautenbach und Melin, 2007). Diese Deckschicht besitzt zudem eine zusätzliche Filtrationswirkung. Kleinere, meist gelöste Substanzen (Proteine oder DNA-Moleküle) können von größeren Partikeln aus der Deckschicht adsorbiert werden. Dies resultiert möglicherweise in ein Zurückhalten von Teilchen, deren Durchmesser kleiner als der Porendurchmesser ist und bei denen theoretisch kein Rückhalt zu erwarten ist. Im Fall eines gelösten Produktes kann es dadurch zu Produktverlust kommen. Die Ausbildung einer Deckschicht hat somit auch immer eine Verschiebung der effektiven Trenngrenze in Richtung kleinerer Partikeldurchmesser zu Folge, was es bei der Prozessplanung zu berücksichtigen gilt. Beim Rücktransport der Partikel aus der Deckschicht dominieren bei der Mikrofiltration die hydrodynamischen Effekte. Einen entscheidenden Einfluss dabei hat die Kraft der Membranüberströmung in unmittelbarer Nähe der Deckschicht. Die dort herrschende Wandschubspannung stellt die wesentliche Kraft beim Herauslösen von Partikeln aus der Deckschicht dar. Eine Erhöhung der feedseitigen Überströmungsgeschwindigkeit (in Hohlfasermodulen als Schergeschwindigkeit angegeben) führt zur Erhöhung der Wandschubspannung und damit zu einem stärkeren Abtragen der Deckschicht, welches wiederum eine dauerhafte Permeatflussstei-

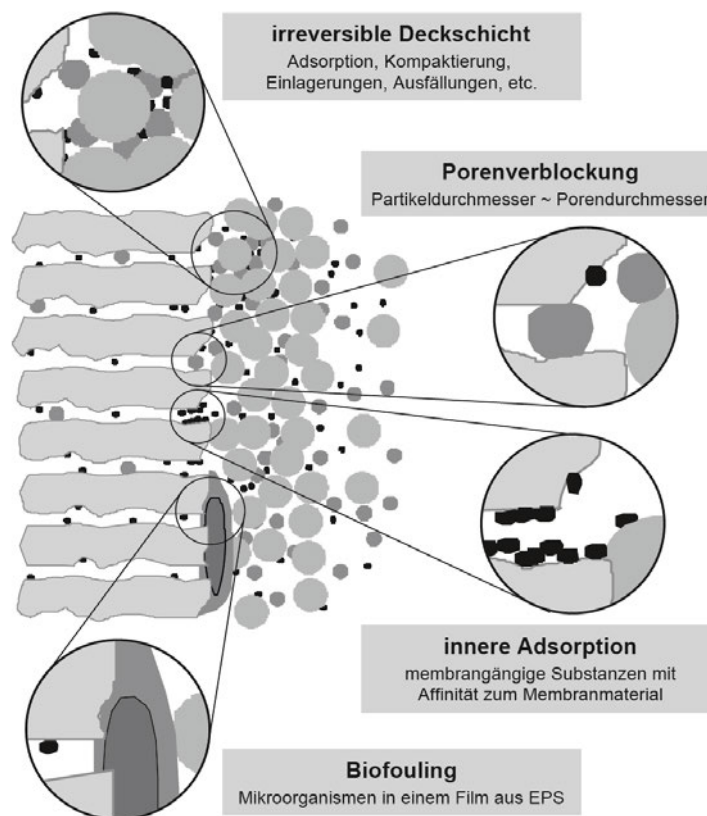
gerung erzielen kann (Rautenbach und Melin, 2007). Eine Erhöhung der prozessbeeinflussenden transmembranen Druckdifferenz hat einen gegenteiligen Effekt. Dies hätte lediglich eine dickere und kompaktere Deckschicht zu Folge, welches den Permeatfluss stark senken würde. Mikrotangentialflussfiltrationen zeigen demnach stabile Permeatflüsse über eine lange Zeitspanne oft bei niedrigen Transmembrandrücken (Anspach, 2018).

Neben nahezu allen Anwendungen in der Mikrofiltration findet man solch eine Deckschichtbildung auch in den meisten Anwendungsfällen der Ultrafiltration wieder (Rautenbach und Melin, 2007). Dabei wird bei der Ultrafiltration der Begriff Grenzschicht verwendet. Aufgrund dem auf der Retentatseite ausgeübten Druck kommt es neben Feststoffen bei der Ultrafiltration vor allem zu einer Anreicherung von zurückgehaltenen gelösten Substanzen (z. B. Proteine) auf der Oberfläche der Membran (Anspach, 2018). Die Ausbildung eines solchen Konzentrationsprofils wird als Konzentrationspolarisation bezeichnet (Rautenbach und Melin, 2007). Dieser Effekt kann das Bilden einer Gelschicht auf der Membranoberfläche hervorrufen. Diese Gelschicht kann sich bei weiterer Filtration unter Druck zunehmend aufbauen, sodass das hochkonzentrierte Gel wie bei der Deckschicht in der Mikrofiltration einen starken Strömungswiderstand bildet und der Permeatfluss deutlich sinkt (Anspach, 2018). Der Rücktransport der gelösten Substanzen der Gelschicht wird bei der Ultrafiltration durch diffusive Effekte entlang dem Konzentrationsgradienten dominiert (Rautenbach und Melin, 2007).

Im realen Betrieb der Mikro- und Ultrafiltration ist es möglich, dass auch nach der Einlaufphase der Filtration ein stetiges Abnehmen des spezifischen Permeatfluss festgestellt werden kann (Rautenbach und Melin, 2007). Solch eine Beeinträchtigung der Leistungsfähigkeit des Filtermoduls, die meist ihren Ursprung in Verschmutzungen oder Änderungen der Membraneigenschaft hat, wird als Fouling bezeichnet. Dabei muss erwähnt werden, dass die bereits beschriebenen Effekte der Konzentrationspolarisation (Grenzschicht) oder der Ausbildung einer reversible Deckschicht nicht dazu zählen (Rautenbach und Melin, 2007). Mögliche Ursachen und Mechanismen werden im Folgenden erklärt und sind in Abbildung 2.12 dargestellt.

Es kann durch Adhäsion und Wachstum von Mikroorganismen zu einem Biofilm auf der Oberfläche der Membran kommen, auch Biofouling genannt. Die Ausbildung einer irreversiblen Deckschicht ist ebenfalls eine mögliche Ursache. Meist ist dies eine

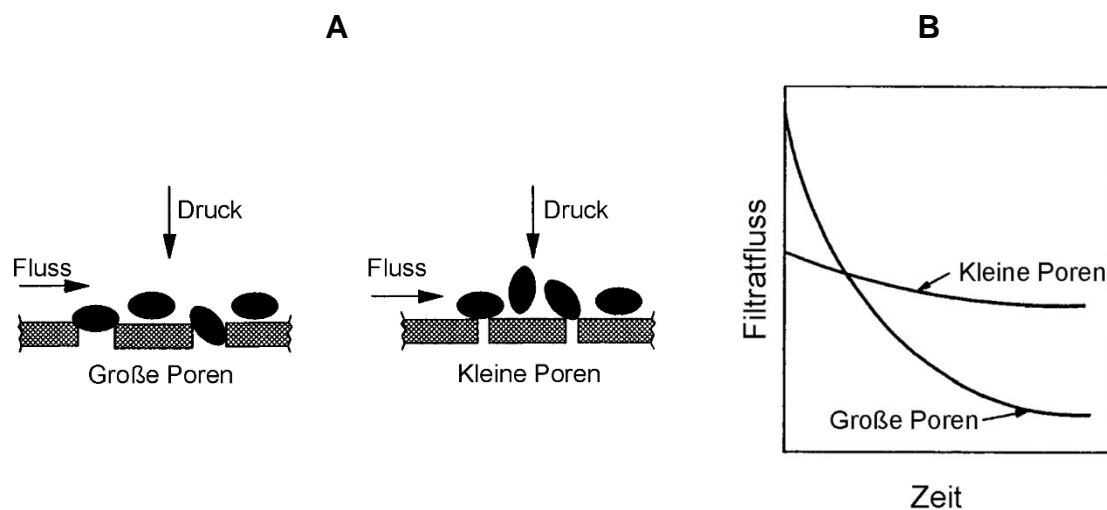
Schicht mit zurückgehaltenen Substanzen, welche durch Partikeladsorption und Wechselwirkungen sich nicht mehr von der Überströmung lösen lassen. Dazu beitragen können auch sehr feine Partikel oder durch die Konzentrationspolarisation ausgelöste Ausfällungen von gelösten Stoffen, welche sich an der Oberfläche der Membran ablagern. Ebenso kann die Adsorption von membrangängigen Makromolekülen im Poreninnenraum zu Porenverengungen führen, welches den Permeatfluss beeinträchtigt. Bei Poren, welche einen ähnlichen Durchmesser haben wie die abzutrennenden Partikel, kann es auch zur sterischen Porenverblockung kommen. Die Partikel lagern sich im Porenmund ab und blockieren den Permeatfluss durch die Pore (Rautenbach und Melin, 2007; Anspach, 2018).



**Abbildung 2.12: Vielfältige Ursachen des Fouling bei porösen Membranen.** Abgebildet sind die Mechanismen der irreversiblen Deckschichtbildung, der sterischen Porenverblockung, der inneren Adsorption und des Biofouling (Rautenbach und Melin, 2007).



Es hat sich herausgestellt, dass die Porengröße einen wichtigen Parameter in Bezug auf Fouling darstellt. Dabei zeigt sich, dass Membranen mit großen Porendurchmessern zu Beginn der Filtration zwar einen höheren Permeatfluss ermöglichen als Membranen mit kleinen Poren (Abbildung 2.13). Jedoch sinkt der Flux deutlich schneller ab und erreicht nach einiger Zeit sogar einen geringen Flux (Anspach, 2018). Dies ist auf den Mechanismus der sterischen Porenverblockung zurückzuführen. Kleinere Poren sind weniger anfällig für Ablagerungen im Poreninnenraum (Abbildung 2.13). Kleinere Porendurchmesser sind auch der Grund, warum Fouling durch Ablagerungen im Poreninnenraum in der Ultrafiltration seltener vorkommt als in der Mikrofiltration (Baker, 2004).



**Abbildung 2.13: (A) Mechanismus der sterischen Porenverblockung in Abhängigkeit von der Porengröße. (B) Der Permeatfluss in Abhängigkeit von der Filtrationszeit von großen und kleinen Poren (Anspach, 2018).**

Eine Möglichkeit, dem Fouling während des laufenden Prozesses entgegenzuwirken, ist das periodische Rückspülen der Membran (Anspach, 2018). Dieses Vorgehen reduziert die aufgebaute Deckschicht und verlängert dabei die Filtrationszyklen ohne chemische Reinigung (Ripperger, 1992; Rautenbach und Melin, 2007). Es hat sich gezeigt, dass sich bei der Ernte von Mikroorganismen der mittlere Permeatfluss dadurch um 30-50 % steigern lässt (Anspach, 2018).

#### 2.1.2.2.4 Membran Reinigung

Die Membran sowie auch die produktführenden Oberflächen des Moduls müssen anwendungsspezifisch in regelmäßigen Abständen von Verunreinigungen und Schmutz, aber auch von Mikroorganismen und Proteinen gereinigt werden (Anspach, 2018). Wichtige Parameter, welche die Effektivität einer Reinigungsprozedur beeinflusst sind Reinigungszeit, chemische Aktivität der Reinigungslösung, die Temperatur sowie die hydrodynamischen Verhältnisse an der Membran (Rautenbach und Melin, 2007). Im Allgemeinen können verwendete Reinigungschemikalien in vier Gruppen eingeteilt werden, wobei stets die Angaben des Membranherstellers zu beachten sind (Anspach, 2018):

- Laugen und Säuren
- Komplexbildner
- Oberflächenaktive Stoffe (Tenside)
- Enzyme

Zur Bewertung der Reinigungsprozedur wird der erzielbare Wasser-Flux unter bestimmten Prozessbedingungen (Druckdifferenz, Überströmungsgeschwindigkeit) nach dem Reinigungsschritt herangezogen (sog. Wasserwert). Werden mind. 60 % des initialen Wasserwertes unter denselben Bedingungen erreicht, gilt die Reinigung als befriedigend (Anspach, 2018).

#### 2.1.2.2.5 Filtrationsbetriebsweisen- und modi

Bei dem Verlauf einer Tangentialflussfiltration kann zwischen einer kontinuierlichen und der diskontinuierlichen Betriebsweise sowie der Diafiltration unterschieden werden.

In der diskontinuierlichen Betriebsweise (sog. Batch-Betrieb) wird eine Vorlage mit einem festen Startvolumen stetig filtriert. Hier verändern sich einzelne Betriebsparameter im Laufe der Filtrationszeit wie zum Beispiel das Feedvolumen und die Retentat Konzentration (Ripperger, 1992). Diese Betriebsweise wird in den Konzentrationsversuchen dieser Masterarbeit verwendet. Die kontinuierliche Betriebsweise wird *Feed- and Bleed-System* genannt. Grund dafür ist die kontinuierliche Produktzufuhr sowie

Permeat- und Retentatabfuhr. Hierbei stellen sich im Gegensatz zum Batch Verfahren nach einer gewissen Zeit stationäre Betriebsbedingungen ein (Ripperger, 1992). Des Weiteren können die folgenden Filtrationsmodi in einer Tangentialflussfiltrationsanlage durchgeführt werden: Eine Konzentrierung und eine Diafiltration.

Ist kein Zufuhrfluss in den Feedbehälter vorhanden, verringert sich durch die Filtration und des dabei abgeführten Lösungsmittels das Volumen im Feedbehälter. Zeitgleich werden bei einer Mikro- und Ultrafiltration biologische Substanzen (MF: Mikroorganismen, UF: Makromoleküle), welche durch die Membran zurückgehalten werden, im Retentatstrom (im Feedbehälter gesammelt) angereichert. Man spricht von einer Konzentrierung oder Aufkonzentrierung (Anspach, 2018). Durch die Aufkonzentrierung verändern sich die rheologischen Eigenschaften des Retentats, die Viskosität steigt an und die Suspension wird dickflüssiger. Dadurch kann ein höherer Druckabfall im Retentatkreislauf beobachtet werden sowie ein niedriger Permeatstrom. Wird ein zu geringer Flux oder ein zu hoher Druckabfall erreicht, wird die Filtration abgebrochen und das Retentat abgelassen. Um die Konzentrierung quantitativ zu erfassen, kann während des Filtrationsvorganges ein Konzentrationsfaktor ( $CF = \textit{Concentration Factor}$  oder  $VCR = \textit{Volume Concentration Ratio}$ ) berechnet werden (Anspach, 2018).

$$CF = \frac{V_{F,0}}{V_R} = \frac{V_{R,0}}{V_{F,0} - V_P} \quad (2.3)$$

Dabei stellt  $V_{F,0}$  das Startvolumen im Feedbehälter,  $V_R$  das jetzige Retentatvolumen und  $V_P$  das bislang filtrierte Permeatvolumen dar.

Eine Diafiltration ist ein Verfahren, bei dem gelöste Stoffe in der zu filtrierenden Flüssigkeit oder lösliche Komponenten der zurückgehaltenen Zielsubstanz, welche kleiner sind als die Membranporen, durch ein Lösungsmittel verdrängt und aus dem Retentat herausgespült werden. Dafür müssen das Lösungsmittel und die gelösten Stoffe ungehindert die Membran passieren können, wohingegen die Zielsubstanz vollständig zurückgehalten werden muss. Es wird bei diesem Filtrationsmodus oft von einem Waschen oder dem Umpuffern der Zielsubstanz gesprochen (Ripperger, 1992). Die membrangängigen Substanzen werden dabei am effizientesten entfernt, wenn das Lösungsmittel während der Filtration kontinuierlich entsprechend dem Permeatfluss

zugeführt wird. Genannt wird dieses Verfahren Diafiltration mit kontinuierlicher Lösungsmittelzufuhr (Anspach, 2018). Um das Volumen im Feedbehälter konstant zu halten, wird dafür meist eine Füllstandsregelung eingesetzt (2.2 Regelungstechnik). In diesem Modus bleibt die Konzentration der zurückgehaltenen Komponente konstant. Das zu diafiltrierende Waschvolumen wird als Vielfaches des Feed-Startvolumens angegeben und wird Diafiltrationsvolumen (kurz DV) genannt. Es kann durch die folgende Formel im laufenden Prozess ermittelt werden (Anspach, 2018).

$$DV = \frac{V_P}{V_{F,0}} \quad (2.4)$$

Dabei stellt  $V_{F,0}$  das Startvolumen im Feedbehälter zum Start der Diafiltration, und  $V_P$  das bislang filtrierte Permeatvolumen dar.

#### **2.1.2.2.6 Kommerziell erhältliche TFF-Anlagen**

In dieser Masterthesis wird eine Tangentialflussfiltrationsanlage im Labor Maßstab (eng: *Lab Scale*) als Benchtop-Format für Mikro-, Ultra- und Diafiltrationen inklusive einer Software zur Steuerung dieser Anlage entwickelt. Kommerziell erhältliche Tangentialflussfiltrations-Systeme für die genannten Anwendungen in der gleichen Größenordnung und dem gleichen Format werden von vielen verschiedenen Herstellern angeboten. Die bekanntesten Anbieter stellen dabei Cytiva (Marlborough, Vereinigte Staaten), Repligen (Waltham, Vereinigte Staaten) und Sartorius AG (Göttingen, Deutschland) dar.

Cytiva bietet mit dem System ÄKTA Flux™ eine semi-automatische Tangentialfluss-filtrationsanlage an. Das System erlaubt das Durchführen von Konzentrations- und Diafiltrationsprozessen sowie Zellernte und -klärung. Dabei können Hohlfasermodule, Plattenmodule sowie Membran Adsorber in die Anlage integriert werden. Es sind zwei Versionen des Systems erhältlich. Die Äkta Flux™ S besitzt einen 0,5 L Feedbehälter inklusive eines Magnetrührers und ist für Forschungszwecke und zum Filter-Screening ausgelegt. Die Äkta Flux™ 6 verfügt über einen 8 L Feedbehälter und wird für die Prozessentwicklung und zur Produktion im kleinen Maßstab angeboten. Endpunktkontrollen eines Filtrationsprozesses von Parametern wie Füllstand des Feedbehälters oder dem Transmembrandruck sowie eine automatische Messdatenerfassung ermöglichen einen unbewachten Betrieb der Anlage. Dieses Feature wird unterstützt durch ein Warnungs- und Alarm-Management-System, welches bei Erreichen vordefinierter Grenzwerte auditive und visuelle Warnungen produziert und schließlich die Feedpumpe stoppt. Eine Pumpe in der Permeatleitung ermöglicht die Regelung eines konstanten Permeatflusses. Ebenso ist die Regelung eines konstanten Filter-Eingangsdrukkes oder eines konstanten Druckverlustes über dem Modul durch Anpassung der Feedpumpenrate möglich. Darüber hinaus verfügt das System über eine *Cleaning in Place* (CIP) Funktion und neben der Gewichts- und Druckmessung über eine zusätzlich Temperaturmessung. Die Bedienung und Planung erfolgt dabei über eine Prozess- und Steuerungssoftware eines integrierten Computers. Steuerventile in der Retentat- sowie Permeatleitung zum Steuern des Transmembrandruckes sind nur manuell bedienbar. Ein Durchflussmesser zur Kontrolle der Flussrate ist nicht verbaut. Cytiva bietet für den großen Produktionsmaßstab voll-automatische Systeme an (Cytiva, 2021).



Abbildung 2.14: Die Äkta Flux™ S (links) und die Äkta Flux™ 6 (rechts) (Cytiva).

Repligen bietet das Tangentialflussfiltrations-System KrosFlo® KR2i für Hohlfasermodule und das System KrosFlo® FS-15 für Plattenmodule für Mikro- und Ultrafiltrationen im Konzentrations- und Diafiltrationsbetrieb an. Der Volumen Arbeitsbereich liegt bei diesen Anlagen bei 2 ml bis 15 L. Der Feedbehälter verfügt dabei nicht über einen integrierten Rührer. Das System ist modular zusammenbaubar und besteht nicht wie das Cytiva System aus einem festen Gehäuse. Neben vielen vorgefertigten Prozessmodi können komplexe Filtrationsprozesse vom Benutzer in der integrierten Software über einen LCD-Display geplant und ausgeführt werden. Eine Vielzahl von gemessenen Parametern wie Konzentrationsfaktor, Diafiltrationsvolumen, Leitfähigkeit, UV und Temperatur etc. können zur Endpunktkontrolle eines Filtrationsprozesses und zum Alarm-Management herangezogen werden. Diese Eigenschaft sowie eine automatische Echtzeit-Messdatenanzeige und -speicherung erlauben ebenso einen semi-automatischen unbewachten Betrieb. Eine zusätzliche Permeatpumpe ermöglicht die Regelung eines konstanten Permeatflusses. Automatisch ansteuerbare Ventile realisieren einen konstanten Transmembrandruck über die gesamte Prozesszeit. Das System verfügt neben den klassischen Filtrationsmodi wie Konzentrierung und Diafiltration über vorprogrammierte Sequenzen zum Reinigen und Spülen der Anlage und unter anderem auch über einen *Normalized Water Permeability* Modus (Wasserwertmessung). Manuelle Benutzer Aktion sind bei letzteren Modi aber notwendig. Das System besitzt wie das von Cytiva keinen Durchflussmesser (Repligen, 2021).



Abbildung 2.15: Das Tangentialflussfiltrations-System KrosFlo® KR2i (Repligen, 2021).

Die Sartorius AG bietet das System Sartoflow® Smart an. Diese semi-automatische Tangentialflussfiltrationsanlage besteht aus einem festen Gehäuse mit teils modularen Bestandteilen. Das System ist für die Ultrafiltration und Diafiltration optimiert, wobei im Gegensatz zu den anderen Herstellern nur Plattenmodule unterstützt werden. Es besitzt einen 1 L Feedbehälter mit integriertem Magnet-Rührer, in dem zusätzliche Parameter wie die Leitfähigkeit, der pH-Wert oder die Temperatur gemessen werden können. Wie schon die bereits beschriebenen Systeme können auch bei dieser Anlage die eben genannten Parameter oder klassische Parameter wie der Füllstand als vordefinierte Endpunkte von vorgefertigte (Konzentrierung, Diafiltration, Reinigung, Entleeren) oder selbst erstellten Prozesssequenzen verwendet werden. Die Endpunktkontrolle sowie ein integriertes Alarm-Management auf Basis dieser Parameter ermöglichen nach dem Start einen automatischen unbewachten Betrieb. Bei Filtrationsprozessen kann der Eingangsdruck durch die Feedpumpe sowie der Transmembrandruck durch das ansteuerbare Retentat- bzw. Permeatventil konstant geregelt werden. Zur Bedienung verfügt das System über einen integrierten Touchscreen. Dort erfolgt ebenso die automatische Messdatenerfassung und -speicherung. Wie bei den vorherigen Systemen ist kein Durchflussmesser eingebaut (Sartorius AG, 2021).

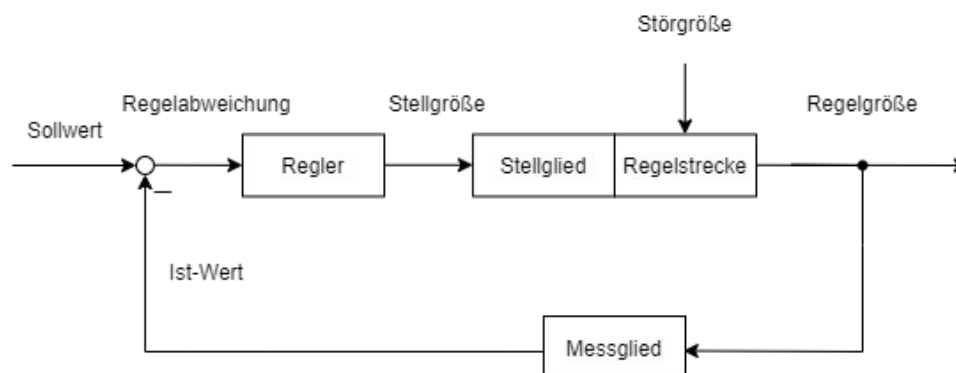


**Abbildung 2.16: Das System Sartoflow® Smart (Sartorius AG, 2021).**

## 2.2 Regelungstechnik

In diesem Kapitel werden die fundamentalen Begriffe der Regelungstechnik kurz erläutert und die Implementierung eines digitalen Reglers umrissen. Dies soll ein grundlegendes Verständnis für die Programmierung der Füllstandsregelung der TFF-Anlage bilden.

Das Regeln einer bestimmten Regelgröße (z. B. Füllstand) eines zu regelnden Systems (z. B. Feedbehälter einer TFF-Anlage) bedeutet das kontinuierliche Messen der Regelgröße und das fortlaufende Vergleichen mit einem vorgegebenen Sollwert. Ein Regler bestimmt aus dieser Regeldifferenz (auch Regelabweichung genannt) und einer Regler-Gleichung eine Stellgröße. Die Regler-Gleichung besteht abhängig vom Reglertyp aus einem Proportional-Anteil (P-Anteil), einem Integral-Anteil (I-Anteil) und/oder einem Differenzial-Anteil (D-Anteil). Die Stellgröße wirkt auf das zu regelnde System ein, genauer gesagt auf ein zwischengeschaltetes Stellglied, sodass trotz einer existierenden Störgröße (z. B. Ausfluss aus dem Feedbehälter) die Regeldifferenz minimiert wird und die Regelgröße sich dem Sollwert anpasst (Cornelissen, 2020). Die voneinander abhängigen Wirkungen der einzelnen Größen werden in einem sogenannten Regelkreis dargestellt (Abbildung 2.17).



**Abbildung 2.17: Blockbilddarstellung eines Regelkreises.** Dargestellt sind die einzelnen Komponenten und Größen eines Regelkreises sowie ihre Wirkung aufeinander.



Regler können analog oder digital realisiert werden. Da in dieser Masterthesis ein digitaler Regler entworfen wurde, wird in dem folgenden Absatz nur auf diese Regler-Art eingegangen.

Digitale Regler werden durch einen Digitalrechner umgesetzt. Dieser ermöglicht die Verarbeitung der anfallenden Messwerte, enthält den Regelalgorithmus und kann durch eine programmierbare Logik innerhalb eines PCs implementiert werden. Die notwendigen Signale werden dafür über eine Schnittstelle zwischen PC und Geräten übermittelt. Bei digitalen Reglern erfolgt die Signalverarbeitung zeitdiskret, was bedeutet, dass der Ist-Wert in konstanten zeitlichen Abständen gemessen wird. Die sogenannte Abtastrate stellt dabei die Frequenz dieser Messung dar (Svaricek, 2012). Bei der in dieser Masterthesis zu programmierenden Füllstandsregelung stellt der Feedbehälter der TFF-Anlage das auszuregelnde System da. Diese sogenannte Strecke des Regelkreises besitzt ein integrales Verhalten (I-Strecke). Im Allgemeinen bieten sich P- oder PI-Regler zum Regeln von I-Strecken an. Die Wahl viel hierbei auf einen PI-Regler, da der Integral-Anteil für das Eliminieren einer bleibenden Regelabweichung notwendig ist (Busch, 2012). Die allgemeine Regler-Gleichung eines PI-Reglers wird in der folgenden Formel abgebildet (Heinrich, 2021).

$$u(t) = K_P * e(t) + K_I * \int_0^t e(\tau) d\tau \quad (2.5)$$

Dabei stellt  $u(t)$  die Stellgröße,  $e(t)$  die Regelabweichung,  $K_P$  den Proportionalfaktor des Proportional-Anteils und  $K_I$  den Integrationsfaktor des Integral-Anteils dar. Die Regelabweichung wird durch die Subtraktion des Istwertes vom Sollwert berechnet. Damit der PC die zeitdiskreten Werte verarbeiten kann, muss die Regler-Gleichung umgeschrieben werden. Der Proportional-Anteil kann dabei problemlos übernommen werden. Für den Integrations-Anteil wird die Integration durch eine Summenbildung ersetzt. Dabei werden bekannte Methoden aus der numerischen Mathematik angewandt (Svaricek, 2012). Hierbei wurde sich für die numerische Integration mit Hilfe der Trapezmethode entschieden. Die folgende Formel zeigt die Summenberechnung.

$$S(t_2) = S(t_1) + (\Delta t) * \left(\frac{e_2 + e_1}{2}\right) \quad \text{mit } \Delta t = t_2 - t_1 \quad (2.6)$$

$S(t_2)$  beschreibt die aktuelle Summe des Messpunktes,  $S(t_1)$  die bei der letzten Messung berechneten Summe,  $t_2$  den jetzigen Zeitpunkt und  $t_1$  der Zeitpunkt der letzten Messung. Gleiches gilt für  $e_2$  und  $e_1$ , welches die Regelabweichung darstellt. Dadurch entsteht die zeitdiskrete Regler-Gleichung für die Implementierung der GUI-Programmierung, welche in Formel 2.7 abgebildet ist.

$$u(t_i) = K_P * e(t_i) + K_I * S(t_i) \quad (2.7)$$

Zusätzlich wird in dieser Masterarbeit zum Filtern von rauschenden Messsignalen ein Verzögerungsglied erster Ordnung verwendet. Auch dieses sogenannte  $PT_1$ -Glied wird für die Implementierung im Programm zu einer zeitdiskreten Gleichung umgeschrieben (Unbehauen, 2008).

$$y_t = y_{t-1} + (K * u_t - y_{t-1}) * \frac{\Delta t}{T_1 + \Delta t} \quad (2.8)$$

Dabei ist  $y_t$  der aktuelle gefilterte Messwert (Ausgangssignal des  $PT_1$ -Glieds) und  $y_{t-1}$  der gefilterte Wert der letzten Messung.  $K$  beschreibt den Verstärkungsfaktor und  $T_1$  die Zeitkonstante des Verzögerungsglieds sowie  $u_t$  den aktuellen ungefilterten Messwert (Eingangssignal des  $PT_1$ -Glieds) und  $\Delta t$  die Abtastzeit der Messung.

## 2.3 Matlab R2021a<sup>®</sup>

Das interaktive Programmsystem Matlab<sup>®</sup> erschien im Jahr 1984 und ist eine Software des Unternehmens The MathWorks, Inc. Es ist eine Datenverarbeitungsumgebung mit proprietärer Programmiersprache dar. Matlab<sup>®</sup> löst und visualisiert mathematische Problemstellungen durch numerische Lösungsverfahren und ermöglicht das Programmieren von kleinen Programmen. Dabei wird Matlab<sup>®</sup> meist für die Erfassung, Analyse und Auswertung von Datensätzen aber auch für die numerische Simulation von technisch-physikalischen Prozessen angewandt. Da oft große Datenmengen gleichzeitig analysiert werden, werden Berechnungen meist mit Hilfe von Matrizen durchgeführt. Matrizen, welche in Matlab<sup>®</sup> nicht vorzeitig dimensioniert werden müssen, stellen somit den grundlegenden Datentyp in Matlab<sup>®</sup> dar. Die Abkürzung Matlab<sup>®</sup> lässt sich auch daher von MATrix LABoratory herleiten (The MathWorks, Inc., 2021; Stein, 2017; Arbenz, 2008).

Matrizen mit nur einem Element, einer Zeile oder Spalte ermöglichen jedoch auch das Erstellen und Arbeiten von Einzelwerten, Zeilen- oder Spaltenvektoren. Zahlen werden standardmäßig als Fließkommazahlen mit doppelter Präzision definiert, dieser Datentyp wird auch *double* genannt. Darüber hinaus können Daten als ganze Zahlen (*integer*), als die Logik-Werte 1 oder 0 (*logical*) oder als Buchstaben in String- oder Arrayform (*string* oder *char*) definiert werden. Diese Datentypen können mit Hilfe von internen Funktionen, z. B. *num2str* (Zahl in Zeichen) ineinander umgewandelt werden (The MathWorks, Inc., 2021). Matlab<sup>®</sup> besitzt von Haus aus viele weitere interne Funktionen. Im folgenden Absatz werden manche in dieser Masterarbeit verwendeten internen Funktionen genauer erklärt.

Ein essenzieller Bestandteil des in dieser Masterarbeit programmierten Codes stellt die Timer-Funktion dar. In Matlab<sup>®</sup> kann ein Timer-Objekt erstellt werden, welches das wiederholte Ausführen eines bestimmten Programmcodes in definierten Zeitintervallen ermöglicht. Dafür wird eine *Callback*-Funktion bei der Timer Definition angegeben, welche den auszuführenden Programmcode enthält. Die vergangene Zeit zwischen dem wiederholenden Aufrufen der Timer-*Callback*-Funktion wird ebenso bei der Timer Definition festgelegt. Wird das Timer-Objekt gestartet, wird so lange, bis das Timer-Objekt gestoppt wird, die Timer-*Callback*-Funktion in den definierten Zeitabständen wiederholt ausgeführt. Die Logik des Erstellens eines Timer-Objektes wird in einem Flowchart in der Abbildung 2.18 dargestellt.

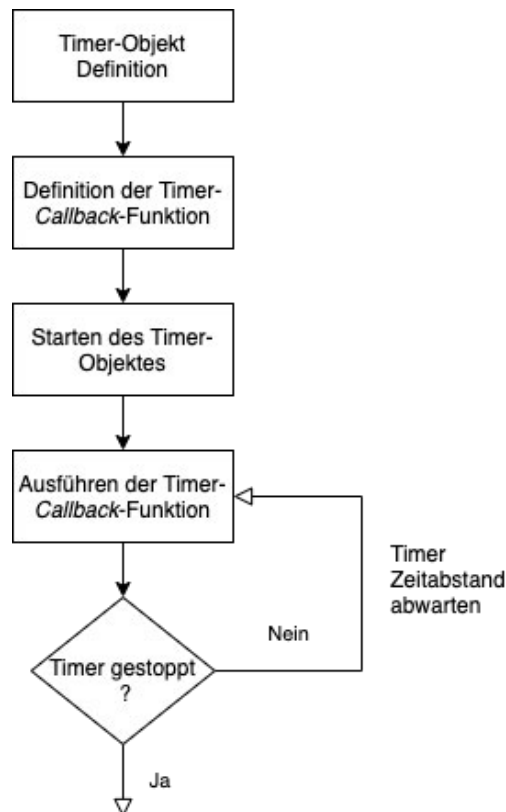


Abbildung 2.18: Grundstruktur der Timer-Logik.

Eine weitere interne Funktion ist der Backslash Operator “\“. Hinter diesem Operator ist die Funktion *mldivide* hinterlegt. Dieser Operator löst das folgende lineare Gleichungssystem nach  $x$  dem Lösungs-Vektor, wobei  $A$  die Koeffizienten-Matrix und  $B$  der Ergebnisvektor genannt wird. Die dafür notwendige Befehlszeile lautet  $x = A \setminus B$ .

$$A * x = B \quad (2.9)$$

Für ein unterbestimmtes System wird die Lösung nach der Methode der kleinsten Fehlerquadrate zurückgegeben.

In Matlab® können kleine Programme als selbst erstellte externe Funktionen oder Skripte realisiert und verbreitet werden. Dadurch sind viele vorgefertigte Skripte oder Anwendungen als „Werkzeugkisten“ den sogenannten Toolboxen für die Nutzer online verfügbar, welches den Funktions- und Anwendungsbereich von Matlab® zunehmend erweitert. Das grundlegende Layout von Matlab® stellt sich aus 5 Bereichen zusammen. In dem Editor-Fenster wird der Programmcode eines Skriptes oder einer

einzelnen Funktion geschrieben. Im Befehlsfenster kann der Benutzer Befehle eingeben, welche live von Matlab® umgesetzt werden. Der sogenannte *Workspace* zeigt alle definierten Variablen und deren Werte an. Zusätzlich gibt es ein Fenster, welches den aktuell ausgewählten Matlab®-Pfad inklusive der Dateien anzeigt und ein Fenster, in dem die Befehlshistorie dokumentiert wird (The MathWorks, Inc., 2021; Stein, 2017; Arbenz, 2008).

### 2.3.1 Matlab App Designer®

Matlab App Designer® ist eine interaktive Entwicklungsumgebung, welche es Benutzern erlaubt, das Layout einer Computer-Anwendung zu designen und das Verhalten der Programmkomponenten gezielt zu programmieren. Mit Hilfe von Matlab App Designer® und des Matlab Compilers können selbstständig laufende, Matlab®-unabhängige Desktop- oder Webanwendungen nur mittels der Matlab® internen Programmiersprache ohne tiefgehende Programmierkenntnisse erstellt werden. Matlab App Designer® verfügt dafür über eine Designansicht, in der das Layout der graphischen Benutzeroberfläche (GUI) durch vielzählige UI-Komponenten (Benutzeroberflächen Komponenten) per Drag and Drop erstellt und angepasst werden kann. Beispiele für verfügbare Softwarekomponenten sind Buttons (Schaltflächen), Kontrollkästchen, Zahlen- und Textfelder, Dropdown-Menüs (Auswahlmenü), Tabellen, Button Gruppen und Graphen. Beim Hinzufügen einer Softwarekomponente erstellt Matlab App Designer® automatisch den objektorientierten Code, welcher Layout und Design festlegt. Der Programmcode kann in dem integrierten Matlab® Editor, welcher sich in der Codeansicht der Anwendung finden lässt, betrachtet und bearbeitet werden. In diesen können auch eigene Funktionen erstellt werden. Die Interaktionen mit den Softwarekomponenten werden über sogenannte *Callback*-Aktionen definiert. Die verschiedenen Elemente besitzen verschiedenen *Callback*-Aktionen (The MathWorks, Inc., 2021). Die folgende Tabelle listet die *Callback*-Aktionen der in dieser Masterarbeit am meisten verwendeten UI-Komponenten auf.

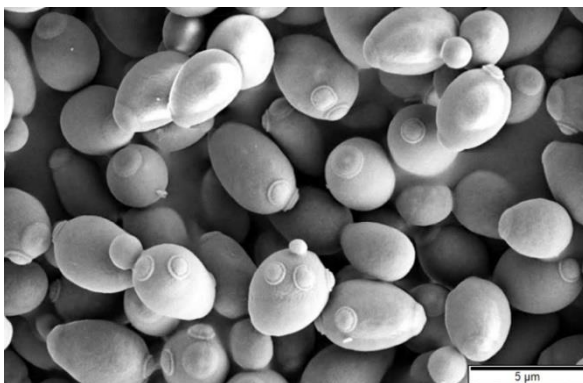
**Tabelle 2.1: *Callback*-Aktionen der in dieser Masterarbeit am meisten vorkommenden UI-Komponenten.**

<b>Softwarekomponente</b>	<b>Mögl. <i>Callback</i>-Aktion</b>
Button	<i>ButtonPushed</i>
Button Gruppe	<i>SelectionChanged, SizeChangedFnc, ButtonDownFnc</i>
Kontrollkästchen	<i>ValueChanged</i>
Zahlen- oder Textfeld	<i>ValueChanged</i>
Dropdown Menü	<i>ValueChanged, DropDownOpeningFnc</i>
Tabelle	<i>TableCellEdit</i>
Graph	<i>ButtonDownFnc</i>
Label	-

In der Designansicht können den Komponenten eine oder mehrere *Callback*-Aktionen zugewiesen werden. Daraufhin wird der Rahmen einer passenden *Callback*-Funktion automatisch im Editor generiert. In dieser *Callback*-Funktion kann der Benutzer nun den gewünschten Programmcode hinzufügen, welcher beim Auslösen der *Callback*-Aktion ausgeführt werden soll. Ein *Code Analyzer* überprüft automatisch den Programmcode auf Probleme oder Fehler, welche in der Programmierung auftreten können, und visualisiert diese dem Benutzer. Zudem gibt es einen Komponenten- und Code-Browser. In ihnen sind alle erstellten Komponenten bzw. *Callback*- oder selbst erstellte Funktionen gelistet (The MathWorks, Inc., 2021).

## 2.4 *Saccharomyces cerevisiae*

Die Hefe (lat.: *Fermentum*) *Saccharomyces cerevisiae* (deu.: Bäckerhefe) gehört zu der Gattung der Zuckerhefen (*Saccharomyces*) welche der Ordnung der echten Hefen (*Saccharomycetales*) angehören. Sie wird seit Jahrhunderten für die Produktion alkoholischer Getränke und Backwaren eingesetzt. Grund dafür ist die Fähigkeit, verschiedene Zucker vollständig zu Ethanol und Kohlenstoffdioxid zu fermentieren. Die Hefe ist ein einzelliger Organismus, der zu einer schnellen Zellteilung auf definiertem Medium befähigt ist, wobei die Vermehrung vegetativ durch Knospung erfolgt (Hartwell, 1974). Die Zellen besitzen einen Durchmesser von ca. 5  $\mu\text{m}$  und eine runde bis ovale Struktur. 1996 wurde das Genom als erster eukaryotischer Organismus von den Forschern um Goffeau *et al.* vollständig sequenziert. Grundlegende Kenntnisse seiner Genetik, Biochemie und Zellbiologie sind seit Jahrzehnten etabliert, weshalb *S. cerevisiae* seit Jahren als Model für die Forschung in den Bereichen der Regulation von Gen-Expressionen, der Signaltransduktion, des Zellzyklus, des Metabolismus, der Apoptose und viele weitere biologische Prozesse verwendet wird (Curran und Bugeja, 2006; Karathia *et al.*, 2011). Darüber hinaus verfügt *S. cerevisiae* über eine sehr aktive homologe Rekombination, was das effiziente Inaktivieren oder Modifizieren von Genen ermöglicht (Ji *et al.*, 2020). Diese Hefe gilt als einer der bestuntersuchten eukaryotischen Mikroorganismen und die genannten Eigenschaften machen ihn zu einem der meistgenutzten eukaryotischen Modellorganismen (Karathia *et al.*, 2011). Zusätzlich kommt heutzutage *S. cerevisiae* außerhalb der Lebensmittelindustrie zur industriellen heterologen Proteinexpression zum Einsatz. Viele pharmazeutische Proteine wie Insulin, Oberflächenantigene des Hepatitis B Virus oder Glukagon werden mittels rekombinanter *S. cerevisiae* Stämmen produziert (Razanskiene *et al.*, 2004).



**Abbildung 2.19: *Saccharomyces cerevisiae*.** Zu sehen ist eine Rasterelektronenmikroskop-Aufnahme von *Saccharomyces cerevisiae* Zellen (Murtey und Ramasamy, 2016).

## 3 Material und Methoden

### 3.1 Materialien

In dem folgenden Abschnitt werden verwendete Geräte, Verbrauchsmaterialien, Chemikalien sowie Lösungen in alphabetischer Reihenfolge aufgelistet. Relevante Einstellungen der Geräte werden in dem Abschnitt Methoden genannt.

#### 3.1.1 Geräte

In der folgenden Tabelle werden die in dieser Masterthesis verwendeten Geräte aufgeführt.

**Tabelle 3.1: Verwendete Geräte.**

Gerät	Hersteller, Typ	Anmerkung
3-Wege-Ventil	Bürkle	
Adapter RS232/USB	LogiLink	
Dispergator	Kinematica Polytron PT 1300D	
Druckmonitor	SciLog, SciPres	
Drucksensoren	SciLog, SciPres	
Exsikkator	Nalgene	
Feinwaage	Sartorius BP221S	
Hohlfasermodule	Repligen, Midikros 20cm	mPES-Membran, P/N: D02-E750-10N
Kulturflaschen	Schott, 1000 ml	mit und ohne unteren Auslassstutzen
Laptop	Dell, Latitude 5510	
Magnetrührer	IKA, biq squid white	
Magnetrührer	VWR	
Nylonsieb	HAW Hamburg	
Peristaltische Pumpen	Watson Marlow, 323Du	Anzahl: 2
Photometer	Pharmacia Biotech, Ultrospec 3000	
Pinzette	HAW	
Pipetten	VWR	
Pumpenköpfe	Watson Marlow, 313DW	Rollen: 3, Anzahl: 3
Schlauchklemmen	HAW Hamburg	nach Hoffmann
Schraubverschluss mit Stutzen	BORA, GL 45	3 Stutzen mit Anschlussoliven



Gerät	Hersteller, Typ	Anmerkung
Schraubverschluss mit Stützen	Schott, GL 45	2 Stützen mit Anschlussoliven
Seriell-Hub Konverter	ATEN, UC2324	4 Port
Trockenschrank	Heraus Instruments, T6060	
Ultrazentrifuge	Thermo Scientific, Sorvall Lynx 6000	Rotor: 6162802
USB-Verlängerungskabel	Delock	
Vortexer	Phoenix Instruments, RS-VA10	
Waage	Sartorius, TE6101	Tischwaage, Anzahl: 2
Waage	Sartorius, BP 8100	Tischwaage
Waage	Sartorius, Quintix3102 - 1S	Tischwaage
Zentrifuge	Sartorius, A-14	Rotor: 12092

### 3.1.2 Verbrauchsmaterialien

In Tabelle 3.2 sind die verwendeten Verbrauchsmaterialien aufgelistet.

**Tabelle 3.2: Verwendete Verbrauchsmaterialien.**

Verbrauchsmaterial	Hersteller	Artikelnummer
20 ml Spritze	Braun	4606736V
250 ml Zentrifugenröhrchen	Nalgene	11375574
Einweg-UV-Küvetten	Roth	91029030
Pasteurpipetten 5 ml	Roth	EA56.1
pH Indikator Streifen	Roth	0549
Pipettenspitzen 1250 µl	Roth	B006.1
Pipettenspitzen 200 µl	Brand	732028
Reaktionsgefäße 1,5 ml	Roth	4189.1
Schläuche (1,6 mm ID und 1,6 mm Wandstärke)	Watson Marlow	913.AJ16.016
Schläuche (3,2 mm ID und 1,6 mm Wandstärke)	Watson Marlow	913.AJ32.016
Schläuche (8,0 mm ID und 1,6 mm Wandstärke)	Watson Marlow	913.AJ80.016
0,2 µm Spritzenvorsatzfilter	GE Healthcare	10462200

### 3.1.3 Chemikalien

Die in dieser Arbeit verwendeten Chemikalien und Agenzien sind in der folgenden Tabelle zusammengefasst.

Tabelle 3.3: Verwendete Chemikalien und Agenzien.

Chemikalien	Hersteller	Artikelnummer
Natriumhydroxid (NaOH)	Roth	6771.1

### 3.1.4 Lösungen

In Tabelle 3.4 sind die verwendeten Lösungen aufgelistet.

Tabelle 3.4: Verwendete Lösungen.

Lösung	Stoff	Stoffmenge
0,1 M NaOH	NaOH	100 mM
	ddH <sub>2</sub> O	ad. 1000 ml

### 3.1.5 Verwendeter Organismus

Für alle Experimente dieser Masterarbeit wurde getrocknete *Saccharomyces cerevisiae* verwendet. Diese Hefe (Artikel Nr.: X0019CVEDX) wurde durch Hanse & Pepper e.K. vermarktet.

## 3.2 Methoden

### 3.2.1 Vorversuche

#### 3.2.1.1 Aktivierungszeit der suspendierten *Saccharomyces cerevisiae*

Um herauszufinden, wie lange die Hefezellen in Lösung gerührt werden müssen, damit sie vollständig aktiviert sind und sich ihre optische Dichte nicht mehr verändert, wird ein Vorversuch durchgeführt. Hierfür werden 80 g getrocknete *Saccharomyces cerevisiae* eingewogen und in eine Schottflasche mit 1 L VE-Wasser überführt. Die Hefezellen werden bei 700 RPM auf einem Magnetrührer resuspendiert. Nach vollständigem Resuspendieren wird die Suspension weiterhin gerührt und es werden im 5-Minuten-Takt Proben entnommen ( $V_{\text{Probe}} = 1 \text{ ml}$ ). Diese werden auf ihre optische Dichte untersucht. Die OD-Messung (optische Dichte) ermittelt den Unterschied der Lichtintensität zwischen Lichtquelle und Detektor des Photometers. Die Methode basiert auf der Lichtstreuung von Partikeln in der Probe (trübungsabhängig), wodurch weniger Licht den Detektor erreicht. Die Messung wird bei einer Wellenlänge von 600 nm in einer Einweg-Küvette durchgeführt. Dafür wird das Photometer von Pharmacia Biotech verwendet. Nach der Messung eines Referenzwertes mit reinem VE-Wasser werden die Proben 1:1000 verdünnt und  $OD_{600\text{nm}}$  gemessen. Die Bestimmung der optischen Dichte erfolgte nach SOP-Nr.: 320101-04 (8.6 Standardarbeitsanweisungen).

Die Ergebnisse (5.1.1 Aktivierungszeit der suspendierten *Saccharomyces cerevisiae*) ergaben, dass nach 40 min Rühren die optische Dichte der Suspension keine größeren Schwankungen mehr aufweist. Im weiteren Verlauf werden resuspendierte Hefesuspensionen demnach 40 min gerührt.

### 3.2.1.2 Ermittlung des Korrelationsfaktors zwischen Biotrockenmassekonzentration und OD<sub>600nm</sub>

Vor jedem Filtrationsversuch muss überprüft werden, ob die angesetzte Hefesuspension die gewünschte Konzentration besitzt. Dies soll durch Messung der optischen Dichte ermöglicht werden. Um von der OD<sub>600nm</sub> der Hefesuspension auf ihre Biotrockenmassekonzentration schließen zu können, benötigt man einen Korrelationsfaktor zwischen beiden Parametern. Hierfür werden zunächst acht Hefesuspensionen mit verschiedenen Konzentrationen angesetzt (3,5,10,15,20,25,30,50 g/L). Die dazugehörigen Einwaagen und Verdünnungen sind dem Anhang zu entnehmen. Die Hefesuspensionen werden auf einem Magnetrührer für 40 min (3.2.1.1 Aktivierungszeit der suspendierten *Saccharomyces cerevisiae*) bei 700 RPM gerührt. Anschließend werden zwei 1 ml Proben von jeder Hefesuspension genommen. Zum einen wird die optische Dichte nach einer Referenzwertmessung (VE-Wasser) und einer möglichen Verdünnung der Proben (OD<sub>600nm</sub> abhängig) im Photometer gemessen. Zum anderen wird eine gravimetrische Bestimmung der Biotrockenmassekonzentration der Proben durchgeführt. Hierfür werden die Proben in ein vorher getrocknetes (Trockenschrank 105 °C über Nacht) und ausgewogenes 1,5 ml Mikroreaktionsgefäß überführt. Die Proben werden bei 13.000 RPM für 3 min zentrifugiert. Der Überstand wird abgenommen und das Pellet im Mikroreaktionsgefäß über Nacht erneut bei 105 °C im Trockenschrank getrocknet. Am nächsten Tag wird das Mikroreaktionsgefäß nach dem Abkühlen in einem Exsikkator erneut gewogen und die Masse des getrockneten Zellpellets ( $m_{\text{Pellet}}$ ) ermittelt. Dieses Vorgehen erlaubt die Berechnung der Biotrockenmassekonzentration (Formel 3.1). Die Bestimmung erfolgte nach SOP-Nr.: 320001-03 (8.6 Standardarbeitsanweisungen).

$$C_{XL} = \frac{m_{\text{Pellet}}}{V_L} \quad (3.1)$$

Durch Auftragen der Biotrockenmassekonzentrationen gegen die dazugehörigen OD<sub>600nm</sub> Werte und Erstellung einer Regressionsgeraden kann der passende Korrelationsfaktor ermittelt werden. Die Ergebnisse (5.1.2 Ermittlung des Korrelationsfaktors zwischen Biotrockenmassekonzentration und OD<sub>600nm</sub>) ergaben einen Korrelationsfaktor von  $0,3667 \frac{\text{gBTM} \cdot \text{L}^{-1}}{\text{AU}}$ . Zusätzlich stellte sich heraus, dass das eingewogene Gewicht

der getrockneten *S. cerevisiae* pro Liter VE-Wasser nicht der Biotrockenmassekonzentration der Suspension entspricht. Auch dieser Umrechnungsfaktor wurde ermittelt und liegt bei  $0,7817 \frac{\text{gBTM} \cdot \text{L}^{-1}}{\text{gTrockenhefe} \cdot \text{L}^{-1}}$ . Dies wird im weiteren Verlauf bei der Einwaage der Hefe berücksichtigt, um die gewünschten Biotrockenmassekonzentrationen zu erreichen.

### 3.2.2 Wasserwert-Versuche des Hohlfasermoduls

Bevor zellhaltige Suspensionen auf das Hohlfasermodul gegeben werden, wird das Modul zunächst mit VE-Wasser eingefahren und die Filtrationsrate des Wassers untersucht und dokumentiert. Diese Werte können als Gütekriterium der Modulreinigung nach einem Versuch mit zellhaltiger Suspension herangezogen werden. Der Wasserwert (Permeatfluss des Wassers durch die Membran) soll nach der Reinigung der Membran bei ca. 60 % des initialen Wasserwertes liegen. Darüber hinaus können Wasserwerte bei verschiedenen Parametereinstellungen (Flussrate/Schergeschwindigkeit und Transmembrandruck) bereits Einflüsse dieser Parameter auf die Permeabilität des Wassers durch die Filtermembran (sog. Flux) zeigen.

Dafür wird zunächst die Feedpumpe mit Hilfe des Computer Programms der TFF-Anlage kalibriert. Anschließend wird 1 L VE-Wasser in den Feedbehälter der TFF-Anlage gefüllt. Mit Hilfe der Anwendung wird die gewollte Flussrate der Feedpumpe und somit eine definierte Schergeschwindigkeit über die Fasern des Moduls vorgegeben. Nach Starten des Prozesses und somit der Feedpumpe wird durch die retentatseitige Schlauchklemme der Transmembrandruck eingestellt. Die permeatseitige Schlauchklemme ist zu 100 % geöffnet. Nach 5-minütiger Filtration wird der vom Programm aufgezeichnete Flux Wert (Wasserwert) dokumentiert. Das Programm wird nun gestoppt und das VE-Wasser zurück in den Feedbehälter geführt. Nach Einstellung neuer Parameter wird das Programm erneut gestartet. Es werden die Schergeschwindigkeiten 6000 1/s (Flussrate = 424 ml/min), 8000 1/s (Flussrate = 565 ml/min) und 10000 1/s (Flussrate = 707 ml/min) jeweils bei Transmembrandrücken von 150, 250, 350 und 450 mbar untersucht.

### **3.2.3 Konzentrationsversuche der *Saccharomyces cerevisiae* Zellsuspensionen**

#### **3.2.3.1 Herstellung und Aktivierung der Zellsuspension**

Für die Konzentrationsversuche werden 1000 ml einer *Saccharomyces cerevisiae* Zellsuspension mit einer Biotrockenmassekonzentration von 50 g/L verwendet. Hierfür werden 63,9 g getrocknete *Saccharomyces cerevisiae* eingewogen und in eine 1 L Schottflasche mit 500 ml VE-Wasser überführt. Die Hefezellen werden bei 700 RPM auf einem Magnetrührer resuspendiert. Nach vollständigem Resuspendieren wird die Hefesuspension zur Aktivierung der Hefezellen für weitere 40 min bei 700 RPM auf dem Magnetrührer gerührt.

#### **3.2.3.2 Waschen der *Saccharomyces cerevisiae***

Um den in der getrocknete *Saccharomyces cerevisiae* enthaltenen Emulgator zu entfernen, wird die aktivierte Zellsuspension gewaschen. Diese wird gleichmäßig auf drei 250 ml Zentrifugenröhrchen aus Polypropylen-Copolymer verteilt. Ein weiteres 250 ml Zentrifugenröhrchen wird zum Austarieren der Zentrifuge mit VE-Wasser befüllt. Jeweils zwei der vier Zentrifugenröhrchen werden aufeinander auf ein zehntel Gramm genau ausgewogen. Es folgt ein 10-minütiger Zentrifugationsschritt bei einer G-Zahl von 5000. Dabei wird die Ultrazentrifuge von Thermo Scientific verwendet. Anschließend werden die Überstände vorsichtig dekantiert und die Pellets mit jeweils 160 ml frischem VE-Wasser resuspendiert. Dieses Vorgehen wird für zwei weitere Waschschrte wiederholt. Nach dem dritten Dekantieren des Überstandes werden die Pellets in nur 130 ml VE-Wasser resuspendiert, um das zu prozessierende Volumen für den nächsten Schritt zu reduzieren.

#### **3.2.3.3 Homogenisierung der gewaschenen Zellsuspension**

Die gewaschenen Hefesuspensionen werden mit Hilfe eines Dispergators (Kinematica Polytron) jeweils für 2 min bei 10.000 RPM gründlich homogenisiert. Anschließend werden die einzelnen Suspensionen zusammengeführt, wobei sie durch ein Nylon

Sieb gefiltert werden. Beide Verfahren sollen sicherstellen, dass sich keine Hefezellklumpen in der Suspension befinden welche das Hohlfasermodule verstopfen könnten. Die zusammengeführte Suspension wird nun auf ein Flüssigvolumen von 1000 ml aufgefüllt, um die Zielkonzentration von 50 g/L zu erreichen.

### 3.2.3.4 Tangentialflussfiltration

Zur Lagerung über Nacht ist die Anlage sowie das Hohlfasermodule mit 0,1 M NaOH befüllt. Diese muss zunächst herausgespült werden (Abbildung 3.1). Dafür wird bei laufender Pumpe kontinuierlich frisches VE-Wasser in den Feedbehälter nachgefüllt, bis retentat- sowie permeatseitig ein pH-Indikatorstreifen ein neutralen pH-Wert anzeigt. Ebenso wird vor jedem Versuch der Wasserwert des Hohlfasermodule unter definierten Parametereinstellungen (Schergeschwindigkeit 6000 1/s; Flussrate = 424 ml/min, TMP = 200 mbar) gemessen und dokumentiert (vgl. 3.2.2 Wasserwert-Versuche des Hohlfasermodule).

Vor Filtrationsbeginn wird von der Zellsuspension eine 100 µl Probe genommen. Diese wird 1:1000 verdünnt und ihre optische Dichte gemessen (3.2.1.1 Aktivierungszeit der suspendierten *Saccharomyces cerevisiae*). Mit Hilfe des Korrelationsfaktors (3.2.1.2 Ermittlung des Korrelationsfaktors zwischen Biotrockenmassekonzentration und OD<sub>600nm</sub>) kann auf die Biotrockenmassekonzentration der Suspension geschlossen werden. Es wird darauf geachtet, dass näherungsweise immer die gleiche Biotrockenmassekonzentration (ca. 50 g/L) vorliegt, um reproduzierbare Ergebnisse zu erzielen. Bei Abweichungen wird die Konzentration ggf. durch Zugabe von VE-Wasser adjustiert oder die Suspension vollständig neu angesetzt. Für jeden Versuch wird die ermittelte Konzentration dokumentiert. Im Computer Programm der Anlage wird zunächst die Feedpumpe kalibriert und daraufhin der gewünschte Prozessablauf sowie die gewünschten Prozessparameter für den jeweiligen Versuch eingestellt. Es werden vier Reproduzierbarkeitsversuche bei jeweils einer Schergeschwindigkeit von 8000 1/s (Flussrate = 565 ml/min) und bei einem Transmembrandruck von 250 mbar untersucht. Nun wird die *Saccharomyces cerevisiae* Zellsuspension in den Feedbehälter überführt, wo diese bei 350 RPM mittels Magnetrührer gerührt wird. Dies soll eine homogene Verteilung der Zellen in der Suspension gewährleisten. Es folgt ein

softwareseitiger Tarierschritt der Feedwaage und ein *Fill Up* der Anlage. Der Filtrationsprozess kann dann am Computer gestartet werden.

Zu Beginn der Filtration ist die permeatseitige Schlauchklemme komplett geschlossen und die Hefesuspension wird lediglich rezirkuliert. Sobald die gewünschte Flussrate erreicht ist, wird die permeatseitige Schlauchklemme sehr vorsichtig geöffnet, damit die Zellen nicht ruckartig in die Poren der Membran gedrückt werden. Im Anschluss wird durch Drosselung der retentatseitigen Schlauchklemme der geforderte Transmembrandruck eingestellt. Um den TMP im Verlauf der Filtration konstant zu halten, muss diese weiterhin zum kontinuierlichen Gegenregeln verwendet werden. Erreicht der Flux einen verschwindend geringen Wert ( $< 10 \text{ L m}^{-2} \text{ h}^{-1}$ ) beendet sich der Filtrationsprozess automatisch (Endpunkt). Nähert der Eingangsdruck des Hohlfasermoduls sich bereits davor einem Wert nahe 2 bar, wird die Filtration über das Programm abgebrochen.

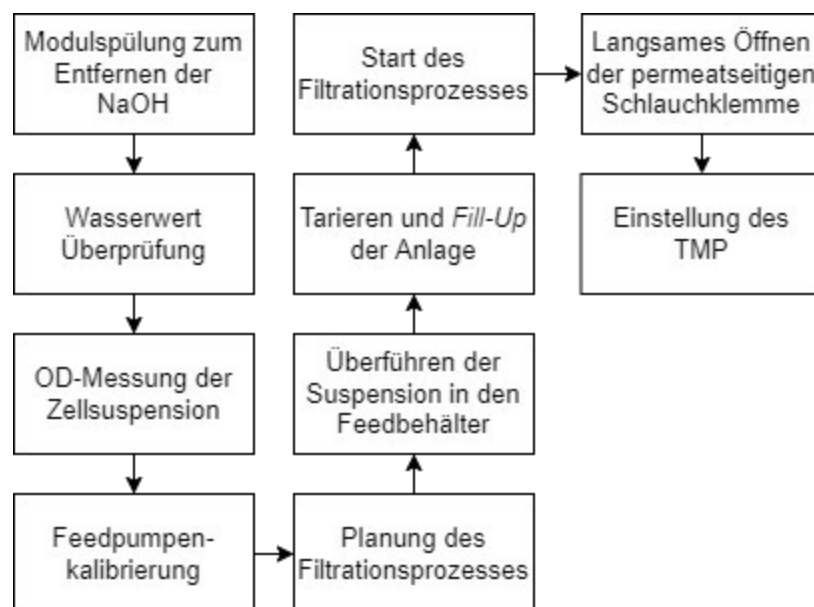


Abbildung 3.1: Flowchart von dem Vorbereitungsablauf eines Filtrationsprozesses.



### 3.2.3.5 Reinigung der TFF-Anlage

Nach Beendigung der Filtration wird das Filtrat sowie das Retentat nach dem Zurückspülen in den Feedbehälter aus der Anlage entfernt. Zum Beseitigen von Suspensionsresten findet ein Spülschritt mit ca. 5 L VE-Wasser unter der jeweiligen Prozessflussrate statt. Hierbei ist die retentatseitige Schlauchklemme geöffnet und die permeatseitige Schlauchklemme vollständig geschlossen. Das Retentat wird während dieses Schrittes in einen Abfallbehälter geleitet. Zur Entfernung von Zellen, welche möglicherweise in die Poren der Membran geraten sind, folgt bei gestoppter Pumpe ein permeatseitiges Rückspülen der Hohlfasermembran. Dafür wird eine mit VE-Wasser gefüllte 20 ml Spritze an den unteren Permeatausgang befestigt und das VE-Wasser sehr vorsichtig von hinten durch die Membranporen gedrückt. Anschließend wird der vorherige Spülschritt mit weiterem VE-Wasser fortgesetzt, bis klares Retentat aus der Anlage fließt.

Im Anschluss werden zum Beseitigen von feststehenden Zellresten Spülschritte mit Natronlauge durchgeführt. Dazu wird 0,1 M NaOH Lösung in den Feedbehälter gefüllt. Die permeatseitige Schlauchklemme wird nun geöffnet und die Feedpumpe langsam auf die jeweilige Prozessflussrate gesteigert. Retentat sowie Permeat fließen in einen Abfallbehälter. Ist der Permeatraum des Hohlfasermoduls vollständig mit 0,1 M NaOH geflutet wird die permeatseitige Schlauchklemme wieder geschlossen. Hierauf findet ein erneutes Rückspülen der Hohlfasermembran mit 20 ml 0,1 M NaOH statt (s. oben), gefolgt von weiterem Spülen in den Abfallbehälter. Der Retentatschlauch wird danach zurück in den Feedbehälter gleitet und es folgt ein 30-minütiges Rezirkulieren frischer 0,1 M NaOH Lösung bei 130 % der Prozessflussrate. Im Anschluss wird die Anlage inklusive Hohlfasermodul erneut zur Lagerung über Nacht mit frischer 0,1 M NaOH Lösung geflutet und beide Schlauchklemmen vollständig geschlossen.

## 4 Entwicklung der TFF-Anlage

In diesem Kapitel werden zum einen die verwendeten Bestandteile sowie der Zusammenbau der Tangentialflussfiltrationsanlage erläutert (4.1 Aufbau der Anlage), zum anderen der Entwicklungsprozess der graphischen Benutzeroberfläche (GUI) zur Durchführung und Überwachung automatisierter Konzentrations- und Diafiltrationsprozesse beschrieben und erklärt (4.2 Entwicklung der graphischen Benutzeroberfläche).

### 4.1 Aufbau der Anlage

Die folgende Abbildung (Abb. 4.1) zeigt eine schematische Darstellung einer Tangentialflussfiltrationsanlage zur Durchführung von Konzentrations- und Diafiltrationsprozessen. Die in dieser Masterarbeit zu entwickelnde TFF-Anlage soll nach diesem Schema aus den abgebildeten Bestandteilen zusammengebaut werden. Ebenso muss die dargestellte Verbindung der Geräte mit einem externen Computer für die Steuerung und Regelung der Geräte realisiert werden. In den folgenden Unterkapiteln werden die einzelnen Komponenten sowie ihr Zusammenbau näher erläutert.

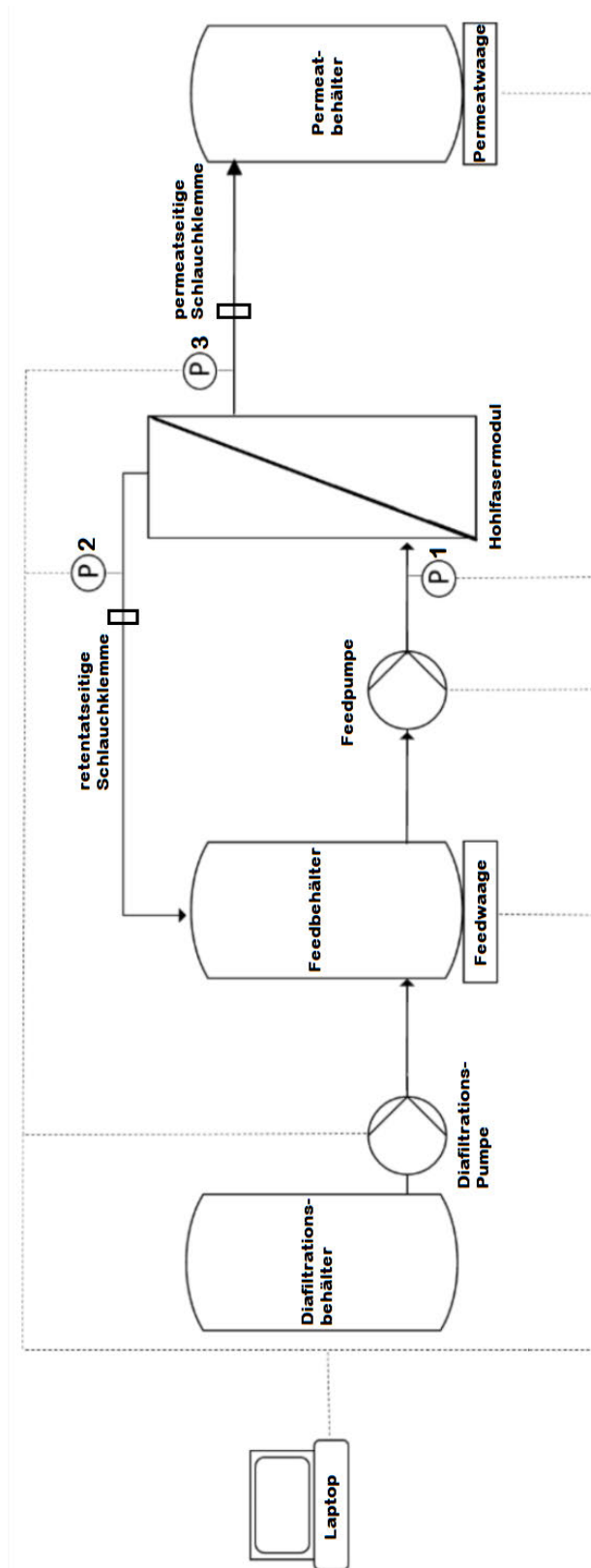
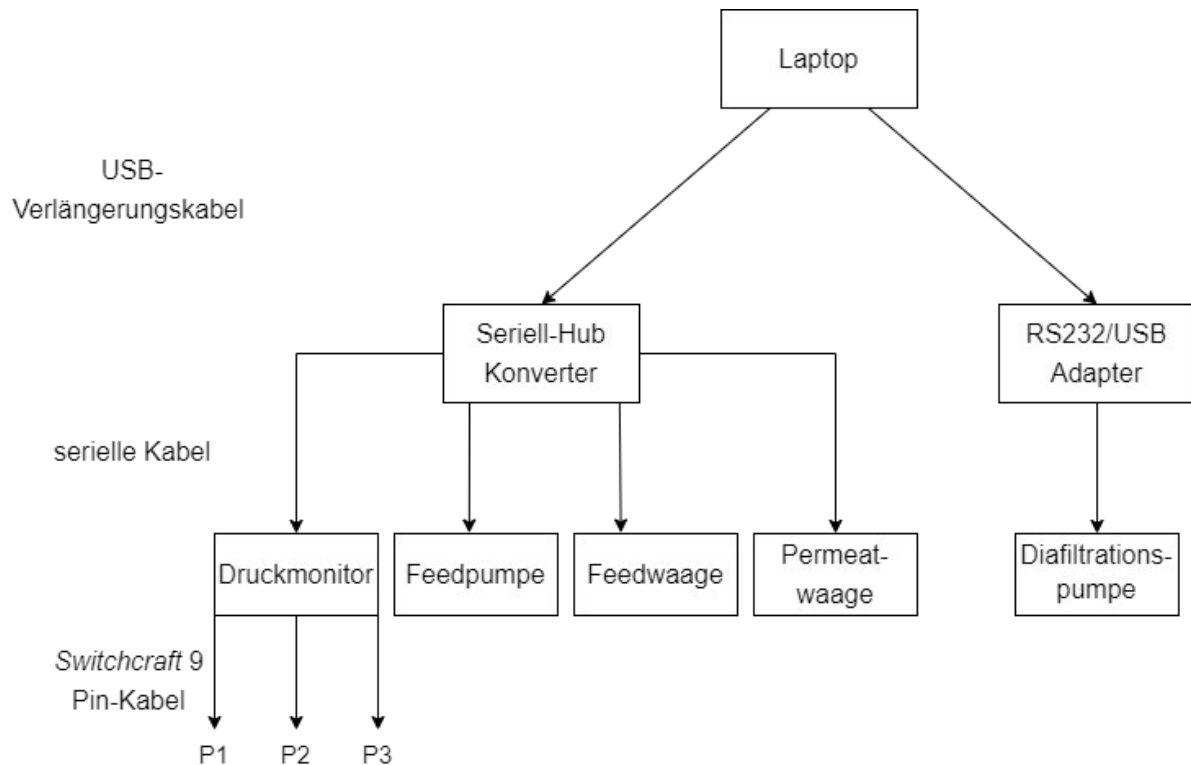


Abbildung 4.1: Schematische Darstellung der TFF-Anlage (verändert nach Loewe et al., 2022). Die gestrichelten Linien stellen eine Anbindung an den externen Computer dar.

### 4.1.1 Systemintegration



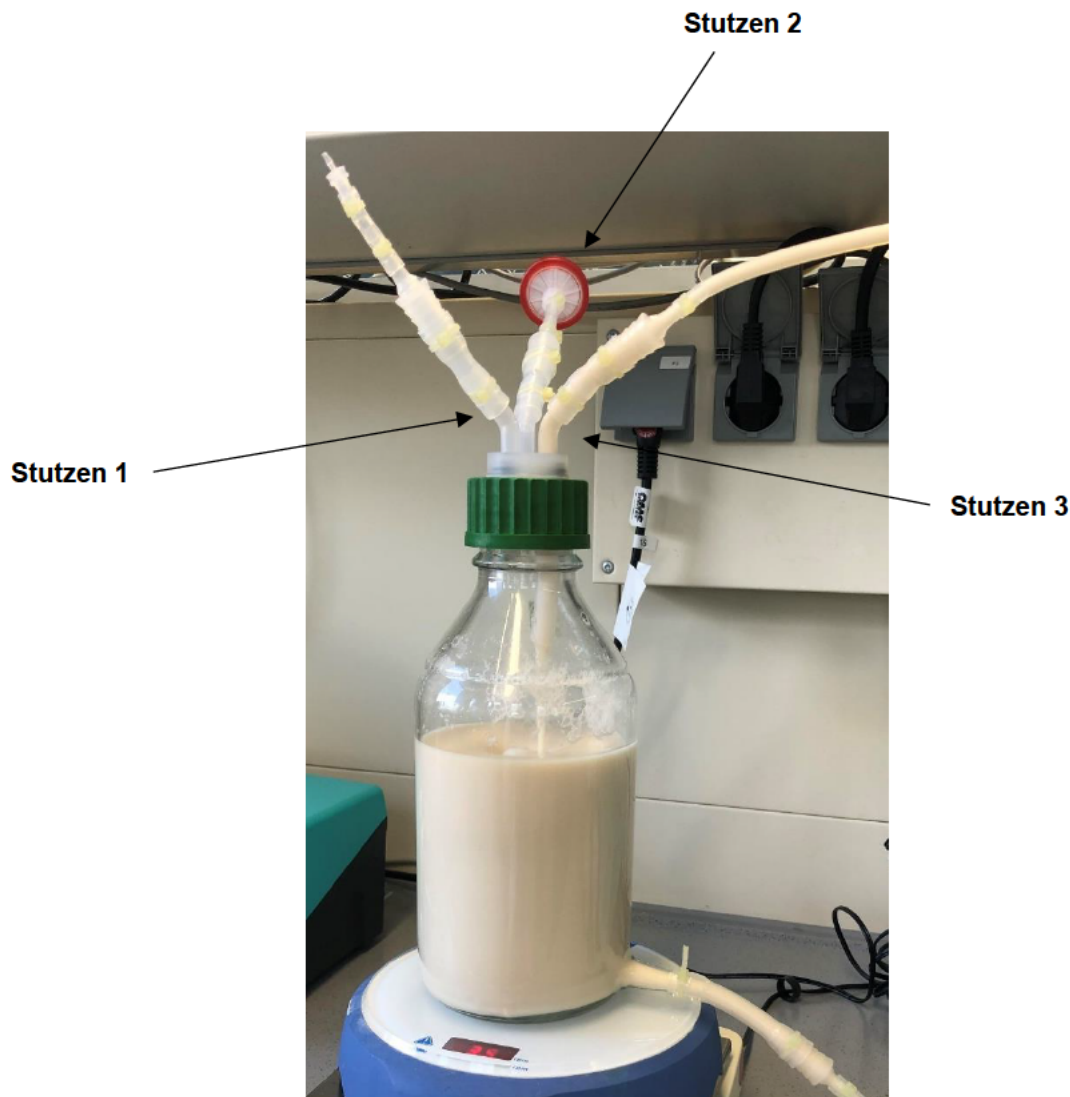
**Abbildung 4.2: Schematische Darstellung der Systemkommunikation zwischen dem externen Laptop und den einzelnen Geräten.** P1, P2 und P3 stellen die mobilen Drucksensoren des Moduleingangs, Modulausgangs und des Permeatausgangs dar.

Die obige Abbildung zeigt die Anbindung der ansteuerbaren Geräte an den externen Computer. Für die Steuerung und Überwachung der TFF-Anlage wird als Endgerät ein Laptop der Marke Dell Inc. (Modell: Latitude 551) eingesetzt. Als Betriebssystem ist Microsoft Windows 10 Pro installiert. Der Laptop besitzt 3 USB 3.1 - Ports, über die die Kommunikation mit den ansteuerbaren Geräten erfolgt. Zwei USB-Verlängerungskabel verbinden dafür den Laptop mit einem Seriell-Hub Konverter der Marke ATEN (4 x RS232 zu USB) und einem RS232/USB Adapter von LogiLink (Abbildung 4.2). Damit verfügt der Laptop über 5 physische COM-Ports (engl.: *Communication Ports*) über die die seriellen Geräte angeschlossen werden können.

## 4.1.2 Gefäße und Schraubverschlüsse

### 4.1.2.1 Feedbehälter

Im Feedbehälter befindet sich die zu konzentrierende/diafiltrierende Zellsuspension. Im Falle einer Konzentrierung wird nur das Retentat zurück in den Feedbehälter geführt und dort aufkonzentriert. Bei einer Diafiltration wird das Volumen im Feedbehälter konstant gehalten. Neben dem Retentat wird hierbei auch Diafiltrationspuffer aus dem Diafiltrationsbehälter in den Feedbehälter gepumpt. Als Feedbehälter wird eine 1 L Schottflasche mit unterem Auslassstutzen verwendet (Abbildung 4.3).



**Abbildung 4.3: Nahaufnahme des Feedbehälters.** Zu sehen sind die 3 Stutzen des Schraubverschlusses des Feedbehälters, sowie der Auslassstutzen am Boden der Flasche. Stutzen 1 ist auf diesem Bild abweichend vom Text nicht mit der Diafiltrationspumpe verbunden, da hier eine einfache Konzentrierung stattgefunden hat.

Der Auslassstutzen ist mit Hilfe eines kurzen Schlauchstückes und eines Schlauchadapters mit dem Schlauch der Feedpumpe verbunden. Geschlossen wird der Feedbehälter mit einem 3 Stutzen-Schraubverschluss (GL 45) aus Polypropylen. Stutzen 1 ist mit dem Schlauch der Diafiltrationspumpe verbunden. Über ihn gelangt im Falle einer Diafiltration frischer Diafiltrationspuffer in den Feedbehälter. Stutzen 2 ist über einen normalen Schlauchadapter, einen Luer-Lock-Schlauchadapter und 2 zusätzlichen kurzen Schlauchstücken mit einem 0,2 µm Spritzenvorsatzfilter verbunden. Dieser Stutzen dient zum sterilen Druckausgleich. Stutzen 3 ist mit dem retentatseitigen Filtermodulaustritt verbunden. Über ihn wird das Retentat zurück in den Feedbehälter geführt. Zusätzlich beinhaltet die Retentatleitung ein 3-Wege-Ventil, welches eine Probenahme des Retentats ermöglicht. An der Innenseite des dritten Stutzens ist ein 3 cm Schlauchstück befestigt, welches ein geführtes Einleiten der Suspension in den Feedbehälter gewährleisten und eine starke Schaumbildung verhindern soll. Der Feedbehälter inklusive Anschlüsse befindet sich auf einem Magnetrührer, welcher auf einer Tischwaage platziert ist (4.1.6.1 Feedwaage).

#### **4.1.2.2 Diafiltrationsbehälter**

Der Diafiltrationsbehälter dient bei einem Diafiltrationsprozess als Diafiltrationspuffer Reservoir. Verwendet wird hierfür eine 2 L Schottflasche mit einem 2 Stutzen-Schraubverschluss (GL 45) aus Polypropylen und Edelstahl. Ein Stutzen dient zum Druckausgleich des Behälters. Der andere Stutzen ist mit dem Schlauch der Diafiltrationspumpe verbunden. An der Innenseite dieses Stutzens ist ein Tauchrohr befestigt, welches bis auf den Boden des Behälters ragt und den Transport des Diafiltrationspuffers ermöglicht.

#### **4.1.2.3 Permeatbehälter**

Im Permeatbehälter sammelt sich die durch das Hohlfasermodule filtrierte Flüssigkeit. Dafür wird bei einem einfachen Konzentrationsprozess eine 1 L Schottflasche und bei Diafiltrationsprozessen eine 5 L Schottflasche verwendet. In beiden Fällen wird ein 2 Stutzen-Schraubverschluss aus Polypropylen und Edelstahl zum Abdichten der Flasche eingesetzt. Einer der beiden Stutzen dient erneut zum Druckausgleich des Behälters. Der zweite Stutzen ist mit einem der zwei Permeatausgänge des Hohlfasermodule verbunden. Der gesamte Permeatbehälter steht auf einer Tischwaage (4.1.6.2 Permeatwaage).

### 4.1.3 Hohlfasermodule

Das in dieser Masterarbeit verwendete Hohlfasermodule (MIDIKROS, Artikel Nr.: D02-E750-10-N) wurde bei Repligen (Ravensburg, Deutschland) in Auftrag gegeben. Die technischen Daten des Hohlfasermoduls sind der folgenden Tabelle zu entnehmen.

**Tabelle 4.1: Technische Daten des verwendeten Hohlfasermoduls.**

Eigenschaft	Wert
MWCO	750 kDa
Filtermaterial	Polyethersulfon (mPES)
Filterfläche	75 cm <sup>2</sup>
Faseranzahl	12
Faser ID	1.0 mm
Gesamte Länge	250 mm
Effektive Länge	200 mm

Der Innendurchmesser von 1 mm wurde gewählt, um ein Verblocken der Fasern durch Zellansammlungen zu verhindern. Der gewählte *Molecular Weight Cut-off* (MWCO) der Membranporen soll ein Zurückhalten von Zellen sicherstellen und gleichzeitig im Vergleich zu größeren Poren eine geringere Tendenz zum Fouling aufweisen (s. 2.1.2.2.3 Deckschichtbildung und Fouling). Das Modul ist auf einen maximalen Druck von 2 bar ausgelegt. Beide Enden der Membran wurden durch den Hersteller abgedichtet und gehen über auf eine weibliche Luer-Lock Verbindung (s. Abbildung 4.5). Das Modul besitzt zwei Permeatausgänge, welche ebenso abgedichtet sind und in eine weibliche Luer-Lock Verbindung übergehen. Mit Hilfe männlicher Luer-Lock-Schlauchadapter sind die retentatseitigen Ausgänge sowie ein Permeatausgang an das Schlauchsystem der Anlage gekoppelt. Der zweite Permeatausgang ist mit einem Luer-Lock Blindstopfen abgedichtet. Ein Stativ und mehrere Stativklemmen halten das Hohlfasermodule in der gewünschten Position.

#### 4.1.4 Druckmonitor und mobile Drucksensoren

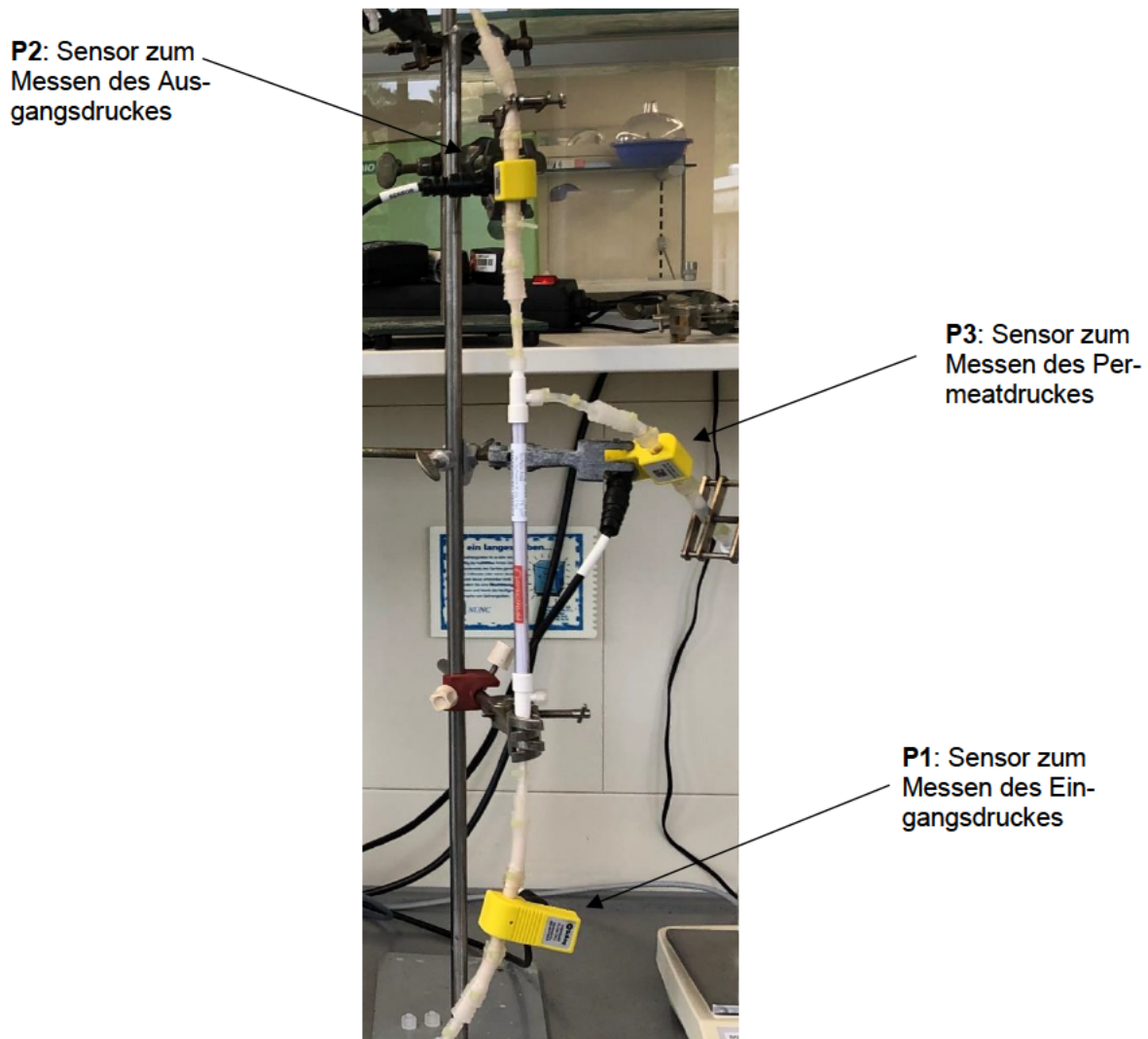
Zur Bestimmung der transmembranen Druckdifferenz über den Filter sind drei mobile Drucksensoren der Firma SciLog in der TFF-Anlage verbaut. Es handelt sich jeweils um ein druckelektrisches Sensorelement aus medizinischem Silikon eingeschlossen in einer Flusszelle aus Polysulfon mit zwei Schlauchanschlüssen. Mit Hilfe von Schlauchadaptern sind diese dem Schlauchsystem des Filtermoduleinganges (P1), des Filtermodulaustrages (P2) und des verwendeten Permeataustrages (P3) zwischengeschaltet (Abbildung 4.5). Direkt hinter den Drucksensoren am Filtermodulaustrag und am Permeataustrag befinden sich die retentatseitige und die permeatseitige Schlauchklemme. Die Drucksensoren sind teils mit einer Stativklemme an einem Stativ befestigt. Ein *Switchcraft* 9 Pin-Kabel verbindet die Sensoren mit dem dazugehörigen Druckmonitor (Abbildung 4.4).



**Abbildung 4.4: Abbildung des Druckmonitors der mobilen Drucksensoren.** Links ist die Vorderseite inklusive Bildschirm dargestellt. Rechts zu erkennen sind die Anschlüsse auf der Rückseite des Monitors (Parker Hannifin Corporation, 2021).

Der Druckmonitor dient zur Anzeige der gemessenen Drücke und des intern ermittelten TMP, sowie zur Stromversorgung der Sensoren. Des Weiteren können verschiedene Einstellungen bezüglich der Messung und der Kalibrierung der Sensoren vorgenommen werden. Der Druckmonitor besitzt zur digitalen Kommunikation eine RS232-Schnittstelle (9-polige D-Steckbuchse) über die die Druckdaten versendet werden können.





**Abbildung 4.5: Nahaufnahme des verbauten Hohlfasermoduls und der mobilen Drucksensoren.** Die Abbildung zeigt, wie das Hohlfasermodul über Luer-Lock Schlauchadapter an das Schlauchsystem der Anlage verbunden und mit Hilfe von Stativklammern fest positioniert ist. Ebenso sind die mobilen Drucksensoren am Filtermoduleingang, -ausgang und Permeatausgang sowie ihre Schlauchverbindungen zu erkennen.

## 4.1.5 Pumpen und Schläuche

### 4.1.5.1 Feedpumpe

Um Schergeschwindigkeiten von bis zu  $10.000 \text{ s}^{-1}$  in den Fasern des ausgewählten Hohlfasermoduls bei der Zirkulation des Retentats erzeugen zu können, sind Flussraten von bis zu  $0,707 \text{ L/min}$  notwendig (Abbildung 8.1). Um dies zu ermöglichen, wurde die peristaltische Schlauchpumpe 323Du, sowie 2 Einkanal-Pumpenköpfe mit 3 Rollen und ein platinveredelter Silikon-Schlauch (3,2 mm ID und 1,6 mm Wandstärke) bei Watson-Marlow GmbH (Rommerskirchen, Deutschland) in Auftrag gegeben. Die Pumpe besitzt eine digitale Drehzahlregelung von 3 RPM bis 400 RPM in Schritten von 1 RPM und ist sofort in ihrer Drehrichtung umkehrbar. Beide Pumpenköpfe werden versetzt an der Pumpe angebracht, um eine pulsationsarme Förderung der Zellsuspension zu ermöglichen. Die Enden der dabei verwendeten Schläuche werden über zwei Y-Schlauchadapter zum einen mit dem Feedbehälter und zum anderen mit dem Filtermoduleingang verbunden (Abbildung 4.6).



**Abbildung 4.6: Nahaufnahme der Feedpumpe.** Die Abbildung zeigt die Feedpumpe mit beiden Pumpenköpfen und die Anbindung an das Schlauchsystem.

Die Pumpe besitzt zur digitalen Kommunikation eine RS232-Schnittstelle (9-polige D-Steckbuchse). Diese wird zur Steuerung der Pumpe über das Computer Programm der Anlage verwendet. Dafür wird ein serielles Kabel (9-polige RS232 Stecker) zunächst auf die korrekte Pin-Belegung der Pumpenschnittstelle (Abbildung 4.7) um gelötet.

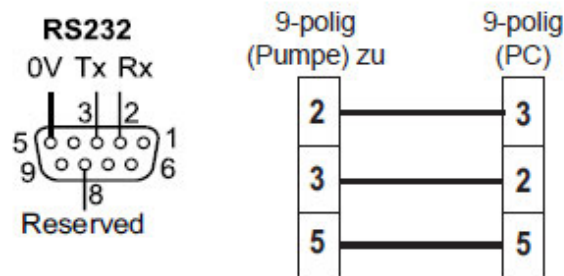


Abbildung 4.7: Pin-Belegung der seriellen Schnittstelle und des notwendigen seriellen Kabels der Watson Marlow Pumpen.

#### 4.1.5.2 Diafiltrationspumpe

Die Aufgabe der Diafiltrationspumpe liegt darin, das Volumen in dem Feedbehälter bei einem Diafiltrationsprozess durch Zugabe eines Diafiltrationspuffers aus einem Puffer-Reservoir (Diafiltrationsbehälter) konstant zu halten. Die dafür benötigten Flussraten entsprechen dem Permeatfluss durch den Filter in den Permeatbehälter. Hierfür wurde ebenfalls eine peristaltische Schlauchpumpe 323Du, sowie ein Einkanal-Pumpenkopf mit 3 Rollen und ein platinveredelter Silikon-Schlauch (1,6 mm ID und 1,6 mm Wandstärke) bei Watson-Marlow in Auftrag gegeben. Die Kenndaten der Pumpe, sowie die digitale Kommunikation mit dem Laptop ist identisch mit der von der Feedpumpe (4.1.5.1 Feedpumpe). Der Pumpenkopf wird auf die Welle der Pumpe gesetzt. Der verwendete Schlauch ist zu einer Seite über einen Schlauchadapter und einem kurzen Schlauchstück mit einem Stutzen des Schraubverschlusses des Diafiltrationsbehälters verbunden (4.1.2.2 Diafiltrationsbehälter). Zur anderen Seite ist der Schlauch über zwei Schlauchadapter und zwei kurzen Schlauchstücken mit einem Stutzen des Schraubverschlusses des Feedbehälters verbunden (4.1.2.1 Feedbehälter).

## 4.1.6 Waagen

### 4.1.6.1 Feedwaage

Das Erfassen des Gewichtes des Retentats ist für die Überwachung über den Laptop sowie für einen automatisierten Prozessverlauf essenziell. Hierfür wird die Tischwaage TE6101 von Sartorius verwendet. Sie verfügt über eine Ablesegenauigkeit von 0,1 g und einen maximalen Wägebereich von 6100 g. Eine RS232-Schnittstelle (25-polige D-Steckbuchse) auf der Rückseite der Waage ermöglicht den Datenaustausch der gemessenen Werte sowie das Senden von Befehlen an die Waage. Um eine homogene Verteilung der Zellen in der Suspension im Feedbehälter sicherzustellen, wird zusätzlich ein Magnetrührer der Firma IKA auf der Feedwaage platziert (Abbildung 4.8). Der Feedbehälter befindet sich im laufenden Prozesse auf diesem Magnetrührer.



Abbildung 4.8: Nahaufnahme der Feedwaage und des darauf platzierten Magnetrührers.

### 4.1.6.2 Permeatwaage

Für die Überwachung und Dokumentation des Permeatflusses durch den Filter ist die Messung des Permeatgewichts notwendig. Eine der Feedwaage baugleiche Tischwaage (Sartorius, TE6101) wird dafür unter den Permeatbehälter platziert. Auch die digitale Verbindung mit dem Laptop der Anlage ist identisch mit der von der Feedwaage.

### 4.1.7 Tangentialflussfiltrationsanlage

Durch den oben genannten Zusammenbau der einzelnen Bestandteile entsteht die funktionsfähige Tangentialflussfiltrationsanlage (Abbildung 4.9). Im laufenden Prozess drehen sich die Pumpenköpfe im Uhrzeigersinn und pumpen die Zellsuspension durch das Hohlfasermodule zurück in den Feedbehälter. Die Drucksensoren 1 und 2 sowie das 3-Wege-Ventil werden dabei durchströmt. Das Fluid, welches die Membran durchdringt, passiert den Drucksensor 3 und gelangt in den Permeatbehälter. Bei einer Diafiltration pumpt die Diafiltrationspumpe zusätzlich Diafiltrationspuffer aus dem Diafiltrationsbehälter in den Feedbehälter.



**Abbildung 4.9: Tangentialflussfiltrationsanlage.** Zu erkennen sind die einzelnen Baubestandteile: Permeatwaage, Permeatbehälter, Hohlfasermodule, Drucksensoren, Druckmonitor, retentatseitige und permeatseitige Schlauchklemme, 3-Wege-Ventil, Feedpumpe, Feedwaage, Magnetrührer, Feedbehälter und die Diafiltrationspumpe. Die Diafiltrationspumpe sowie der Diafiltrationsbehälter (nicht abgebildet) sind in diesem Bild abweichend vom Text nicht angeschlossen, da dieses Bild während einer laufenden Konzentrierung gemacht wurde. Der Laptop der Anlage befindet sich auf einem Schreibtisch gegenüber von der hier abgebildeten TFF-Anlage.

Für die Verbindungsleitungen wurden Silikonschläuche mit einem Innendurchmesser von 4 mm und einer Wandstärke von 1 mm verwendet, um das Totvolumen der Anlage so gering wie möglich zu halten. Die oben erwähnten Schlauchstücke und Schlauchadapter ermöglichen die Verbindung auf die Stutzen des Feedbehälters (ID: 8 mm) und auf die Schlauchanschlüsse der Drucksensoren (ID: 8 mm). Diese Schläuche bestehen aus Silikon und besitzen einen Innendurchmesser von 8 mm und eine Wandstärke von 1,6 mm. Um ein Abplatzen der Schläuche zu verhindern, werden alle Schlauchverbindungen mit Kabelbindern befestigt.

## 4.2 Entwicklung der graphischen Benutzeroberfläche

Die graphische Benutzeroberfläche (GUI) wird in der Softwareentwicklungsumgebung "App Designer" des Programms Matlab® R2021a vom Hersteller The MathWorks, Inc. erstellt. Die dafür benötigte Lizenz wurde von der HAW Hamburg zur Verfügung gestellt.

### 4.2.1 Konzept

Die Anwendung soll dem Endbenutzer ermöglichen, die in Kapitel 4.1 beschriebene TFF-Anlage von dem dazugehörigen Laptop zu bedienen und einen Filtrationsprozess zu planen und durchzuführen. Dabei wird das GUI hinsichtlich der Einflussfaktoren von guter Übersicht, simplem und effizientem Handling sowie verständlichem Design konzipiert. Das Konzept wird zum einen danach ausgerichtet, eine einfache Planung und Automatisierung von Konzentrations- oder Diafiltrationsprozessen zu ermöglichen. Zum anderen steht eine klar überschaubare, aber ausreichend detaillierte Überwachung des Prozesses im Fokus der Programmierung. Dabei soll dem Endbenutzer eine Wahl in der Gestaltung der Überwachungskomponenten aber auch in der Dokumentation der Messdaten erlaubt werden. Zuverlässigkeit sowie Robustheit gegenüber vergessenen oder fehlerhaften Eingaben des Endbenutzers soll gewährleistet werden.

### 4.2.2 Layout der Anwendung

Die Grundstruktur des GUI besteht aus einer Einteilung in 7 Reiter:

- *Hardware Setup*
- *Calibration*
- *Process Setup*
- *Process Overview*
- *Tables*
- *Graphs*
- *Save Data*

Die Aufteilung in einzelne Reiter wurde gewählt, um die Softwarekomponenten zu kategorisieren und damit zu einer übersichtlichen Navigation durch das GUI beizutragen.

#### 4.2.2.1 Reiter 1: *Hardware Setup*

Im ersten Reiter *Hardware Setup* (Abbildung 4.10) wird die softwareseitige serielle Verbindung zwischen Feedpumpe, Diafiltrationspumpe, Feedwaage, Permeatwaage sowie Druckmonitor und dem Laptop der Anlage ermöglicht. Für jedes Gerät der TFF-Anlage gibt es ein beschriftetes Textfeld (Nr. 1). Die COM-Ports, an denen die jeweiligen Geräte am Laptop angeschlossen sind, müssen in das dazugehörige Textfeld eingetragen werden (z. B. "COM1").

Hardware Connection

Com Port Feed Pump:	<input type="text"/>
Com Port Pressure Monitor:	<input type="text"/>
Com Port Feed Scale:	<input type="text"/> 1
Com Port Permeat Scale:	<input type="text"/>
Com Port Diafiltration Pump:	<input type="text"/>

2

Ensure that every component has a green check!

**Abbildung 4.10: Ausschnitt des Reiters *Hardware Setup*.** Die relevanten Softwarekomponenten (Tabelle 4.2) des Reiters sind mit roten Zahlen markiert und nummeriert.

Anschließend wird durch das Klicken des Buttons "*Initialization of Com Ports*" (Nr. 2) eine *Callback*-Funktion ausgelöst, welche die seriellen Verbindungen zwischen den Geräten und der Anwendung herstellt.



**Tabelle 4.2: Softwarekomponenten des Reiters *Hardware Setup*.** Dazugehörige *Callback* Aktionen und Nummern der Bildschirmaufnahme sind mit aufgelistet.

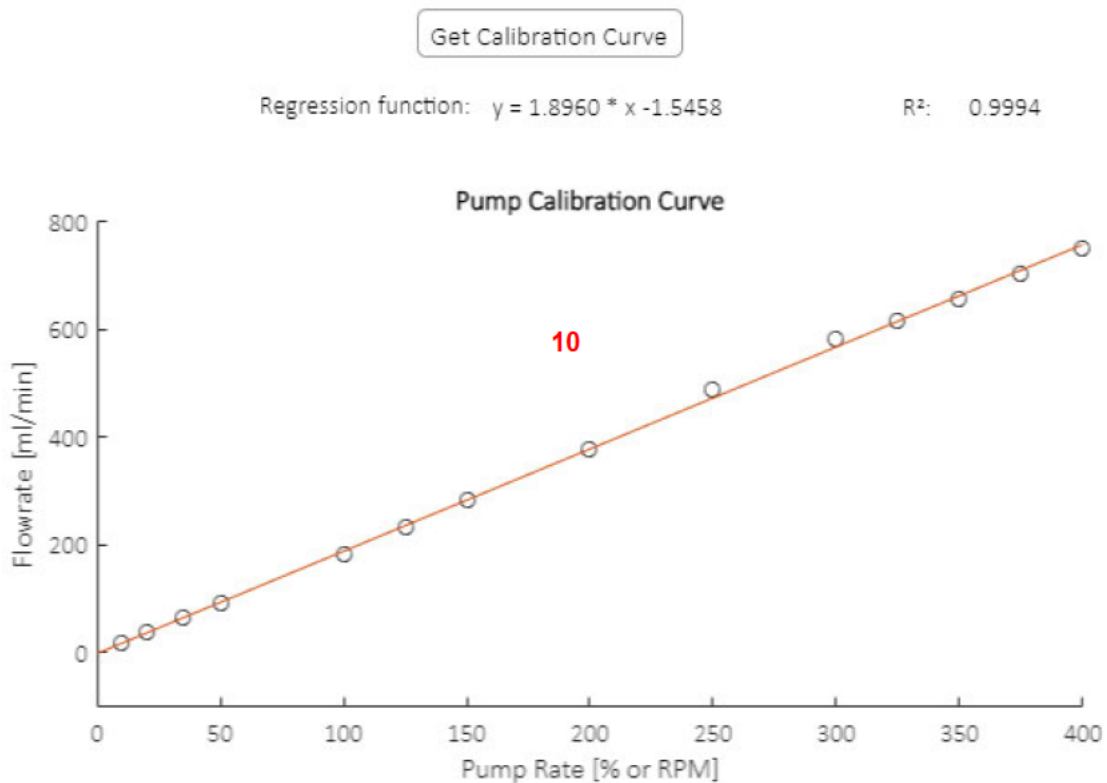
Num- mer	Name	<i>Callback</i> Aktion
1	ComPortFeedPumpEditField	-
1	ComPortPressureMonitorEditField	-
1	ComPortFeedScaleEditField	-
1	ComPortPermeatScaleEditField	-
1	ComPortDiafiltrationPumpEditField	-
2	InitializationofComPortsButton	<i>ButtonPushed</i>

#### 4.2.2.2 Reiter 2: Calibration

Im zweiten Reiter, *Calibration* (Abbildung 4.11), kann die Feedpumpe der TFF-Anlage kalibriert und die dazugehörige Kalibriergerade abgebildet werden. Im Zahlenfeld *Calibration Point* wird die Nummer des jeweiligen Kalibrierpunktes notiert (Nr. 1). Das Dropdown-Menü (Nr. 2) ermöglicht die Wahl zwischen RPM oder Pumpleistung in % als Einheit der x-Werte der Kalibrierung. In das darauffolgende Zahlenfeld (Nr. 3) wird der x-Wert (RPM- oder Pumpleistungswert) des derzeitigen Kalibrierungspunktes eingegeben. Durch Anklicken des Buttons "*Start Pumping*" (Nr. 4) bzw. "*Stop Pumping*" (Nr. 5) startet die Feedpumpe mit dem im Zahlenfeld eingegebenen Wert bzw. stoppt sie erneut. Im Zahlenfeld *VolumeafterrunEditField* (Nr. 6) wird das durch die Pumpe in 60 s geförderte Volumen eingetragen und der jeweilige Kalibrierungspunkt durch Klicken des Buttons "*Save Calibration Point*" (Nr. 7) gespeichert. Am Ende der Kalibrierung kann die Kalibriergerade, ihre Regressionsfunktion sowie das Bestimmtheitsmaß (Nr. 10) durch Klicken des Buttons "*Get Calibration Curve*" (Nr. 8) abgebildet werden (Abbildung 4.12). Das Abbilden der Kalibriergerade der zuletzt durchgeführten Kalibrierung eines vorherigen Anwendungsstart wird durch Klicken des Buttons "*Load Calibration Curve*" (Nr. 9) ermöglicht.



Abbildung 4.11: Bildschirmaufnahme 1 des Reiters *Calibration*. Die relevanten Softwarekomponenten des Reiters (Tabelle 4.3) sind mit roten Zahlen markiert und nummeriert.



**Abbildung 4.12: Bildschirmaufnahme 2 des Reiters Calibration.** Die relevanten Softwarekomponenten des Reiters (Tabelle 4.3) sind mit roten Zahlen markiert und nummeriert.

**Tabelle 4.3: Softwarekomponenten des Reiters Calibration.** Dazugehörige *Callback* Aktionen und Nummern der Bildschirmaufnahme sind aufgelistet.

<b>Nummer</b>	<b>Name</b>	<b>Callback Aktion</b>
1	CalibrationPointEditField	-
2	PumpRateDropDown_4	-
3	PumpRateValueEditField_4	<i>ValueChanged</i>
4	StartPumpingButton	<i>ButtonPushed</i>
5	StopPumpingButton	<i>ButtonPushed</i>
6	VolumeafterrunEditField	-
7	SaveCalibrationPointButton	<i>ButtonPushed</i>
8	GetCalibrationCurveButton	<i>ButtonPushed</i>
9	LoadcalibrationcurveButton	<i>ButtonPushed</i>
10	UIAxes_PumpCalibration	-

### 4.2.2.3 Reiter 3: *Process Setup*

Der Reiter *Process Setup* (Abbildung 4.13) vereint alle Softwarekomponenten (Tabelle 4.4), welche für die Planung eines Filtrationsversuches, die Vorbereitung der Anlage, sowie die Speicherung der Messdaten benötigt werden. Zur Identifikation des Prozesses kann als erstes der Prozessname, der Name des Operators und ein Kommentar zum Prozess in die vorgesehenen Textfelder (Nr. 1-3) notiert werden. Zudem besitzt dieser Reiter sowie die Reiter *Process Overview*, *Tables* und *Graphs* in der oberen rechten Ecke des Reiters eine Lampe, die den Prozess Status anzeigt (grau – noch nicht gestarteter oder gestoppter Prozess, grün - laufender Prozess, gelb - pausierter Prozess). Des Weiteren ist der Reiter in 3 Abschnitte aufgeteilt.

Der Abschnitt *Create Trial Recipe* zeigt die verschiedenen Möglichkeiten und Einstellungen eines Filtrationsprozesses. In der *FiltrationModeButtonGroup* (Nr. 11) kann der Filtrationsmodus gewählt werden. Neben einer einfachen Konzentrierung (C) oder Diafiltration (D) können zusätzlich Kombinationen (C/D oder C/D/C) aus diesen beiden Modi durchgeführt werden. Eine weitere *Button Group* (Nr. 12) lässt dem Endbenutzer die Wahl zwischen einem automatischen oder manuellen Übergang der einzelnen Filtrationsschritte bei den Multifiltrationsmodi C/D und C/D/C. Die Dropdown-Menüs 13, 16, 19 und Zahlenfelder 14, 17, 20 legen die Endpunkte der Filtrationsschritte fest (Endpunkttypen einer Konzentrierung: Konzentrationsfaktor CF, Flux-Wert, Zeit, Feedgewicht; Endpunkttypen einer Diafiltration: Diafiltrationsvolumen DV, Zeit, Permeatgewicht). In die Zahlenfelder 15 und 18 werden die jeweiligen Start Volumina des Feedbehälters eingetragen. *Button Group* 24 legt fest, ob ein Hohlfasermodule oder eine Filterkassette in der TFF-Anlage verbaut ist. Da in dieser Masterarbeit ausschließlich ein Hohlfasermodule verwendet wird, wurde auch softwareseitig nur auf die Filtereinstellungen eines Hohlfasermodule eingegangen. Die Faseranzahl, der Faserinnen-durchmesser und die Filterfläche müssen in die Zahlenfeldern 22-24 eingetragen werden.

Es folgen die Einstellungen der beiden Pumpen. Die *FeedPumpSetupButtonGroup* (Nr. 25) lässt den Endbenutzer entscheiden, ob die Leistung der Feedpumpe als Flussrate in ml/min oder als Schergeschwindigkeit in  $s^{-1}$  angegeben werden soll. In Zahlenfeld 26 kann dieser Wert entweder unter "*Flow Rate*" bzw. unter "*Shear Rate*" eingetragen werden. Im laufenden Prozess kann der Wert durch Klicken des Buttons "*Change Flowrate at process*" (Nr. 27) angepasst werden. Zahlenfeld 28 erlaubt das

Eintragen des Innendurchmessers der Feed-Pumpenschläuche und *Button Group 29* die Wahl der Feed-Pumprichtung. Die *DiafiltrationPumpControllerSetupButtonGroup* (Nr. 30) ermöglicht bei eingestellter Diafiltration die Wahl zwischen manueller und geregelter Flussrate der Diafiltrationspumpe. In Zahlenfeld 32 kann der Innendurchmesser des Pumpenschlauches sowie im geregelten Fall in Zahlenfeld 31 der RPM-Startwert und in Zahlenfeld 33 die maximale Drehzahl der Diafiltrationspumpe festgelegt werden. Genauere Reglereinstellungen sind in den Zahlenfeldern 34-37 einzugeben.

Der *Auto Save* Abschnitt besitzt lediglich den Button *SearchpathButton* (Nr. 38). Durch Anklicken öffnet sich ein Dialogfenster, in welchem der Pfad zur automatischen Speicherung der einzelnen Prozessdaten während des Prozesses ausgesucht werden kann. Die Daten werden in .txt-Dateien mit automatisch generierten Namen gespeichert. Neben den Prozessdaten (Behältergewichte, Druckdaten, Flux-Werte, Flussraten, Konzentrationsfaktoren oder Diafiltrationsvolumina) wird eine zusätzliche Parameter-Datei erstellt. Dort werden folgende Informationen über den Prozess zur Dokumentation abgespeichert: Name des Prozesses, Startvolumen der Konzentrierung, Startvolumen der Diafiltration, eingestellte Flussrate und Schergeschwindigkeit, Innendurchmesser der Pumpenschläuche, Eigenschaften des Hohlfasermoduls (Faseranzahl, Faserinnendurchmesser, Filterfläche).

Der Abschnitt *Plant Preparation* besitzt wichtige Elemente für die Vorbereitung der TFF-Anlage. Mit Anklicken des Auswahlkästchens *CheckBox\_UseConcStartValue* (Nr. 4) kann angegeben werden, dass kein manuelles Trieren der Feedwaage stattgefunden hat. Mit Hilfe des Buttons *AutoTareButton* (Nr. 6) kann in diesem Fall durch vorherige Eingabe des Startvolumens des Feedbehälters in Zahlenfeld 5 und durch Klicken des Buttons ein nachträgliches internes Trieren des Feed-Gewichtes durchgeführt werden. Des Weiteren kann eine Berechnung des Totvolumens der Anlage mit einem *Fill Up* durch Anklicken des Buttons "Start Fill Up" und vorheriger Eingabe der Pumpendrehzahl in Zahlenfeld 7 ausgeführt werden. Das permeatseitige Totvolumen muss manuell bestimmt werden und in Zahlenfeld 10 eingetragen werden.

MATLAB App

XFlow Filtration Plant

Hardware Setup | Calibration | **Process Setup** | Process Overview | Tables | Graphs | Save Data

Process Name:  Operator:  Comment:  Process State

---

**Plant Preparation**

1. Belated Taring:  **4** Start volume Feed [ml]   **5**

**6** Auto Tare

Offset Value  [g]

2. System Fill Up: Pump Rate [RPM]:  **7**

**8** Start Fill Up **9** Stop Fill Up

Dead volume  [ml]

Dead volume permeate site  [ml] **10**

---

**Create Trial Recipe**

Filtration Mode **11**  
 Concentration  Diafiltration  C/D  C/D/C

Change Mode **12**  
 Automatically  Manually

1st Concentration **13** End Point Type Concentration Factor  End Point Value  Start volume Feed [ml]  **15**

1st Diafiltration **16** End Point Type Diafiltration Volume [DV]  End Point Value   **18**

2nd Concentration **19** End Point Type Concentration Factor  End Point Value

Every field must be filled out!

---

**Filter Setup** **21**

Hollow Fiber  Cassette

Fiber Count  **22**

Fiber ID [mm]  **23**

Surface Area [cm<sup>2</sup>]  **24**

Every field must be filled out!

---

**Feed Pump Setup** **25**

Set Flowrate  Set Shear

Flow Rate [ml/min]  Shear Rate [1/s]

Every field must be filled out!

**Change Flowrate at process** **27**

Tubing Size (ID in mm)  **28**

Pump Direction  Clockwise **29**  Counter Clockwise

---

**Diafiltration Pump/Controller Setup** **30**

Controlled Flowrate  Manually

Starting Point [rpm]  Tubing Size (ID in mm)  **31** **32**

max. Pump Rate:  [rpm] **33**

max. possible Error (emax):  [ml] **34**

Time Constant Scale:  [s] **35**

Kp:  **36** Ki:  **37**

**Auto-Save!**

Start Data Storage by choosing the path where the process data are auto-saved:

**38**  (Name is generated automatically)

Abbildung 4.13: Bildschirmaufnahme des Reiters **Process Setup**. Die relevanten Softwarekomponenten des Reiters (Tabelle 4.4) sind mit roten Zahlen markiert und nummeriert.

**Tabelle 4.4: Softwarekomponenten des Reiters *Process Setup*.** Dazugehörige *Callback* Aktionen und Nummern der Bildschirmaufnahme sind mit aufgelistet.

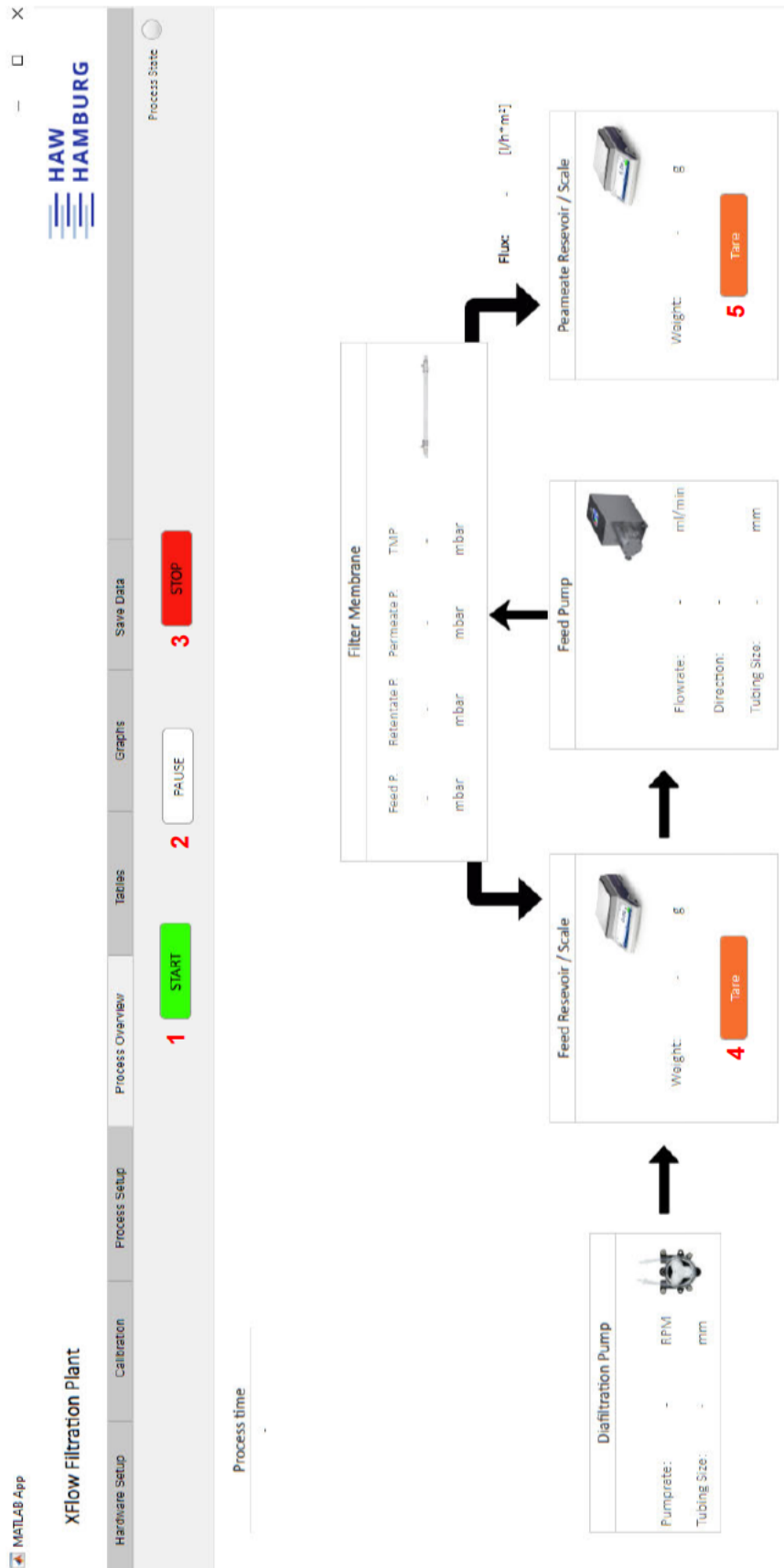
<b>Nummer</b>	<b>Name</b>	<b>Callback Aktion</b>
1	ProcessNameEditField	-
2	OperatorEditField	-
3	CommentTextArea	-
4	CheckBox_UseConcStartValue	<i>ValueChanged</i>
5	EditField_StartVolumen_forAutoTaring	-
6	AutoTareButton	<i>ButtonPushed</i>
7	PumpRateRPMEditField	-
8	StartFillUpButton	<i>ButtonPushed</i>
9	StopFillUpButton	<i>ButtonPushed</i>
10	DeadvolumepermeatesiteEditField	-
11	FiltrationModeButtonGroup	<i>SelectionChanged</i>
12	ChangeModeButtonGroup	-
13	EndPointTypeDropDown_1st_Concentration	<i>ValueChanged</i>
14	EndPointValueEditField_1stConcentration	-
15	EditField_StartVolumen_1stConcentration	-
16	EndPointTypeDropDown_1st_Diafiltration	-
17	EndPointValueEditField_D	-
18	EditField_StartVolumen_1stDiafiltration	-
19	EndPointTypeDropDown_2nd_Concentration	-
20	EndPointValueEditField_C1	-
21	FilterSetupButtonGroup	<i>SelectionChanged</i>
22	FiberCountEditField	-
23	FiberIDmmEditField	<i>ValueChanged</i>
24	SurfaceAreacm_HollowFiberEditField	-
25	FeedPumpSetupButtonGroup	<i>SelectionChanged</i>
26	Set_FlowRate_EditField / Set_Shear_EditField_2	<i>ValueChanged</i>
27	ChangeFlowrateatprocessButton	<i>ButtonPushed</i>
28	Tubing_Size_FeedPump_EditField	-
29	PumpDirectionButtonGroup	<i>SelectionChanged</i>
30	DiafiltrationPumpControllerSetupButtonGroup	<i>SelectionChanged</i>
31	Anfangspunkt_Dia_Pumpe_Editfield	-
32	Tubing_Size_DiafiltrationPump_EditField	-
33	maxPumpRateEditField	-
34	maxVolumeemaxEditField	-
35	TimeConstantScaleEditField	-

<b>Nummer</b>	<b>Name</b>	<b>Callback Aktion</b>
36	KpEditField	-
37	KiEditField	-
38	SearchpathButton	<i>ButtonPushed</i>

#### 4.2.2.4 Reiter 4: *Process Overview*

Der Reiter *Process Overview* beinhaltet eine schematische Übersicht der TFF-Anlage und soll diese grafisch veranschaulichen. Die Geräte der Anlage sind hier jeweils in einem Panel mit den wichtigsten Kenn- und Messdaten abgebildet (Abbildung 4.14). Die Prozesszeit sowie der Flux-Wert sind ebenso dargestellt. Dieser Reiter soll dem Endbenutzer einen schnellen und einfachen Überblick der gesamten Anlage, sowie auch essenzieller Eigenschaften der einzelnen Bestandteile ermöglichen. Die Button 1-3 initiieren den Start, eine Pause (oder Fortfahren beim erneuten Anklicken des Buttons) und die Beendigung des Filtrationsprozesses. Mit Hilfe von Button 4 und 5 können die Feed- und Permeatwaage vor oder während des Prozesses tariert werden.





**Abbildung 4.14: Bildschirmaufnahme des Reiters Process Overview.** Die relevanten Softwarekomponenten des Reiters (Tabelle 4.5) sind mit roten Zahlen markiert und nummeriert.

**Tabelle 4.5: Softwarekomponenten des Reiters *Process Overview*.** Dazugehörige *Callback* Aktionen und Nummern der Bildschirmaufnahme sind mit aufgelistet.

<b>Nummer</b>	<b>Name</b>	<b>Callback Aktion</b>
1	STARTButton	<i>ButtonPushed</i>
2	PAUSEButton_Process_Setup	<i>ButtonPushed</i>
2	CONTINUEButton_Process_Setup	<i>ButtonPushed</i>
3	STOPButton_Process_Setup	<i>ButtonPushed</i>
4	TareButton_Feed	<i>ButtonPushed</i>
5	TareButton_Permeat	<i>ButtonPushed</i>

#### 4.2.2.5 Reiter 5: *Tables*

Der *Tables*-Reiter (Abbildung 4.15) dient einer detaillierten tabellarischen Darstellung aller wichtigen Prozessdaten inklusive Prozess- und Operatorname, Datum und Uhrzeit des jeweiligen Messpunktes. Die Messdaten werden in diesem Reiter mit den meisten Nachkommastellen angezeigt. Alle 3 Sekunden aktualisiert sich die Tabelle (Nr. 3) mit neuen Daten. Der Button “*PAUSE*“ (Nr. 1) und der Button “*STOP*“ (Nr. 2) pausieren (oder Fortfahren beim erneuten Anklicken des Buttons) bzw. stoppen den Filtrationsprozess.

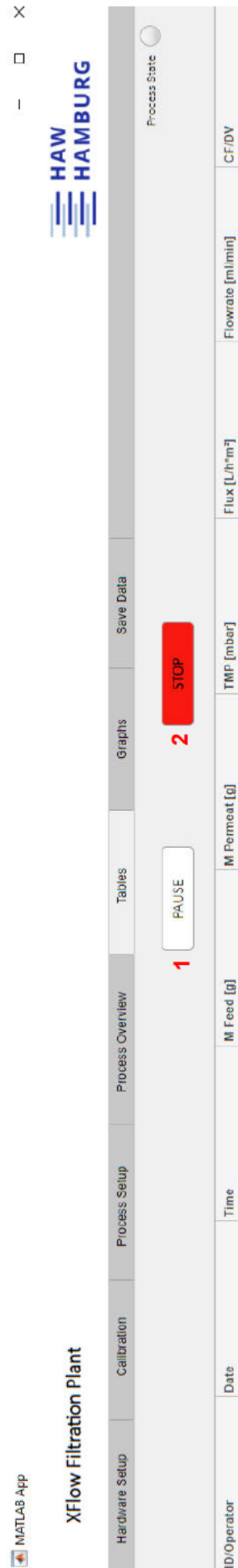


Abbildung 4.15: Bildschirmaufnahme des Reiters *Tables*. Die relevanten Softwarekomponenten des Reiters (Tabelle 4.6) sind mit roten Zahlen markiert und nummeriert.

**Tabelle 4.6: Softwarekomponenten des Reiters *Tables*.** Dazugehörige *Callback* Aktionen und Nummern der Bildschirmaufnahme sind mit aufgelistet.

Nummer	Name	Callback Aktion
1	PAUSEButton_Tables	<i>ButtonPushed</i>
1	CONTINUEButton_Tables	<i>ButtonPushed</i>
2	STOPButton_Tables	<i>ButtonPushed</i>
3	TabelleAlles	-

#### 4.2.2.6 Reiter 6: *Graphs*

Eine grafische Darstellung der Messdaten kann im Reiter *Graphs* gefunden werden. Der Endbenutzer hat mittels einer *Button Group* (Nr. 3) die Wahl zwischen der Anzeige eines großen einzelnen Graphens (Abbildung 4.16) oder mehrerer kleiner Graphen (Abbildung 4.17). Entscheidet sich der Endbenutzer für einen einzelnen Graphen, kann er mit Hilfe eines Dropdown-Menüs (Nr. 4) den zu überwachenden Graphen (Nr. 5) aussuchen: *Weights vs. Time*, *Pressure vs. Time*, *Flow Rate vs. Time*, *Flux vs. Time*. Hat er sich für die Ansicht mehrerer Graphen entschieden, können durch das Aktivieren von Auswahlkästchen (Nr. 6-9) und Anklicken des Buttons "*Show Graphs*" (Nr. 10) die ausgewählten Graphen (Nr. 11-14) gleichzeitig angezeigt werden. Unabhängig von der Auswahl der Anzeige der Graphen ermöglichen die Button 1 und 2 erneut das Pausieren (oder Fortfahren beim erneuten Anklicken des Buttons) bzw. das Stoppen des Filtrationsprozesses.

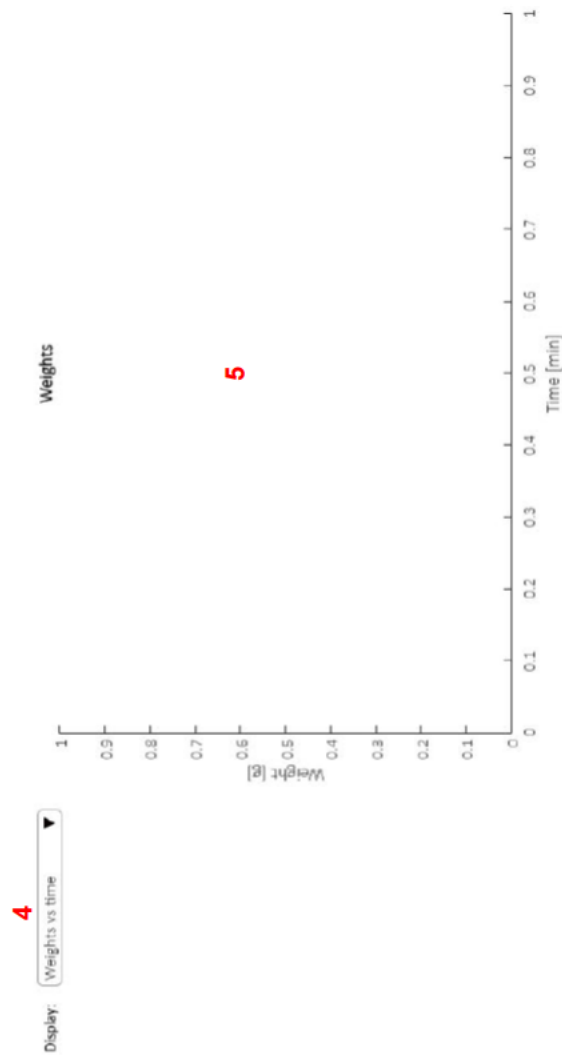
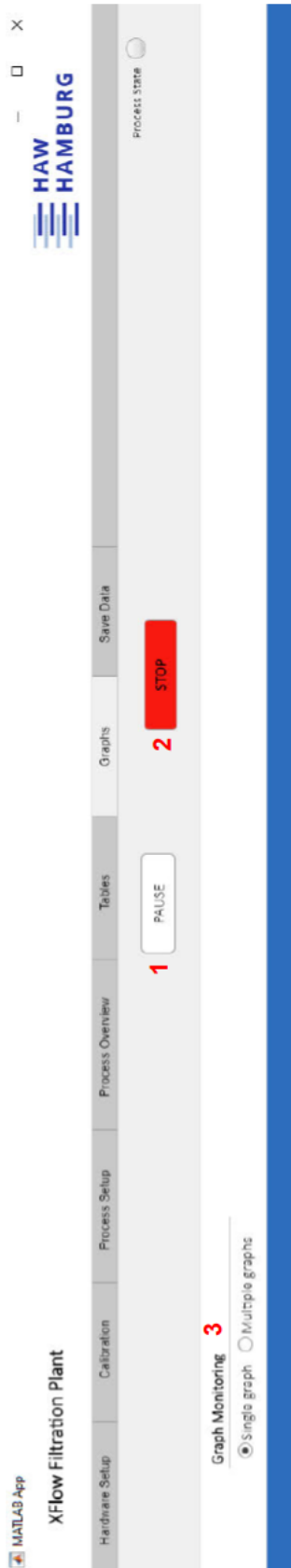
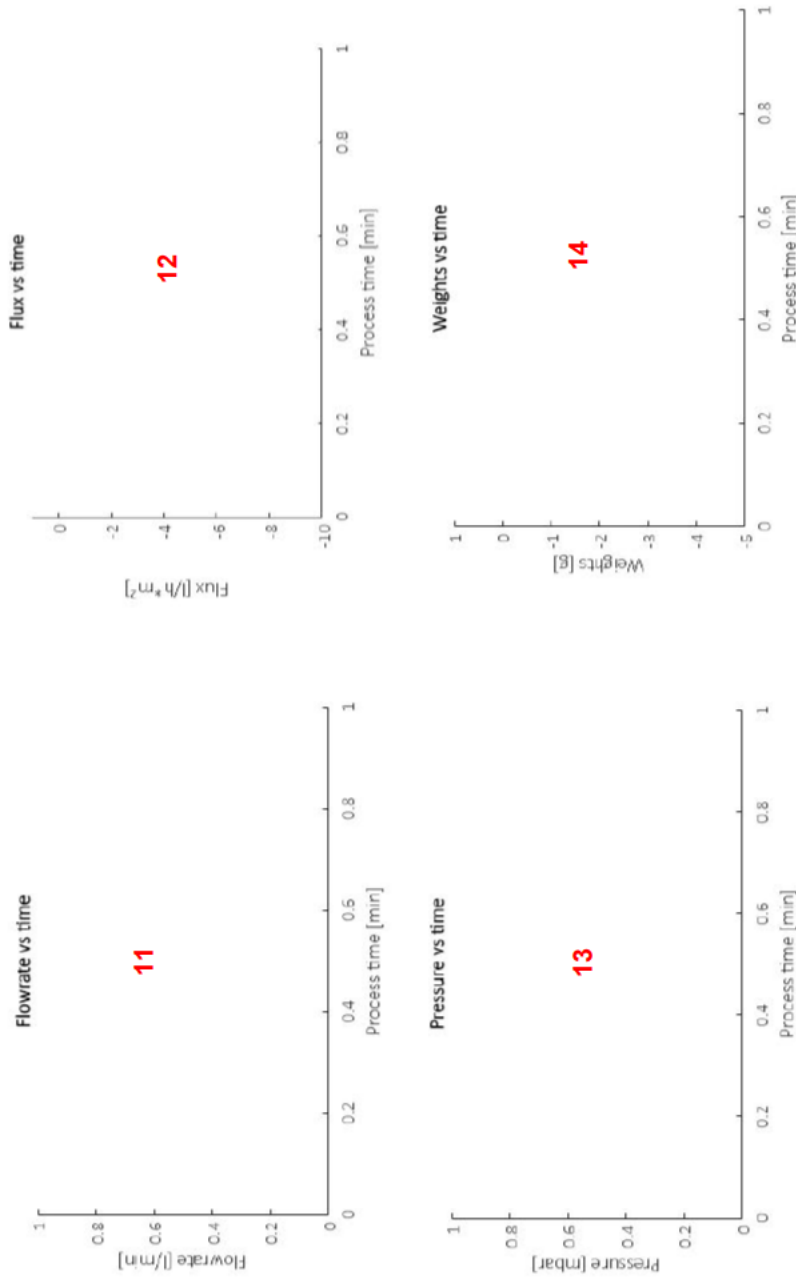


Abbildung 4.16: Bildschirmaufnahme 1 des Reiters **Graphs**. Die relevanten Softwarekomponenten des Reiters (Tabelle 4.7) sind mit roten Zahlen markiert und nummeriert.

- 6 Pressure vs time
- 7 Weights vs time
- 8 Flowrate vs time
- 9 Flux vs time
- 10 Show graphs



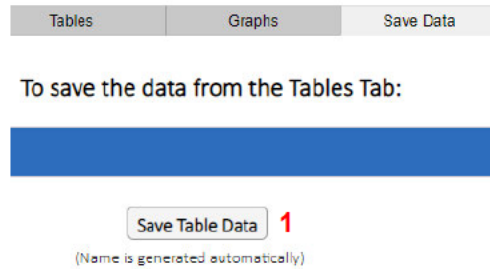
**Abbildung 4.17: Bildschirmaufnahme 2 des Reiters Graphs.** Die relevanten Softwarekomponenten des Reiters (Tabelle 4.7) sind mit roten Zahlen markiert und nummeriert.

**Tabelle 4.7: Softwarekomponenten des Reiters *Graphs*.** Dazugehörige *Callback* Aktionen und Nummern der Bildschirmaufnahme sind mit aufgelistet.

<b>Nummer</b>	<b>Name</b>	<b>Callback Aktion</b>
1	PAUSEButton_Graphs	<i>ButtonPushed</i>
1	CONTINUEButton_Graphs	<i>ButtonPushed</i>
2	STOPButton_Graphs	<i>ButtonPushed</i>
3	GraphMonitoringButtonGroup	<i>SelectionChanged</i>
4	DisplayDropDown	<i>ValueChanged</i>
5	UIAxes_FluxVsTime	-
5	UIAxes_FlowrateVsTime	-
5	UIAxesDruckMonitor	-
5	UIAxes_Waagen	-
6	PressurevstimeCheckBox	<i>ValueChanged</i>
7	WeightsvstimeCheckBox	<i>ValueChanged</i>
8	FlowratevstimeCheckBox	<i>ValueChanged</i>
9	FluxvstimeCheckBox	<i>ValueChanged</i>
10	ShowgraphsButton	<i>ButtonPushed</i>
11	UIAxesMultipleGraphs_FlowrateVsTime	-
12	UIAxesMultipleGraphs_FluxVsTime	-
13	UIAxesMultipleGraphs_PressureVsTime	-
14	UIAxesMultipleGraphs_WeightsVsTime	-

#### 4.2.2.7 Reiter 7: *Save Data*

Der letzte Reiter *Save Data* erlaubt neben dem automatischen Speichern der einzelnen Prozessdaten, das manuelle Abspeichern der Datentabelle aus dem Reiter *Tables*. Durch Anklicken des Buttons “*Save Table Data*“ (Nr. 1) öffnet sich ein Dialogfenster in welchem der Pfad für die Speicherung ausgewählt werden kann. Auch diese Datei wird als .txt-Datei mit einem automatisch generierten Namen gespeichert.



**Abbildung 4.18: Ausschnitt des Reiters Save Data.** Die relevanten Softwarekomponenten des Reiters (Tabelle 4.8) sind mit roten Zahlen markiert und nummeriert.

**Tabelle 4.8: Softwarekomponenten des Reiters Save Data.** Dazugehörige *Callback* Aktionen und Nummern der Bildschirmaufnahme sind mit aufgelistet.

Nummer	Name	Callback Aktion
1	SaveTableDataButton	ButtonPushed

### 4.2.3 Serielle Kommunikation mit der Hardware

Bevor die graphische Benutzeroberfläche in Matlab App Designer<sup>®</sup> programmiert wird, muss eine digitale Kommunikation zwischen dem Laptop der Anlage und den ansteuerbaren Geräten etabliert werden. Hierfür muss neben der physischen Verbindung durch serielle Kabel (4.1 Aufbau der Anlage) eine softwareseitige Verbindung hergestellt werden. Dieses wurde zunächst im Befehlsfenster von Matlab<sup>®</sup> getestet. Durch den Matlab<sup>®</sup>-Befehl `serialport (COMPort,Baudrate,"Timeout",value,"DataBits",value,"StopBits",value,"Parity",value)` inklusive der seriellen Kommunikationseinstellungen des Gerätes erstellt Matlab<sup>®</sup> ein serielles Schnittstellenobjekt und erzeugt automatisch eine Verbindung mit dem angegebenen COM-Port. Dies ermöglicht das Senden von Befehlen an das Gerät, welches an diesen COM-Port angeschlossen ist. Die Befehle müssen, abhängig von den Geräten, dem Schema eines bestimmten Befehlscodes folgen. Dies beinhaltet die Art und Weise, wie bestimmte Befehlsbuch-



staben in den Befehl eingebunden sind und wie der Befehl abgeschlossen wird. Letzteres geschieht in der Regel durch einen sogenannten Terminator. Dieser wird Matlab® mit Hilfe des Befehls `configureTerminator(SerialObject, Terminator)` mitgeteilt. Das Senden von Befehlen wird durch die Matlab®-Anweisung `writeline(SerialObject, Command)` ausgeführt. Hierbei muss der `Command` zunächst immer als String deklariert werden. Dies realisiert der Befehl `sprintf('CommandString')`. Die Kommunikationseinstellungen und das Kommunikationsprotokoll (Befehlscode) der jeweiligen Geräte (Tabelle 4.9) konnten den Handbüchern entnommen werden. Da der Druckmonitor nur für die Datenaufnahme verwendet wird und seine Messdaten automatisch in einem 5-Sekunden-Takt gesendet werden, ist hier das Übermitteln von Befehlen nicht notwendig.

**Tabelle 4.9: Serielle Kommunikationseinstellungen und -protokolle von den Geräten der TFF-Anlage.** Das X in dem Befehlscode stellt ein beliebigen Befehlsbuchstabe dar. CR steht für *Carriage return* (deu.: Wagenrücklauf), LF für *Line feed* (deu.: Zeilenvorschub) und Esc für Escape-Taste.

Geräte	RS232 Einstellungen	Wert
<b>Druckmonitor</b>	Baudrate	9600
	Zeitüberschreitung [s]	10
	Stoppbit	8
	Datenbit	1
	Parität	keine
	Befehlscode	-
<b>Feed- und Permeatwaage</b>	Baudrate	9600
	Zeitüberschreitung [s]	10
	Stoppbit	7
	Datenbit	1
	Parität	gerade
	Befehlscode	Esc X CR LF
<b>Feed- und Diafiltrationspumpe</b>	Baudrate	9600
	Zeitüberschreitung [s]	10
	Stoppbit	8
	Datenbit	2
	Parität	keine
	Befehlscode	1 X CR

## 4.2.4 Implementierung

### 4.2.4.1 Globale Variablen

Die globalen Variablen der Anwendung werden zu Beginn des Programmcodes einmalig deklariert. Es sind Variablen, auf die ohne erneute Deklaration an jeder Stelle des gesamten Programmcodes zugegriffen werden kann. Sie werden beim Zugriff, wobei das Lesen als auch das Überschreiben der Variable erlaubt ist, in Matlab App Designer® mit `app.VariableName` aufgerufen. Auf Variablen, welche innerhalb einer Funktion deklariert werden, kann lediglich innerhalb dieser Funktion zugegriffen werden und nicht an anderen Stellen des Programmcodes. Diese werden lokale Variablen genannt. In der Tabelle 8.4 (s. Anhang) sind die globalen Variablen der Anwendung und ihre Beschreibungen in alphabetischer Reihenfolge aufgelistet.

### 4.2.4.2 Startup-Funktion

Eine *Startup*-Funktion ist eine *Callback*-Funktion eines Programmcodes, welche beim Start des Programmes ausgeführt wird. In Matlab® werden beim Anwendungsstart zunächst alle Softwarekomponenten erstellt und registriert, woraufhin der Befehl `runStartupFcn(app, @startupFcn)` das Durchführen der *Startup*-Funktion auslöst.

Die *Startup*-Funktion dieses GUI sorgt hauptsächlich dafür, dass globale Variablen, welche wichtig für den Programmablauf oder anstehende Berechnungen sind, Initialwerte zugewiesen bekommen. In den meisten Fällen werden die Werte der Variablen mit Nullen gefüllt (Abbildung 4.19). Ebenso werden feste negative Limits der y-Achsen einzelner Graphen definiert, um dort die automatische Achsenskalierung von Matlab® zu deaktivieren. Diese kontinuierlichen Skalierungsänderungen erschweren deutlich eine übersichtliche Überwachung der Graphen. Zuletzt wird eine Zustandsladungs-Funktion `app.loadState` aufgerufen. Im folgenden Abschnitt wird die Funktion und ihre Implementierung näher erläutert (4.2.4.3 Funktion zur Zustandsspeicherung und -ladung).

```
% Code that executes after component creation
function startupFcn(app)
    clc;

    app.Process_status      = 0;

    %TrendGrafik
    app.trendData            = [0,0,0,0];
    app.trendDataTime       = [0,0,0];
    app.trendDataDruck      = [0,0,0,0];
    app.trendDataTimeDruck  = (0);
    app.datasizeDruck       = 200;
    app.datasizeWaagen      = 400;

    %Pausen Daten und Totvolumen
    app.speed = 10 ;
    app.ramp_up = 0;
    app.pausenzeit = 0;
    app.dead_Volume = 0;
```

**Abbildung 4.19:** Auszug aus dem Programmcode der Funktion *startupFcn(app)*. Dargestellt sind einige Initialwert-Zuweisungen der globalen Variablen.

#### 4.2.4.3 Funktion zur Zustandsspeicherung und -ladung

Um gemäß dem Konzept eine einfache und effiziente Handhabung zu ermöglichen, verfügt das GUI über eine Zustandsspeicherung und -ladung. Dies bedeutet, dass Eingaben des Endbenutzers in Text- oder Zahlenfelder der Hardware-Initialisierung sowie der Filter- und Pumpeneinstellungen beim Beenden des Programmes gespeichert werden. Die letzte Kalibrierung der Feedpumpe wird ebenfalls gespeichert. Beim erneuten Start der Anwendung werden die gespeicherten Daten automatisch geladen und in die jeweiligen Text- oder Zahlenfelder eingetragen, ohne sie erneut angeben zu müssen.

Die zu speichernden Variablen werden dafür in der Funktion *saveState(app)* in der Form *state.VariableName* neu deklariert. Mit dem Befehl *save(path,'state')* werden alle *state*-Variablen in einem *.mat*-File unter dem angegebenen Pfad gespeichert. Diese Funktion wird beim Schließen der Anwendung (4.2.4.9 *UIFigureClose*-Funktion) aufgerufen. Das Laden der gespeicherten Daten erfolgt in der *Startup*-Funktion durch den Aufruf der Funktion *loadState(app)*. Hier werden die *state*-Variablen aus dem *.mat*-File mit Hilfe des Befehls *load(path,'state')* geladen und als globale Variablen oder als Werte der Text- oder Zahlenfelder definiert (Abbildung 4.20). Beide Funktionsaufrufe sind in einer *try/catch*-Anweisung eingerahmt. Beim ersten Start des Programmes ohne vorherige Speicherung oder durch keine Eingabe in den zu speichernden

Text- oder Zahlenfelder würden die einfachen Funktionsaufrufe zu Fehlermeldungen führen. Die *try/catch*-Anweisung fängt diese Fehler ab und ermöglicht so ein fehlerfreies Laden und Speichern der ausgefüllten Felder.

```
function loadState(app)
    load('C:\Users\abz810\Desktop\Fabian\MyAppDefaultValues.mat','state');

    %Laden der Com Ports
    app.comPortDruckmonitor = state.ComPortPressureMonitorEditField.Value;
    app.ComPortPressureMonitorEditField.Value = state.ComPortPressureMonitorEditField.Value;
    app.comPortFeedWaage = state.ComPortFeedScaleEditField.Value;
    app.ComPortFeedScaleEditField.Value = state.ComPortFeedScaleEditField.Value;
    app.comPortPermeatWaage = state.ComPortPermeatScaleEditField.Value;
    app.ComPortPermeatScaleEditField.Value = state.ComPortPermeatScaleEditField.Value;
    app.comPortPump = state.ComPortFeedPumpEditField.Value;
    app.ComPortFeedPumpEditField.Value = state.ComPortFeedPumpEditField.Value;
    app.comPort_diaPump = state.ComPortDiafiltrationPumpEditField.Value;
    app.ComPortDiafiltrationPumpEditField.Value = state.ComPortDiafiltrationPumpEditField.Value;
```

**Abbildung 4.20: Auszug aus dem Programmcode der Funktion *loadState(app)*.** Abgebildet sind das Laden und Speichern der *state*-Variablen.

#### 4.2.4.4 Initialisierung der COM-Ports

Die Bedienung der Anwendung startet im Reiter *Hardware Setup*. Hier ruft die *ButtonPushed-Callback* Aktion des Buttons “*Initialization of Com Ports*“ (Nr. 2 in Abbildung 4.10) die Funktion *ComPortInitialisierungen(app)* auf (Abbildung 4.22). Der Funktionsinhalt ist in einer *if/else*-Anweisung eingerahmt. Diese prüft zunächst mit Hilfe von *if*-Bedingungen, ob die in den Textfeldern eingegebenen COM-Ports verfügbare COM-Ports des Laptops sind. Dieses wird durch den Befehl *contains(app.comPortName,serialportlist) == 0* realisiert. Wird ein nicht verfügbarer COM-Port eingetragen, erscheint ein Dialogfenster mit einer darauf hinweisenden Fehlermeldung. Werden ausschließlich aktive COM-Ports eingetragen werden alle ansteuerbaren Geräte mit den Befehlen, wie in Kapitel 4.2.3 beschrieben, mit der Anwendung verbunden. Nach den einzelnen Verbindungsbefehlen wird als Test ein *writeline*-Befehl an die Pumpen und Waagen gesendet, der das geräteseitige Senden der Messdaten auslöst und somit den Empfangspuffer der Geräte füllt

(Waagenbefehlsstring: 'Esc P CR LF', Pumpenbefehlsstring: '1 RS CR', s. Abbildung 4.21). Die Messdaten des Druckmonitors werden ohne Senden eines Befehles automatisch vom Gerät gesendet und der Empfangspuffer gefüllt. Bei jedem Gerät wird mittels einer weiteren *if/else*-Anweisung überprüft, ob das Verbinden erfolgreich war und die Kommunikation möglich ist. Dafür wird mit der Bedingung `app.SerialObject.NumBytesAvailable > 0` getestet, ob der Empfangspuffer der Geräte tatsächlich gefüllt ist. Wenn die Bedingung erfüllt ist, und somit die Verbindung zu dem Gerät funktional ist, wird dies durch einen grünen Haken hinter den Textfeldern angezeigt und der Empfangspuffer mit dem Befehl `flush(app.SerialObject)` gelöscht. Des Weiteren wird an dieser Stelle jeweils für die Waagen und dem Druckmonitor eine *Callback*-Funktion definiert (Abbildung 4.21). Der Befehl `configureCallback(app.SerialObject,"terminator",callbackFcn)` bestimmt eine Funktion "callbackFcn", welche aufgerufen wird sobald sich Daten im Empfangspuffer des Gerätes befinden. Mit anderen Worten, sobald das Gerät Messdaten sendet. Die Implementierungen dieser *Callback*-Funktionen werden in den folgenden Unterkapiteln genauer erläutert. Ein Beschriftungsfeld, welches von "Wait" auf "Ready!" übergeht, informiert den Endbenutzer über das Ende des Initialisierungsprozesses.

```

%% Serielle Schittstelle Initialisieren Feed Waage
app.SeriellObjektFeedWaage = serialport(app.comPortFeedWaage,9600,"Timeout",10,"DataBits",7,"StopBits",1,"Parity","even");
configureTerminator(app.SeriellObjektFeedWaage,'CR/LF');
%Was senden
commandFeed= sprintf('%cP\r\n',27); % ESC P CR LF
writeline(app.SeriellObjektFeedWaage, commandFeed);
pause(0.2)
if app.SeriellObjektFeedWaage.NumBytesAvailable > 0
    flush(app.SeriellObjektFeedWaage); % Löscht den Empfangspuffer
    %%Callback zur automatischen Datenaufnahme der Feed Waage
    configureCallback(app.SeriellObjektFeedWaage,"terminator",@app.readSerialDataFeedWaage);
    app.Haken_gruen_WaageFeed_COMPort.Visible = 'on';
else
    errordlg('Please check if Feed Scale is powered on and restart the programm!','COM Port not online');
end

```

**Abbildung 4.21: Ausschnitt aus dem Programmcode der Funktion *ComPortInitialisierungen(app)*.** Abgebildet ist die serielle Verbindungsherstellung der Feedwaage, ihre Kommunikationsüberprüfung und die Deklaration ihrer *Callback*-Funktion.

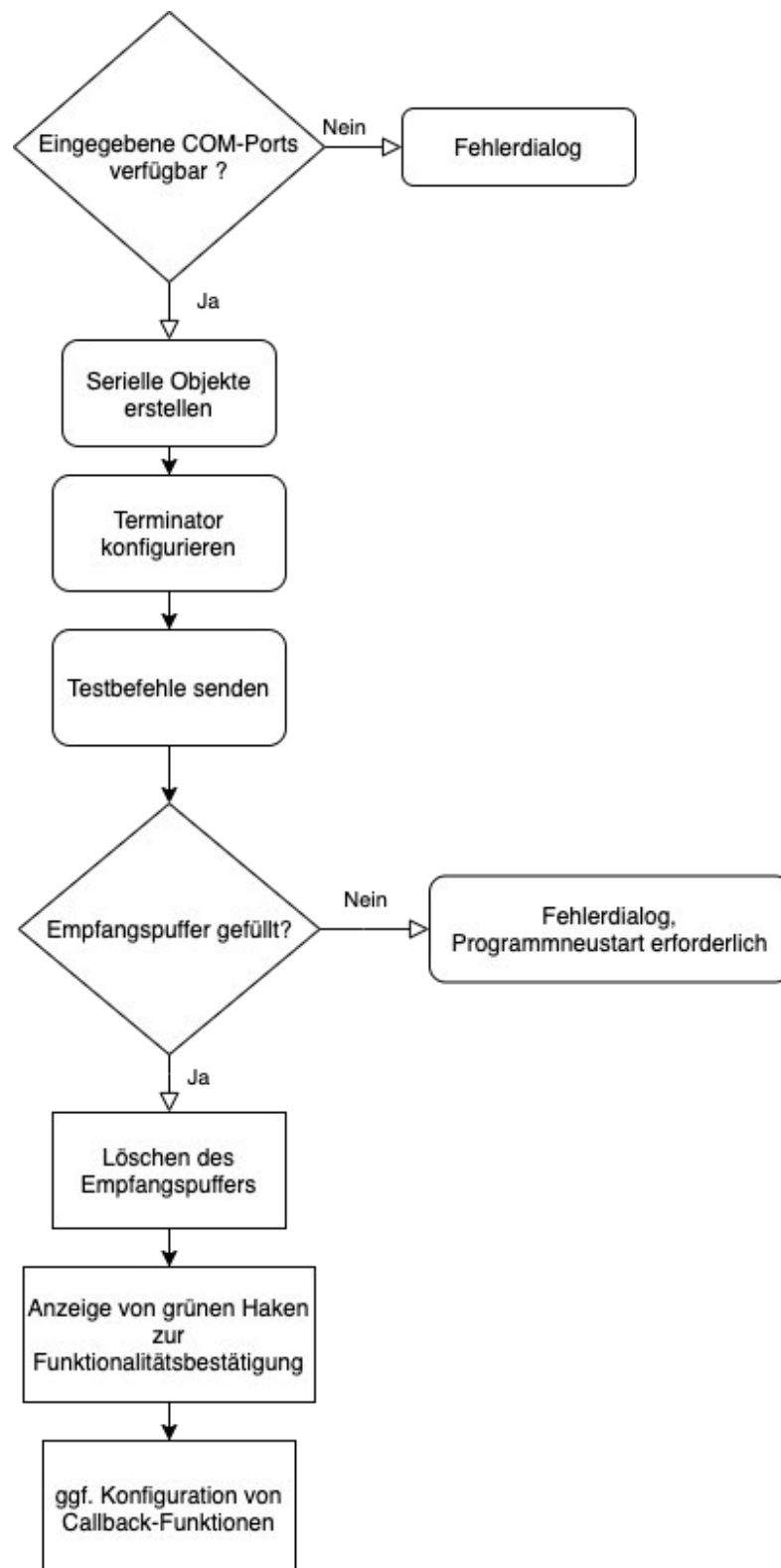


Abbildung 4.22: Flowchart der *ButtonPushed-Callback-Funktion ComPortInitialisierungen(app)*.

#### 4.2.4.4.1 **Callback-Funktionen der Waagen**

Die Waagen senden ihre Messwerte nur nach Aufforderung über die serielle Schnittstelle an den Laptop. Dafür muss der Befehlsstring 'ESC P CR LF' an die jeweilige Waage gesendet werden. Daraufhin antwortet die Waage und hinterlegt den Messwert in ihrem Empfangspuffer. Dies ist der Trigger der hier beschriebenen *Callback*-Funktionen, die daraufhin ausgeführt werden.

##### 4.2.4.4.1.1 **Feedwaage**

Die *Callback*-Funktion der Feedwaage heißt `readSerialDataFeedWaage(app,ser,~)` wobei das Input-Argument "ser" dem seriellen Objekt der Feedwaage entspricht. Als Erstes wird beim Aufruf der Funktion der gesendete Wert aus dem Empfangspuffer mit dem Befehl `readline(ser)` in einer lokalen Variable (`feedWaageData`) gespeichert. Da der gespeicherte Daten-String aus Vorzeichen, Zahlenwert und Einheit besteht wird der Befehl `SS = strsplit(feedWaageData," ")` verwendet um den gesamten String in drei Einzelstrings zu teilen. Das Vorzeichen sowie der Zahlenwert wird mit Hilfe des Befehls `str2double(SS(1)+SS(2))` in der globalen Hilfsvariable `app.GewichtFeedWaageHilfe` als Double-Datentyp gespeichert.

Nur wenn der Prozess beim Aufruf der Funktion bereits gestartet ist (`app.Process_status = 1`), werden die globalen Hauptdatenvariablen `app.GewichtFeedWaage` und `app.t_feedWaage` mit dem aktuellen Messwert bzw. dem dazugehörigen Prozesszeitpunkt gefüllt (Abbildung 4.23). Dafür wird zum einen das Totvolumen der Anlage (s. 4.2.4.6.3 Anlagen Vorbereitung) zu der Hilfsvariable `app.GewichtFeedWaageHilfe` addiert und im Falle eines internen Trierens das konstante Taragewicht (s. 4.2.4.6.3 Anlagen Vorbereitung) zusätzlich subtrahiert. Zum anderen wird der aktuelle Prozesszeitpunkt mit Hilfe der Matlab<sup>®</sup>-internen Stoppuhrbefehlen `tic/toc` gemessen. Von diesem Wert wird die Zeit möglicher Prozessunterbrechungen (4.2.4.7.4 Prozessunterbrechung) subtrahiert. Ob der Prozess bereits gestartet ist und das interne Trieren verwendet wurde, wird mit Hilfe von *if*-Bedingungen überprüft. Die Hauptdatenvariablen `app.GewichtFeedWaage` und `app.t_feedWaage` werden im gesamten Programmcode verwendet, um die aktuellen Messwerte der Feedwaage graphisch sowie als Text in der Überwachungstabelle und in der Prozessübersicht anzuzeigen.

```
%Daten werden erst in die Arrays eingelesen wenn der Process läuft
if app.Process_status == 1 && app.CheckBox_UseConcStartValue.Value == 1

    app.GewichtFeedWaage = (app.GewichtFeedWaageHilfe + app.dead_Volume) - app.AutoTarierGewicht;
    app.t_feedWaage = (toc(app.startZeitpunkt)/60)-app.pausenzeit;

elseif app.Process_status == 1

    app.GewichtFeedWaage = app.GewichtFeedWaageHilfe + app.dead_Volume;
    app.t_feedWaage = (toc(app.startZeitpunkt)/60)-app.pausenzeit;

end
```

**Abbildung 4.23:** Ausschnitt aus dem Programmcode der *Callback-Funktion readSerialDataFeedWaage(app,ser,~)*. Zu sehen ist das Einlesen der Feedwaagenmessdaten in die globalen Hauptdatenvariablen.

#### 4.2.4.4.1.2 Permeatwaage

In der *Callback-Funktion readSerialDataPermeatWaage(app,ser,~)* wird in gleicher Art und Weise wie bei der Feedwaage das aktuelle Gewicht auf der Permeatwaage in der globalen Hilfsvariable *app.GewichtPermeatWaageHilfe* gespeichert. Die globale Hauptdatenvariable des Prozesszeitpunktes *app.t\_permeatWaage* wird ebenso wie oben beschrieben ermittelt. Die globale Hauptdatenvariable *app.GewichtPermeatWaage* wird durch das Addieren der Hilfsvariable *app.GewichtPermeatWaageHilfe* und dem permeatseitigem Totvolumen der Anlage (s. 4.2.4.6.3 Anlagen Vorbereitung) berechnet und abgespeichert. Ebenso werden in dieser *Callback-Funktion* die externen Funktionen zur Berechnung des Flux-Wertes und des Konzentrationsfaktors bzw. im Falle einer Diafiltration des Diafiltrationsvolumens aufgerufen. Die Implementierungen dieser Funktionen werden in Kapitel 4.2.4.7.2.2 näher erläutert. Die Hauptdatenvariablen *app.GewichtPermeatWaage* und *app.t\_permeatWaage* werden im gesamten Programmcode verwendet, um die aktuellen Messwerte der Permeatwaage graphisch sowie als Text in der Überwachungstabelle und in der Prozessübersicht anzuzeigen.



#### 4.2.4.4.2 *Callback*-Funktion des Druckmonitor

Der Druckmonitor füllt alle 5 Sekunden automatisch seinen Empfangspuffer mit den aktuellen Messwerten der Drucksensoren (P1, P2, P3) und dem intern-ausgerechneten Transmembrandruck (TMP). Auch hier agiert das Füllen des Empfangspuffers als Trigger, weshalb alle 5 Sekunden die *Callback*-Funktion *readSerialDataDruck(app,ser,~)* ausgeführt wird (Abbildung 4.24). Da dieses auch bereits vor Prozessstart der Fall ist, wird der Inhalt der Funktion von einem *if*-Anweisungsblock umrahmt. Der Inhalt der Funktion wird nur ausgeführt, wenn die globale *app.DruckCallback\_Variable* auf 1 gesetzt ist. Dies geschieht in der *ButtonPushed-Callback* Funktion des Start Buttons (4.2.4.7.1 Prozess Start). Wie bei den Waagen wird nach Prozessstart mit dem Befehl *readline(ser)* der Empfangspuffer ausgelesen und die Messdaten mittels *str2double*-Befehl als Double-Datentyp zwischengespeichert. Diese werden für eine Einheitsumrechnung von bar in mbar mit 1000 multipliziert und in die globalen Hauptdatenvariablen *app.P1*, *app.P2*, *app.P3* und *app.TMP* gespeichert. Die globale Hauptdatenvariable des Prozesszeitpunktes *app.DruckT* wird in gleicherweise wie in den *Callback*-Funktionen der Waagen ermittelt. Als letztes wird die externe Funktion *trendgrafik\_druck()* aufgerufen. In ihr wird die Graphen-Anzeige der Druck-Messdaten im Reiter *Graphs* geregelt. In Kapitel “Prozessdatenanzeige“ wird die Implementierung dieser Funktion genauer erklärt.

```

if app.DruckCallback_Variable == 1

% Druckwert Einlese
data1 = readline(ser);

% Verarbeitung der Daten
SS = strsplit(data1, ' ');
S=string(SS);
P=str2double(S);

%Zeit (x)
app.DruckT = (toc(app.startZeitpunkt)/60)-app.pausenzeit;

%Daten (y)
app.P1 = 10^3*P(2);           %in mbar
app.P2 = 10^3*P(3);           %in mbar
app.P3 = 10^3*P(4);           %in mbar
app.TMP = 10^3*P(5);          %in mbar

```

**Abbildung 4.24:** Ausschnitt aus dem Programmcode der *Callback*-Funktion *readSerialDataDruck(app,ser,~)*. Zu sehen ist das Einlesen der Druckmessdaten in die globalen Hauptdatenvariablen.

#### 4.2.4.5 Kalibrierung der Feedpumpe

Bevor der Prozess innerhalb der Anwendung geplant wird, ist die Kalibrierung der Feedpumpe notwendig (Reiter 2: Calibration). Die Pumpe muss jeweils an einem ausgewählten Kalibrierpunkt die Testflüssigkeit 60 s lang transportieren und danach erneut stoppen. Diese Anweisung ist in der *ButtonPushed-Callback*-Funktion des Start- bzw. Stoppbuttons hinterlegt. Der mit dem *writeline*-Befehl an die Feedpumpe gesendete Befehlsstring '1 GO CR' startet bzw. '1 ST CR' stoppt die Pumpe. Die Pumpe startet dabei immer mit dem RPM- oder Pumpleistungswert, der in das dazugehörige Zahlenfeld (Nr. 3 in Abbildung 4.11) eingetragen wird. Dafür sorgt die *ValueChanged-Callback*-Funktion des Zahlenfeldes. In dieser Funktion wird zunächst der eingetragene Wert in die globale Variable *app.PumpRate\_prozent* gespeichert und mittels *sprintf('1SP%2.f',app.PumpRate\_prozent)* als Befehlsstring an die Pumpe gesendet. Nachdem das geförderte Volumen pro Minute eingetragen wurde, ruft die *ButtonPushed-Callback* Aktion des Buttons "Save Calibration Point" eine weitere Funktion auf. In dieser werden die RPM- oder Pumpleistungswerte (x-Werte) und die dazugehörigen ermittelten Flussraten (ml/min) (y-Werte) in zwei Vektoren mit der Nummer des jeweiligen Kalibrierpunktes (Nr. 1 in Abbildung 4.11) als Index gespeichert.

Die *ButtonPushed-Callback*-Funktion des Buttons "Get Calibration Curve" ist für die weitere Verarbeitung der Kalibrierdaten zuständig (Abbildung 4.25). Zunächst werden beide Vektoren mit dem Matlab®-Operator ' transponiert, um sie für die lineare Regression mit Hilfe des Backslash-Operators vorzubereiten. Anschließend wird aus dem Vektor der x-Werte (*app.PumpRate\_prozentForCali*) eine Koeffizienten-Matrix "M" erstellt. Der Vektor der y-Werte (*app.PumpRate\_mlpermin*) agiert als Ergebnis-Vektor. Dieses lineare Gleichungssystem kann nun mit dem Backslash-Operator durch den Befehl  $P = M \backslash app.PumpRate\_mlpermin$  gelöst werden. Der Lösungs-Vektor "P" enthält die Steigung der Regressionsgeraden und den y-Achsenabschnitt, welche als globale Variable *app.pumpCali\_steigung* respektive *app.pumpCali\_AchsenAbschnitt* gespeichert werden. Aus diesen wird das Funktions-Handle  $freg = @(x) app.pumpCali\_steigung * x + app.pumpCali\_AchsenAbschnitt$  definiert woraus die Werte der Regressionsgeraden ermittelt werden können. Die *ButtonPushed-Callback*-Funktion berechnet ebenso das Bestimmtheitsmaß der Regressionsgeraden. Dafür wird zunächst die Gesamtstreuung der gemessenen Flussraten (y-Werte) und die erklärte Streuung der durch die Regression ermittelten Flussraten errechnet.

Die Berechnung der Streuungswerte wird dabei durch zwei *for*-Schleifen realisiert wird. Das Bestimmtheitsmaß ergibt sich aus der Division der erklärten Streuung durch die Gesamtstreuung. Im letzten Schritt der Funktion werden die Rohdaten der Kalibrierung zusammen mit der Regressionsgerade, ihrer Funktionsgleichung und dem berechneten Bestimmtheitsmaß im Graphen *app.UIAxes\_PumpCalibration* abgebildet. Dafür wird der Matlab®-Befehl *plot()* verwendet.

Wenn der Endbenutzer bei erneutem Anwendungsstart anstelle einer neuen Kalibrierung die zuletzt gespeicherte Kalibriergerade laden möchte, wird dies durch die *ButtonPushed-Callback-Funktion LoadcalibrationcurveButtonPushed(app,event)* ermöglicht. Diese wird durch das Klicken des Buttons “*Load calibration curve*” (Nr. 9 in Abbildung 4.11) ausgeführt. In ihr findet erneut das Definieren des Funktions-*Handle freg*, das Plotten der Rohdaten sowie der Regressionsgeraden und das der Funktionsgleichung und dem Bestimmtheitsmaß der letzten Kalibrierung statt.

```
% Button pushed function: GetCalibrationCurveButton
function GetCalibrationCurveButtonPushed(app, event)
    app.PumpRate_prozentForCali = app.PumpRate_prozentForCali';
    app.PumpRate_mlpermin = app.PumpRate_mlpermin';

    %Backslash Operator
    M = [app.PumpRate_prozentForCali, ones(size(app.PumpRate_prozentForCali))];
    P = M\app.PumpRate_mlpermin;
    app.pumpCali_steigung = P(1);
    app.pumpCali_AchsenAbschnitt = P(2);

    %Regression
    freg=@(x)app.pumpCali_steigung * x + app.pumpCali_AchsenAbschnitt;
    T=0:0.5:app.PumpRate_prozentForCali(end);
    Z=freg(T);
```

**Abbildung 4.25: Auszug aus dem Programmcode der Funktion *GetCalibrationCurveButtonPushed(app, event)*.** Zu sehen ist das Transponieren der Daten-Vektoren, das Lösen des Gleichungssystems mittels Backslash-Operator und das Berechnen der Werte der Regressionsgeraden.

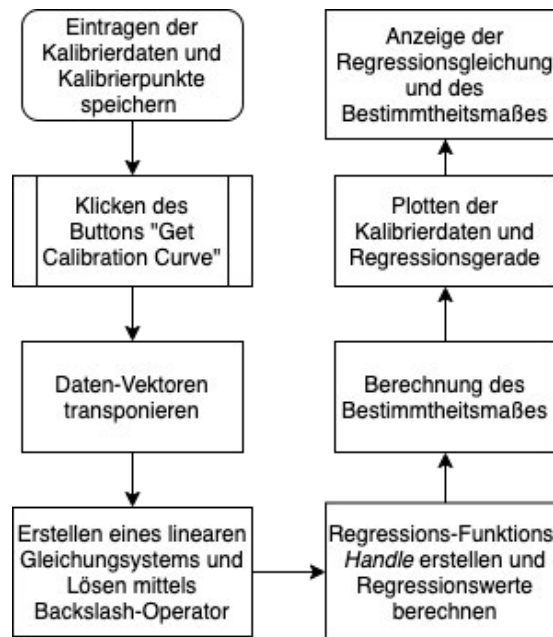


Abbildung 4.26: Struktureller Ablauf der Feedpumpenkalibrierung.

#### 4.2.4.6 Prozess Einrichtung

Dieses Kapitel beinhaltet die Erläuterungen aller wichtigen *Callback*-Funktionen des Reiters *Process Setup*. Auf *Callback*-Funktionen von Zahlen- oder Textfeldern, welche lediglich die Eingaben speichern, wird nicht genauer eingegangen.

##### 4.2.4.6.1 Prozess Planung

Der nächste Schritt ist die Planung des Filtrationsprozesses. Dies wird im Abschnitt *Create Trial Recipe* des Reiters *Process Setup* (Kapitel 4.2.2.3) umgesetzt. Die *SelectionChanged-Callback* Aktion der *FiltrationModeButtonGroup* (Nr. 11 in Abbildung 4.13) ruft eine Funktion auf, welche die korrekte Anzeige der Endpunkteinstellungen je nach ausgewähltem Filtrationsmodus realisiert. Dieses wird mit vier einfachen *if*-Anweisungen und der jeweiligen Komponenteneigenschaft *app.ComponentName.Visible = 'on'* bzw. *'off'* implementiert. Von den Softwarekomponenten der Endpunkteinstellungen besitzt nur das Dropdown-Menü *EndPointTypeDropDown\_1st\_Concentration* (Nr. 13) eine *ValueChanged-Callback* Aktion. Diese *Callback*-Funktion macht mit Hilfe einer *if/else*-Anweisung eine weitere

Checkbox “*Constant Flux reached*“ sichtbar, falls als Endpunkttyp “Flux-Wert“ ausgewählt wird. Diese Checkbox muss aktiviert werden, sobald sich ein stabiler Flux-Wert nach Einfahren der Anlage eingestellt hat. Dies verhindert ein sofortiges Beenden des Prozesses durch vorzeitiges Erreichen des Flux-Endpunktes beim Hochfahren der Pumpe.

Die *SelectionChanged-Callback*-Funktion der *FilterSetupButtonGroup* (Nr. 21) regelt mittels zwei *if*-Anweisungen und der *Visible*-Eigenschaft der Komponenten die Anzeige der Hohlfasereinstellungen oder der Kassetteneinstellungen. Das Zahlenfeld *FiberIDmmEditField* (Nr. 23 in Abbildung 4.13) besitzt eine *ValueChanged-Callback*-Funktion in der der eingegebene Innendurchmesser der Fasern von mm in cm umgerechnet und in der globalen Variable *app.fiberRadius\_cm* gespeichert wird.

Auch die *SelectionChanged-Callback* Aktion der *FeedPumpSetupButtonGroup* (Nr. 25) regelt die Anzeige der Zahlenfelder und der Beschriftungsfelder der Flussrate- bzw. Schergeschwindigkeitseinstellung. Durch eine Eingabe in das Zahlenfeld *Set\_FlowRate\_EditField* (Nr. 26) wird die Funktion *Set\_FlowRate\_EditFieldValueChanged(app, event)* ausgeführt (Abbildung 4.27). Diese ermittelt zunächst aus der eingegebenen Flussrate des Endbenutzers mit Hilfe der Regressionsgleichung der Feedpumpenkalibrierung (4.2.4.5 Kalibrierung der Feedpumpe) den entsprechenden RPM- oder Pumpleistungswert. Daraufhin wird die theoretische Schergeschwindigkeit in den Fasern berechnet. Dafür wird die eingegebene Flussrate in ml/min durch 60 dividiert, dieser Wert ebenso durch die Faseranzahl (Nr. 22, *FiberCountEditField*) dividiert um die Einheit cm<sup>3</sup>/s pro Faser zu erzeugen. Nun wird mit Hilfe der folgenden Formel die Schergeschwindigkeit errechnet und in der globalen Variable *app.shearRate* gespeichert.

$$\dot{\gamma} = \frac{4 * \dot{V}_F}{\pi * r^3} \quad (4.1)$$

Dabei stellt  $\dot{\gamma}$  die Schergeschwindigkeit in s<sup>-1</sup>,  $\dot{V}_F$  die Feed-Flussrate pro Hohlfaser in cm<sup>3</sup>/s und r den Radius der Hohlfasern in cm dar. Mit dem Befehl *app.Set\_Flowrate\_ShearLabel.Text = num2str(app.shearRate, '%.1f')* wird die

ermittelte Schergeschwindigkeit im Beschriftungsfeld der Schergeschwindigkeitseinstellung abgebildet.

```

% Value changed function: Set_FlowRate_EditField
function Set_FlowRate_EditFieldValueChanged(app, event)

%Umrechnung mit Kalibrierung in Pumprate [%/RPM]
app.gesetzteFlowrate_in_mlpermin = app.Set_FlowRate_EditField.Value;
app.berechneteFlowrate_vonOperator_Vorgabe_Prozent_RPM = (app.gesetzteFlowrate_in_mlpermin-app.pumpCali_AchsenAbschnitt)/app.pumpCali_steigung;

%Shear Rate Berechnung (!!BIS JETZT NUR MIT FEED FLOWRATE BERECHNET!!)
flowrate_cm3pers_perFiber = (app.gesetzteFlowrate_in_mlpermin / 60) / app.FiberCountEditField.Value;
app.shearRate = (4*flowrate_cm3pers_perFiber)/(pi*app.fiberRadius_cm^3);

%Shear Rate Label anpassen]
app.Set_FlowRate_ShearLabel.Text = num2str(app.shearRate, '%.1f') ;

```

**Abbildung 4.27: Programmcode der Funktion `Set_FlowRate_EditFieldValueChanged(app, event)`.** Abgebildet ist die Berechnung und anschließende Anzeige der Schergeschwindigkeit nach eingegebener Flussrate.

Entscheidet sich der Endbenutzer nicht für das Eintragen der Flussrate, sondern für die Schergeschwindigkeit, wird die Funktion `Set_Shear_EditField_2ValueChanged(app, event)` aufgerufen (Abbildung 4.28). In dieser wird mit Hilfe der gleichen Formeln (Formel 4.1) aus der Funktion `Set_FlowRate_EditFieldValueChanged(app, event)` die eingegebene Schergeschwindigkeit in die Flussrate in  $\text{cm}^3/\text{s}$  pro Faser, die Flussrate dann in  $\text{ml}/\text{min}$  und schließlich in den entsprechenden RPM- oder Pumpleistungswert umgerechnet (Abbildung 4.29). Ebenso wird die berechnete Flussrate in  $\text{ml}/\text{min}$  im Beschriftungsfeld der Flussrateneinstellung dargestellt.

```

% Value changed function: Set_Shear_EditField_2
function Set_Shear_EditField_2ValueChanged(app, event)
    app.shearRate = app.Set_Shear_EditField_2.Value;

    %Hier die Berechnung von Shear Rate auf Flowrate
    flowrate_cm3pers_perFiber = ((pi*app.fiberRadius_cm^3)*app.shearRate)/4;
    app.gesetzteFlowrate_in_mlpermin = flowrate_cm3pers_perFiber * app.FiberCountEditField.Value * 60 ;
    app.berechneteFlowrate_vonOperator_Vorgabe_Prozent_RPM = (app.gesetzteFlowrate_in_mlpermin-app.pumpCali_AchsenAbschnitt)/app.pumpCali_steigung;

    %Flowrate Label Anzeige
    app.Set_Shear_FlowrateLabel.Text = num2str(app.gesetzteFlowrate_in_mlpermin, '%2.f');
end

```

**Abbildung 4.28: Programmcode der Funktion `Set_Shear_EditField_2ValueChanged(app, event)`.** Dargestellt ist die Berechnung und Anzeige der Flussrate in  $\text{ml}/\text{min}$  sowie des RPM- oder Pumpleistungswertes.

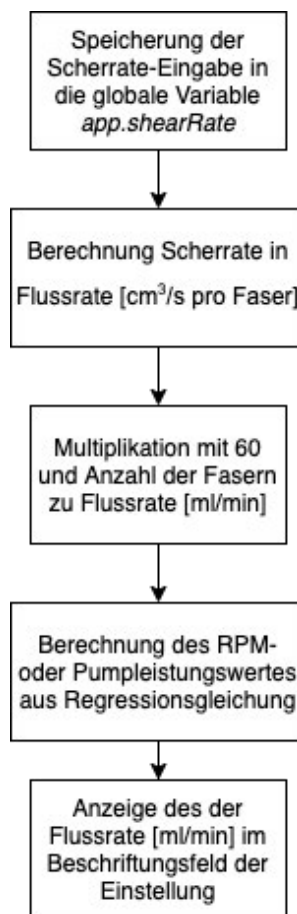


Abbildung 4.29: Flowchart der *ValueChanged-Callback-Funktion Set\_Shear\_EditField\_2ValueChanged(app, event)*.

Die *ButtonPushed-Callback-Funktion* des Buttons “*Change Flowrate at process*” (Nr. 27 in Abbildung 4.13) sendet einen Befehlsstring mit dem, aus den eben beschriebenen Funktionen, berechneten RPM- oder Pumpleistungswert an die Feedpumpe. Diese Implementierung ermöglicht die Anpassung der Flussrate während eines Prozesses. Durch eine Auswähländerung der *PumpDirectionButtonGroup* (Nr. 29) wird eine Funktion mit einer *if/else*-Anweisung aufgerufen. Abhängig von der Buttonauswahl, wird der Befehlsstring ‘1 RR CR’ für eine Pumpen-Drehrichtung im Uhrzeigersinn an die Pumpe gesendet bzw. ‘1 RL CR’ für eine Pumpen-Drehrichtung entgegen dem Uhrzeigersinn. Die Funktion, welche die *SelectionChanged-Callback* Aktion der *DiafiltrationPumpControllerSetupButtonGroup* (Nr. 30) ausführt, regelt wie bereits vorher erklärt das Anzeigen bzw. Ausblenden der Reglereinstellungen.



#### 4.2.4.6.2 Automatische Prozessdatenspeicherung

Um die automatische Speicherung der Prozessdaten (Waagen Signale, Druckdaten, Flux-Werte, Flussraten, Konzentrationsfaktoren oder Diafiltrationsvolumina) sowie der Parameter-Datei zu aktivieren, muss die *ButtonPushed-Callback* Aktion des Buttons "Search path" (Nr. 38 in Abbildung 4.13) ausgelöst werden. Die daraufhin aufgerufene Funktion *SearchpathButtonPushed2(app, event)* (Abbildung 4.31) öffnet durch den Befehl `[~,pfadAutoSave] = uiputfile` ein Dialogfenster für die Pfadauswahl und speichert den ausgewählten Pfad in der lokalen Variable *pfadAutoSave*. Das Datum sowie die Uhrzeit des Prozessstartes wird mit Hilfe der Matlab<sup>®</sup>-Anweisung *datestr(now)* ermittelt und in der Variable *datum* gespeichert. Da der Uhrzeit-String in den Namen der Dateien zur Identifikation vorkommen soll, müssen die Doppelpunkte im String mit Bindestrichen ausgetauscht werden. Dies wird mit dem Befehl `dp=strfind(datum, ':')` gefolgt von `datum(dp)='-'` realisiert. Im nächsten Schritt wird für jede Datei der Pfad, der Name der Prozessdaten, das Datum und die Uhrzeit sowie die Endung '.txt' zu einem String (Namen-String) kombiniert (Abbildung 4.30). Die dabei entstandenen Strings werden als globale Variablen gespeichert (z. B. `app.dateiname_autoSave_FeedGewicht`).

Im Anschluss beginnt das Erstellen der Dateien. Dabei wird eine externe eigens erstellte Funktion *writeDataFile(~,msg,filename)* verwendet. Diese Funktion öffnet eine Datei mit dem Namen *filename* und schreibt in diese mit Hilfe des Befehls `fprintf("msg\n")` den String *msg* in die nächste Zeile. Anschließend wird die Datei wieder geschlossen. Mit Hilfe dieser externen Funktion wird für alle Prozessdaten zunächst jeweils eine Datei geöffnet und mit einer Kopfzeile bestehend aus Prozessname, Operatorname und dem Prozess-Kommentar gefüllt. Durch einen erneuten Aufruf der Funktion werden die jeweiligen Tabellenüberschriften (z. B. Time [min], Flux [l/h\*m<sup>2</sup>]) in die nächste Zeile der Dateien geschrieben. Die gemessenen oder berechneten Prozessdaten können nun während des laufenden Prozesses Zeile für Zeile in die dazugehörigen Dateien geschrieben und somit dokumentiert werden (4.2.4.7.2.4 Prozessdatenspeicherung). Die Parameter-Datei wird dahingegen bereits zu diesem Zeitpunkt vollständig erstellt. Auch hierfür wird die externe Funktion *writeDataFile()* wie eben beschrieben verwendet. Abschließend wird der Status der Automatischen Speicherung durch die Zuweisung der globalen Variable `app.autoSave_status = 1` auf aktiv gesetzt.

```

% Button pushed function: SearchpathButton
function SearchpathButtonPushed2(app, event)
    [~,pfadAutoSave] = uiputfile;

    %Pfad und Dateiname
    datum = datestr(now);
    dp=strfind(datum, '-:');
    datum(dp)='-:~';

    %Namen der Files
    app.dateiname_autoSave_FeedGewicht = [pfadAutoSave 'Feed Weight Data ' datum '.txt'];
    app.dateiname_autoSave_PermeateGewicht = [pfadAutoSave 'Permeate Weight Data ' datum '.txt'];
    app.dateiname_autoSave_Druck = [pfadAutoSave 'Pressure Data ' datum '.txt'];
    app.dateiname_autoSave_Flux = [pfadAutoSave 'Flux Data ' datum '.txt'];
    app.dateiname_autoSave_Flowrate = [pfadAutoSave 'Flowrate Data ' datum '.txt'];
    app.dateiname_autoSave_Faktor = [pfadAutoSave 'CF_DV Data ' datum '.txt'];
    dateiname_Parameterfile = [pfadAutoSave 'Parameter File ' datum '.txt'];

    %Parameter File schreiben
    msg=sprintf('Parameter File for Process: %s',app.process_Name);
    writeDataFile(app,msg,dateiname_Parameterfile);
    writeDataFile(app, "", dateiname_Parameterfile);
    writeDataFile(app,sprintf("Process Data"),dateiname_Parameterfile);
    writeDataFile(app,sprintf("Feed Start Volume (Concentration)
    : %3.2f ml",app.EditField_StartVolumen_1stConcentration.Value),dateiname_Parameterfile);
    writeDataFile(app,sprintf("Feed Start Volume (Diafiltration)
    : %3.2f ml/min",app.EditField_StartVolumen_1stDiafiltration.Value),dateiname_Parameterfile);
    writeDataFile(app,sprintf("Flowrate
    : %3.2f l/s",app.shearRate),dateiname_Parameterfile);
    writeDataFile(app,sprintf("ShearRate
    : %3.2f mm",app.Tubing_Size_FeedPump_EditField.Value),dateiname_Parameterfile);
    writeDataFile(app,sprintf("Diafiltration Pump Tubing ID
    : %3.1f",app.FiberCount_EditField.Value),dateiname_Parameterfile);
    writeDataFile(app,sprintf("Hollow Fiber Data"),dateiname_Parameterfile);
    writeDataFile(app,sprintf("Fiber Count
    : %3.1f mm",app.FiberIDmm_EditField.Value),dateiname_Parameterfile);
    writeDataFile(app,sprintf("Surface Area
    : %3.1f cm²",app.SurfaceAreaacm_HollowFiber_EditField.Value),dateiname_Parameterfile);

    %Feed Gewicht Daten
    msg=sprintf('%6s %15s %15s %15s %15s\n', 'Process Name:', string(app.process_Name), 'Operator:', string(app.operator_Name), 'Comment:', string(app.process_Comment));
    writeDataFile(app,msg,app.dateiname_autoSave_FeedGewicht);

    msg=sprintf('%6s %15s\n', 'Time [min]', 'Weight Feed Reservoir [g]');
    writeDataFile(app,msg,app.dateiname_autoSave_FeedGewicht);
  
```

**Abbildung 4.30: Ausschnitt aus dem Programmcode der Funktion SearchpathButtonPushed2(app, event).** Abgebildet ist das Öffnen des Pfaddialoges, die Erstellung der Dateinamen und das Füllen der Dateien mit Hilfe der Funktion writeDataFile.

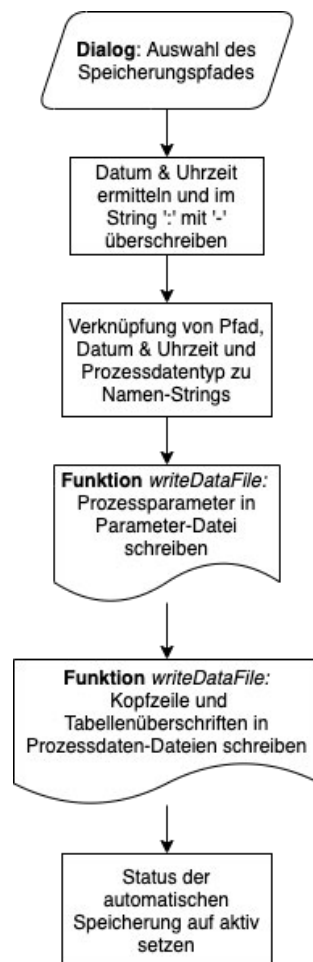


Abbildung 4.31: Flowchart der Funktion *SearchpathButtonPushed2(app, event)*.

#### 4.2.4.6.3 Anlagen Vorbereitung

Bevor der Prozess gestartet werden kann, muss die Anlage vorbereitet werden. Dazu zählt zum einen das interne Trieren der Feedwaage, wenn dies nicht bereits im Vorhinein manuell getan wurde. Zum anderen muss ein *Fill Up* der Anlage folgen, wobei die Anlage gefüllt und so das Totvolumen bestimmt wird.

Die Anzeige des Menüs für das interne Trieren der Feedwaage wird über die *ValueChanged-Callback*-Funktion der dazugehörigen Checkbox (Nr. 4 in Abbildung 4.13) mit Hilfe von einer *if*-Anweisung geregelt. Entscheidet sich der Endbenutzer für das interne Trieren wird durch Klicken des Buttons "Auto Tare" die Funktion *AutoTareButtonPushed(app, event)* aufgerufen. In ihr wird als erstes der Befehlsstring

'ESC P CR LF' an die Feedwaage gesendet (Abbildung 4.32). Durch die automatisch ausgeführte *Callback*-Funktion der Feedwaage (4.2.4.4.1 *Callback*-Funktionen der Waagen) kann die globale Hilfsvariable *app.GewichtFeedWaageHilfe* verwendet werden, um das aktuelle Gesamtgewicht auf der Feedwaage zu ermitteln. Von diesem Gesamtgewicht wird das in das Zahlenfeld (Nr. 5 in Abbildung 4.13) eingegebene Startvolumen des Feedbehälters subtrahiert, um das konstante Taragewicht zu erhalten und in der globalen Variable *app.AutoTaraGewicht* zu speichern. Auch hier wird der Autotarier-Status durch den Befehl *app.Tarier\_status = 1* auf aktiv gesetzt. Zusätzlich wird das Taragewicht in einem Beschriftungsfeld unter dem Button angezeigt.

Das Zahlenfeld im Abschnitt "System Fill Up" besitzt die *ValueChanged-Callback*-Funktion *PumpRateRPMEditFieldValueChanged2(app, event)*. In ihr wird der eingegebene *Fill Up*-RPM-Wert in den Befehlsstring zum Ändern der Pumpleistung eingebaut und an die Feedpumpe gesendet. Nun kann der Button "Start Fill Up" gedrückt werden, wodurch die *Callback*-Funktion *StartFillUpButtonPushed2(app, event)* ausgeführt wird (Abbildung 4.33). In dieser wird als erstes der Befehlsstring 'ESC P CR LF' an die Feedwaage gesendet. Der Befehl *app.FillUpStartVolumen = app.GewichtFeedWaageHilfe* speichert das Gewicht auf der Feedwaage vor dem *Fill Up*. Der daraufhin an die Feedpumpe gesendete Befehlsstring '1 GO CR' startet nun die Pumpe. Nachdem die retentatseitige Anlage vollständig gefüllt ist, löst das Klicken des Buttons "Stop Fill Up" die *Callback*-Funktion *StopFillUpButtonPushed2(app, event)* aus (Abbildung 4.33). Hier wird erneut der notwendige Befehlsstring an die Waage gesendet und das aktuelle Gewicht auf der Feedwaage durch Auslesen der Hilfsvariable in der Variable *app.FillUpStopVolumen* gespeichert. Anschließend stoppt der gesendete Befehlsstring '1 ST CR' die Feedpumpe. Die Differenz der beiden gespeicherten Gewichts-Variablen ergeben das retentatseitige Totvolumen der Anlage, welches ebenso in der globalen Variable *app.dead\_Volume* gespeichert wird. Das Totvolumen wird in dem dazugehörigen Beschriftungsfeld durch den Befehl *app.Label\_TotVolumen.Text = string(app.dead\_Volume)* angezeigt.

```

function AutoTareButtonPushed(app, event)
    %Feed Waage Befehl senden
    commandFeed= sprintf('%cP\r\n',27); % ESC P CR LF
    writeline(app.SeriellObjektFeedWaage, commandFeed);
    pause(0.5)

    %Offset berechnen
    app.AutoTariertGewicht = app.GewichtFeedWaageHilfe - app.EditField_StartVolumen_forAutoTaring.Value ;

    %Label schreiben
    app.Label_TareValue.Text = sprintf('%.1f',app.AutoTariertGewicht);

    %Tariert Status AN setzen
    app.Tariert_status = 1 ;
end

```

**Abbildung 4.32: Programmcode der Funktion *AutoTareButtonPushed(app, event)*.** Zu sehen ist das Senden des Waagenbefehlsstrings, die Berechnung und Anzeige des Taragewichtes und die Aktivierung des Tariertstatus.

```

function StartFillUpButtonPushed2(app, event)
    %Feed Waage Befehl senden
    commandFeed= sprintf('%cP\r\n',27); % ESC P CR LF
    writeline(app.SeriellObjektFeedWaage, commandFeed);
    pause(0.2)

    %Pumpe Start Befehl
    commandPumpe= sprintf('1G0');
    writeline(app.SeriellObjektPumpe, commandPumpe);

    %Gewicht vor Start merken
    app.FillUpStartVolumen = app.GewichtFeedWaageHilfe;
end

% Button pushed function: StopFillUpButton
function StopFillUpButtonPushed2(app, event)
    %Feed Waage Befehl senden
    commandFeed= sprintf('%cP\r\n',27); % ESC P CR LF
    writeline(app.SeriellObjektFeedWaage, commandFeed);
    pause(0.2)

    %Stop Befehl für Pumpe
    commandPumpe= sprintf('1ST');
    writeline(app.SeriellObjektPumpe, commandPumpe);

    %Gewicht nach Fill Up merken und Gesamtes Totvolumen bestimmen
    app.FillUpStopVolumen = app.GewichtFeedWaageHilfe;

    app.dead_Volumen = app.FillUpStartVolumen - app.FillUpStopVolumen;

    app.Label_TotVolumen.Text = string(app.dead_Volumen);
end

```

**Abbildung 4.33: Callback-Funktionen der Totvolumen-Berechnung.** Abgebildet sind die beiden Funktionen *StartFillUpButtonPushed2(app, event)* und *StopFillUpButtonPushed2(app, event)* und der dazugehörige Programmcode.

#### 4.2.4.7 Prozess Ablauf

Nachdem die ansteuerbaren Geräte mit der Software verbunden, die Feedpumpe kalibriert und der Prozess geplant und vorbereitet wurde, kann der Filtrationsprozess gestartet werden. In diesem Kapitel wird der Programmcode, welcher während eines laufenden Prozesses ausgeführt wird, beschrieben und erklärt.

##### 4.2.4.7.1 Prozess Start

Im Reiter *Process Overview* kann durch Klicken des Buttons "START" (Nr. 1 in Abbildung 4.14) die *Callback*-Funktion *STARTButtonPushed(app, event)* ausgelöst und damit der Filtrationsprozess gestartet werden (Abbildung 4.35). Dem Konzept zu folge, soll die Anwendung Robustheit gegenüber fehlerhaften oder vergessenen Eingaben ermöglichen. Dies gewährleisten 4 verschachtelte *if/else*-Blöcke (Abbildung 4.34). Die erste *if*-Bedingung prüft, ob das Totvolumen der Anlage bestimmt und gespeichert wurde. Ist dies der Fall, prüft die nächste *if*-Bedingung, ob alle Hohlfasereinstellungen ausgefüllt und gespeichert worden sind. Des Weiteren wird die Eingabe einer Flussrate oder einer Schergeschwindigkeit des Prozesses geprüft. Die letzte *if*-Bedingung fragt ab, ob der Status der Automatischen Speicherung aktiviert wurde. In dem Fall, dass eine der Bedingungen nicht erfüllt ist, wird der Endbenutzer durch einen spezifischen Fehlerdialog auf den jeweiligen Fehler aufmerksam gemacht.

Sind alle Bedingungen erfüllt, wird der durch die *if/else*-Blöcke eingerahmte Programmcode ausgeführt. Dem Endbenutzer wird dieses durch die Nachricht "*Process started successfully!*" kenntlich gemacht. Zunächst wird mit Hilfe der Matlab®-Funktion *tic* ein Zeitstempel in der globalen Variable *app.startZeitpunkt* gespeichert. Der Prozessstatus sowie der Druck-*Callback*-Status wird daraufhin durch den Befehl *app.Process\_status = 1* bzw. *app.DruckCallback\_Variable = 1* aktiviert. Wurde vom Endbenutzer ein Prozess geplant in dem eine Konzentrierung vorkommt (durch *if*-Bedingung geprüft), wird ebenso der Konzentrierungsstatus durch den Befehl *app.conc\_status = 0* aktiviert. Die an die Feedpumpe gesendeten Befehlsstrings '1 SP9 CR' und '1 GO CR' lassen die Pumpe mit 9 RPM starten. Nun wird der Haupttimer des Programmes durch die folgende Befehlszeile definiert: *app.t = timer("ExecutionMode","fixedRate","Period",3,"BusyMode","drop","TimerFcn",@app.startDatenAufnahmeundAnzeige)*.

Ebenso wird der Regelungstimer für die Füllstandsregelung einer Diafiltration definiert: `app.dia_timer = timer ("ExecutionMode","fixedRate","Period",1,"Busy-Mode","drop","TimerFcn", @app.diafiltrations_Regler_function)`. Der Haupttimer wird danach durch den Befehl `start(app.t)` aktiviert (4.2.4.7.2 Haupttimer *Callback*-Funktion).

Als Nächstes wird mit Hilfe einer weiteren *if*-Anweisung überprüft, ob der Endbenutzer eine einfache Diafiltration geplant hat. Ist dies der Fall, werden die folgenden Anweisungen zusätzlich ausgeführt. Zunächst wird der Diafiltrationsstatus aktiviert (`app.Dia_status = 0`). Daraufhin wird das eingetragene Startvolumen der Diafiltration (`app.Dia_startVolumen`) als Sollwert der Füllstandsregelung in der globalen Variable `app.gewichtSollwert` gespeichert. Wurde als Endpunkttyp der Diafiltration ein bestimmtes Diafiltrationsvolumen angegeben, wird das mit Diafiltrationspuffer auszutauschende Volumen berechnet und in der globalen Variable `app.Dia_volume_to_pump` gespeichert. Dies wird durch Multiplikation des Startvolumens der Diafiltration mit dem Diafiltrationsvolumen realisiert. Die Diafiltrationspumpe wird durch Senden der gleichen Befehlsstrings wie bei der Feedpumpe gestartet. Dabei wird abweichend zur Feedpumpe kein fester RPM-Wert verwendet, sondern der unter den Reglereinstellungen eingegebene RPM-Startwert der Diafiltrationspumpe. Schließlich wird der Regelungstimer durch den Befehl `start(app.dia_timer)` gestartet (4.2.4.7.3 Regelungstimer *Callback*-Funktion).

```

% Button pushed function: STARTButton
function STARTButtonPushed(app, event)
    if app.dead_Volume ~= 0
        if app.FiberCountEditField.Value ~= 0 && app.FiberIDmmEditField.Value ~= 0 && app.SurfaceAreaacm_HollowFiberEditField.Value ~= 0
            if app.Set_Shear_FlowRateLabel.Text ~= "Label" || app.Set_FlowRate_EditField.Value ~= 0
                if app.autoSave_status == 1
                    msgbox('Process started successfully!');
                end
            end
        end
    end
    app.startZeitpunkt = tic ;

    %Prozess Status AN schalten
    app.Process_status = 1;

    %Druck Callback Variable auf Start
    app.DruckCallback_Variable = 1;

    %Konzentrierungs Variable auf AN wenn C / CD / CDC ausgewählt ist
    if app.ConcentrationButton.Value == 1 || app.CDButton.Value == 1 || app.CDCButton.Value == 1
        app.conc_status = 0;
    end

    %Start der Pumpe mit niedrigem Wert
    commandPumpe= sprintf('1SP9');
    writeline(app.SeriellesObjektPumpe, commandPumpe);
    pause(0.01)
    commandPumpe= sprintf('1G0');
    writeline(app.SeriellesObjektPumpe, commandPumpe);
  
```

**Abbildung 4.34: Ausschnitt aus dem Programmcode der Funktion `STARTButtonPushed(app, event)`.** Dargestellt sind die verschachtelten `if`-Bedingungen, die Speicherung des Startzeitpunktes, die Aktivierung einzelner Status-Variablen sowie das Starten der Feedpumpe.



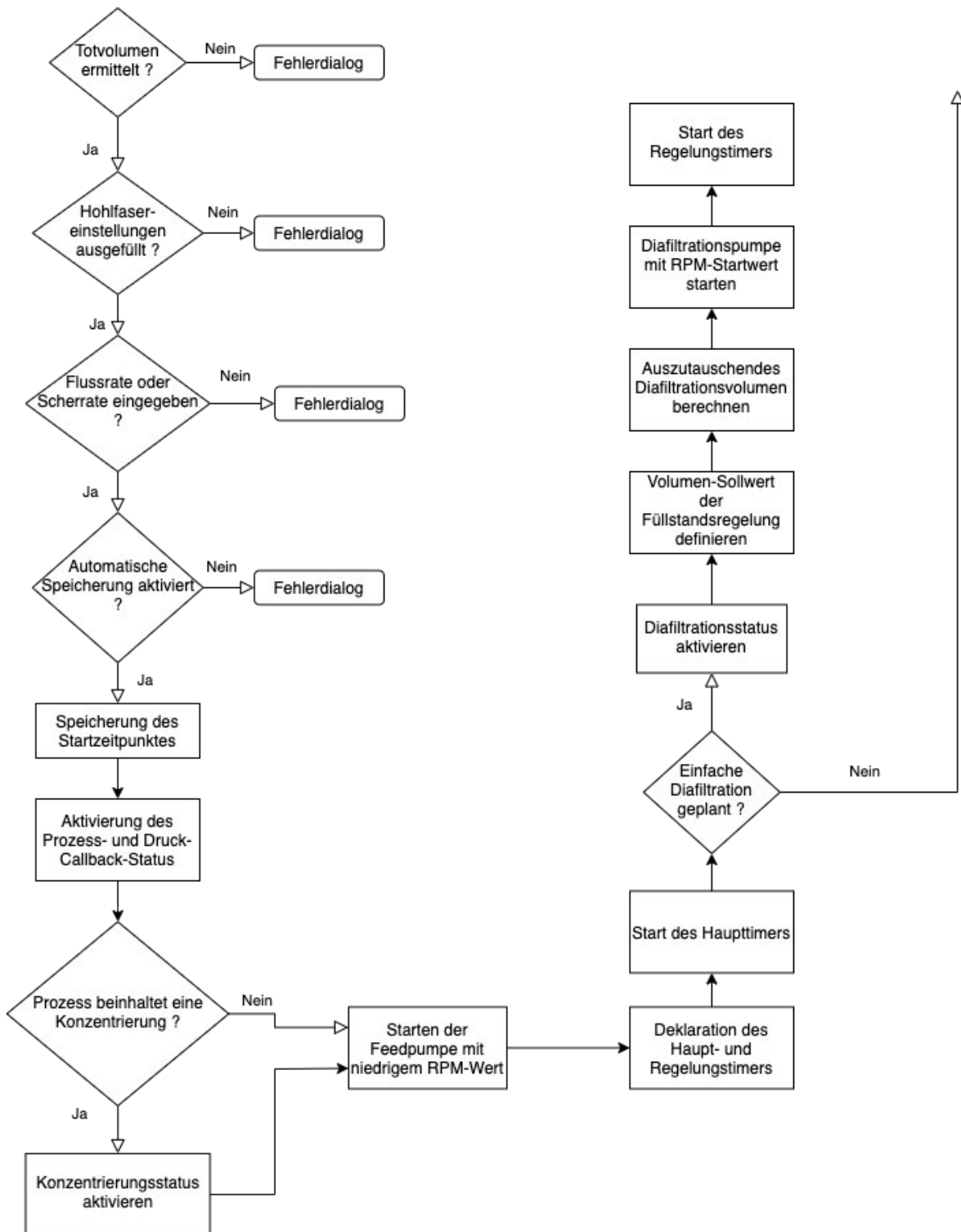


Abbildung 4.35: Flowchart der Funktion *STARTButtonPushed(app, event)*.

#### 4.2.4.7.2 Haupttimer *Callback*-Funktion

Der Haupttimer *app.t* wird bei jedem Filtrationsprozess gestartet, unabhängig davon welcher Filtrationsmodus ausgewählt wird. Nach Filtrationsstart wird dadurch seine *Callback*-Funktion *startDatenAufnahmeundAnzeige(app,~,~)* alle 3 Sekunden erneut aufgerufen. Die folgenden Unterkapitel erläutern den dabei ausgeführten Programmcode.

##### 4.2.4.7.2.1 Anlaufen der Feedpumpe

Damit die Zellen einer Suspension nicht durch ruckartiges Starten der Feedpumpe in die Poren der Filtermembran gedrückt werden, wird die Pumpleistung der Feedpumpe stufenweise erhöht. Dies wird durch die globale Hilfsvariable *app.speed* implementiert. Ihr wurde zunächst in der *Startup*-Funktion (4.2.4.2 *Startup*-Funktion) ein niedriger RPM-Wert zugewiesen.

In der Haupttimer *Callback*-Funktion wird der Wert dieser Variable in den Befehlsstring zum Ändern der Pumpleistung eingebaut und an die Feedpumpe gesendet. Die darauffolgende Codezeile erhöht den Wert der Hilfsvariable um 5 RPM (Abbildung 4.36). Durch den alle 3 Sekunden wiederholten Aufruf der Haupttimer *Callback*-Funktion entsteht dadurch ein langsames Steigern der Pumpleistung. Dieser Programmcode ist in einer *if/else*-Anweisung eingerahmt und wird nur ausgeführt, wenn die folgende *if*-Bedingung erfüllt ist. Der Wert der Hilfsvariable muss kleiner sein als der vom Endbenutzer vorgegebene und in RPM umgerechnete Wert der Flussraten- bzw. Schergeschwindigkeitseinstellung. Ist das Hochfahren der Pumpleistung so weit fortgeschritten, dass dies nicht mehr der Fall ist, wird die Pumpleistung der Feedpumpe auf den eingegebenen Wert des Endbenutzers eingestellt. Dies wird durch das Senden des Befehlsstrings *sprintf('1SP%2.2f',round(app.berechneFlowrate\_vonOperator\_Vorgabe\_Prozent\_RPM))* an die Feedpumpe realisiert. Ebenso wird in diesem Fall der *Ramp-Up*-Status durch den Befehl *app.ramp\_up = 1* aktiviert. Dies ist notwendig, denn der komplette *if/else*-Block inklusive *Ramp-Up*-Code wird nur ausgeführt solange dieser Status noch deaktiviert (*app.ramp\_up = 0*) ist.

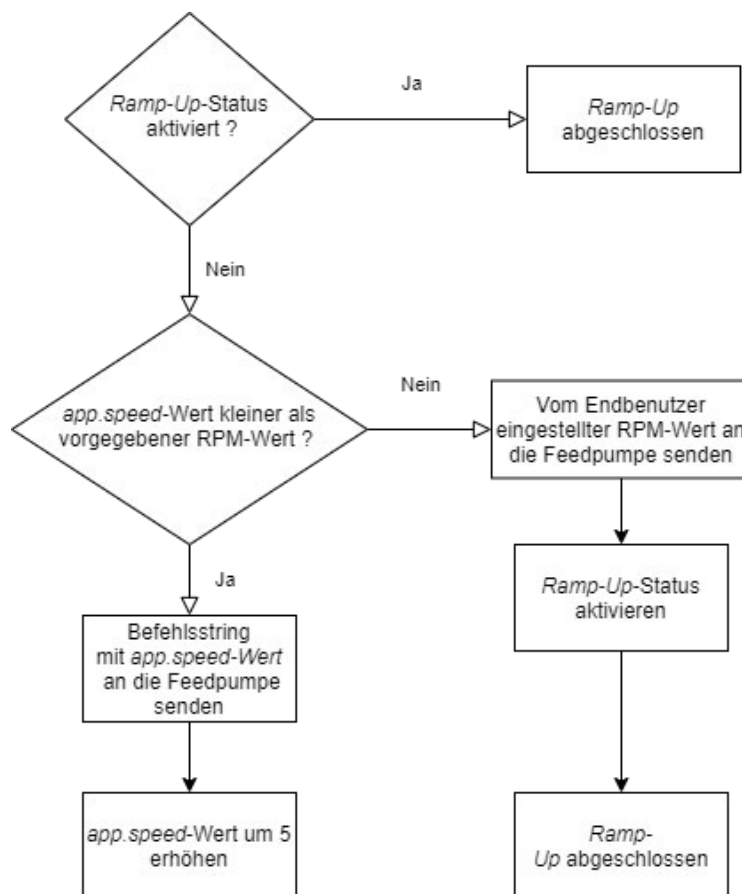
```

commandPumpe= sprintf('1SP%2.f',app.speed);
writeline(app.SeriellesObjektPumpe, commandPumpe);
pause(0.2)

app.speed = app.speed + 5 ;

```

**Abbildung 4.36: Ausschnitt 1 aus dem Programmcode der Timer-Funktion *startDatenAufnahmeundAnzeige(app,~,~)*.** Abgebildet ist das Senden des RPM-Wertes der Hilfsvariable *app.speed* an die Feedpumpe sowie das Erhöhen des Wertes dieser Variable.



**Abbildung 4.37: Flowchart des Anlaufens der Feedpumpe**

#### 4.2.4.7.2.2 Prozessdatenaufnahme

In diesem Kapitel wird der Programmcode erklärt, welcher für die Aufnahme der Prozessdaten zuständig ist.

##### 4.2.4.7.2.2.1 Waagen

Zu Beginn der Haupttimer *Callback*-Funktion wird der Befehlsstring 'Esc P CR LF' an die Feed- sowie Permeatwaage gesendet (Abbildung 4.38). Dieser löst wie in Kapitel 4.2.4.4 bereits beschrieben die *Callback*-Funktion beider Waagen aus. In diesen Funktionen werden die aktuellen Messwerte der Waagen in den globalen Hauptdatenvariablen *app.GewichtFeedWaage*, *app.t\_feedWaage*, *app.GewichtPermeatWaage* und *app.t\_permeatWaage* gespeichert. In der *Callback*-Funktion der Permeatwaage werden zusätzlich die externen Funktionen zur Berechnung des Flux-Wertes, des Konzentrationsfaktors bzw. des Diafiltrationsvolumen aufgerufen. Die Implementierungen dieser Funktionen werden in den folgenden Unterkapiteln näher erläutert.

```
%Feed Waage Befehl senden
commandFeed= sprintf('%cP\r\n',27); % ESC P CR LF
writeline(app.SeriellesObjektFeedWaage, commandFeed);

pause(0.02)

%Permeat Waage Befehl senden
commandPermeat= sprintf('%cP\r\n',27); % ESC P CR LF
writeline(app.SeriellesObjektPermeatWaage, commandPermeat);

pause(0.07)
```

**Abbildung 4.38:** Ausschnitt 2 aus dem Programmcode der Timer-Funktion *startDatenAufnahmeundAnzeige(app,~,~)*. Der Ausschnitt zeigt das Senden des Befehlsstrings zum Auslösen der Waagen *Callback*-Funktionen.

#### 4.2.4.7.2.2.1.1 Flux-Wert Berechnung

In der Funktion  $f\_permeatFlux(app)$  werden zunächst Kopien der Permeat-Hauptdatenvariablen mit den Namen  $t\_permeatWaage\_cpy$  und  $GewichtPermeatWaage\_cpy$  angefertigt. In dieser externen Funktion (Abbildung 4.39) wird mit diesen Kopie-Variablen gerechnet, um mögliche Fehler durch Überschreibungskomplaktionen auszuschließen. Anschließend wird mit Hilfe der Permeatwaagen Signale und der Formel 2.1 der Permeatfluss durch den Filter in ml/min berechnet. Durch Multiplikation mit 0,06 wird der Permeatfluss in L/h umgerechnet. Die darauffolgende Division mit der Filterfläche (in  $m^2$ ) des Hohlfasermoduls errechnet den aktuellen Flux-Wert in  $\frac{L}{m^2 \cdot h}$ . Da diese Flux-Werte ein starkes Rauschen besitzen, werden die Werte mit Hilfe der Formel 2.8 durch einen  $PT_1$ -Filter gefiltert. Der aktuelle gefilterte Flux-Wert wird in der globalen Variable  $app.permeat\_FLux\_gefiltert$  gespeichert. Am Ende der Funktion werden für den nächsten Funktionsaufruf den Variablen  $app.permeat\_FLux\_gefiltert\_last$ ,  $app.GewichtPermeatWaage\_cpy\_last$  und  $app.t\_permeatWaage\_cpy\_last$  jeweils die aktuellen Werte der dazugehörigen Variable dieses Funktionsaufrufes zugewiesen.

```

function f_permeatFlux(app)
%Copy
t_permeatWaage_cpy = app.t_permeatWaage;
GewichtPermeatWaage_cpy = app.GewichtPermeatWaage;

app.permeateMassFlowMLproMIN = (GewichtPermeatWaage_cpy-app.GewichtPermeatWaage_cpy-last)/(t_permeatWaage_cpy-app.t_permeatWaage_cpy-last);
app.permeateFlowrate = app.permeateMassFlowMLproMIN * 0.06; %In l/h
app.permeatFlux = app.permeateFlowrate / (app.SurfaceAreacm_HollowFiberEditFieId.Value /10000); %In L/m^2*h

%PT1 Filter auf die Flux Werte
app.permeat_FLux_gefiltert = app.permeat_FLux_gefiltert_last + (1*app.permeatFlux - app.permeat_FLux_gefiltert_last)*...
(((t_permeatWaage_cpy-app.t_permeatWaage_cpy-last)/(40/60)+(t_permeatWaage_cpy-app.t_permeatWaage_cpy-last));

%Werte neu schreiben
app.permeat_FLux_gefiltert_last = app.permeat_FLux_gefiltert;
app.GewichtPermeatWaage_cpy_last = GewichtPermeatWaage_cpy;
app.t_permeatWaage_cpy_last = t_permeatWaage_cpy;

end

```

Abbildung 4.39: Programmcode der Funktion **f\_permeatFlux(app)**. Dargestellt sind die Berechnung und Filterung des aktuellen Flux-Wertes.

#### 4.2.4.7.2.2.1.2 Konzentrationsfaktor Berechnung

Der Konzentrationsfaktor wird in der Funktion  $f\_volumeConcentrationFactor(app)$  mit Hilfe der Formel 2.3 berechnet. Dafür wird auch hier eine Kopie der aktuellen Permeat-Hauptdatenvariable  $app.GewichtPermeatWaage$  verwendet. Bei einer einfachen bzw. in einem Multifiltrationsmodus bei der ersten Konzentrierung wird diese lokale Kopie zur Berechnung des aktuellen Volumens des Feedbehälters verwendet (Abbildung 4.40). Wird eine Konzentrierung im Anschluss einer Diafiltration durchgeführt, wird zur Berechnung des aktuellen Volumens des Feedbehälters die lokale Hilfsvariable  $GewichtPermeat\_seit\_2Conc$  verwendet. Diese wird am Anfang der Funktion berechnet. Dafür wird von dem aktuellen Wert der Kopie  $GewichtPermeatWaage\_cpy$  das Volumen im Permeatbehälter zum Startpunkt der zweiten Konzentrierung subtrahiert. Letzteres wird in der globalen Variable  $app.Conc2\_startPermeatGewicht$  bei der Endpunktkontrolle (4.2.4.7.2.5 Endpunktkontrolle) gespeichert. Mit Hilfe einer *if/elseif*-Anweisung und den Filtrationsmodus-Status-Variablen  $app.conc\_status$  und  $app.conc2\_status$  wird geregelt, in was für einer Konzentrierung sich der Prozess gerade befindet und welche Konzentrationsfaktor-Berechnung verwendet wird.

```

if app.conc_status == 0
    app.concentration_Factor = app.EditField_StartVolumen_1stConcentration.Value /...
        (app.EditField_StartVolumen_1stConcentration.Value - GewichtPermeatWaage_cpy);
        %Feed Starvolumen Feld verwendet
elseif app.conc2_status == 0
    app.concentration_Factor = app.EditField_StartVolumen_1stDiafiltration.Value /...
        (app.EditField_StartVolumen_1stDiafiltration.Value-GewichtPermeat_seit_2Conc);
        %Dia Starvolumen Feld verwendet
end

```

**Abbildung 4.40: Ausschnitt aus dem Programmcode der Funktion  $f\_volumeConcentrationFactor(app)$ .** Zu erkennen sind die zwei verschiedenen Berechnungsformeln des Konzentrationsfaktors.

#### 4.2.4.7.2.2.1.3 Diafiltrationsvolumen Berechnung

Auch in der Funktion  $f\_diafiltrationsVolumen(app)$  wird als Erstes eine aktuelle Kopie  $GewichtPermeatWaage\_cpy$  erstellt (Abbildung 4.41). Im Anschluss wird das aktuelle Diafiltrationsvolumen berechnet. Dafür wird die Differenz zwischen der Kopie und dem Volumen im Permeatbehälter zum Startpunkt der Diafiltration durch das Diafiltrationsstartvolumen des Feedbehälters dividiert (Formel 2.4).

```
function f_diafiltrationsVolumen(app)
    %Copy
    GewichtPermeatWaage_cpy = app.GewichtPermeatWaage;

    %Jetztige Diafiltrations Volumen
    app.diafiltrationsVolumen = (GewichtPermeatWaage_cpy-app.Dia_startPermeatGewicht) / app.Dia_startVolumen;
end
```

**Abbildung 4.41: Programmcode der Funktion  $f\_diafiltrationsVolumen(app)$ .** Zu sehen ist die Berechnung des Diafiltrationsvolumens.

#### 4.2.4.7.2.2.2 Druckmonitor

Die Prozessdatenaufnahme der Druckmesswerte wird vollständig in der *Callback*-Funktion  $readSerialDataDruck(app,ser,~)$  des Druckmonitors geregelt. Die Erklärung dieser Funktion kann in Kapitel 4.2.4.4.2 gefunden werden.

#### 4.2.4.7.2.2.3 Pumpen

Da in der TFF-Anlage kein Durchflussmesser verbaut ist, beschränkt sich die Prozessdatenermittlung der Feed- und Diafiltrationspumpe auf den aktuellen RPM-Wert der jeweiligen Pumpe. Zusätzlich wird die Pumpendrehrichtung der Feedpumpe ermittelt. In der Haupttimer *Callback*-Funktion  $startDatenAufnahmeundAnzeige(app,~,~)$  wird an beide Pumpen der Befehlsstring '1 RS CR' gesendet. Daraufhin füllen die Pumpen ihre Empfangspuffer mit den folgenden Daten: Pumpenmodell, aktuelle Drehzahl, aktuelle Drehrichtung, angehalten oder in Betrieb (0/1). Mit dem Befehl  $readline()$  werden diese Daten ausgelesen und jeweils in eine Hilfsvariable  $pumpdata$  gespeichert. Die Datenstrings werden dann mit Hilfe des Befehls  $strsplit()$  in einzelne Strings aufgeteilt.



Für die Prozessdaten der Feedpumpe wird mittels `str2double()` der aktuelle RPM-Wert und die Drehrichtung der Pumpe respektiv in die globalen Variablen `app.pumprate_IstWert` und `app.pumpe_DrehRichtung` als Double-Datentyp gespeichert. Mit Hilfe der Regressionsfunktion der Feedpumpenkalibrierung kann der RPM-Wert in die aktuelle theoretische Flussrate in ml/min umgerechnet werden. Dieser Wert wird in die globale Hauptdatenvariable `app.flowrate_istWert` gespeichert. Die Hauptdatenvariable des dazugehörigen Prozesszeitpunktes `app.t_pumpe` wird wie bereits bei den Waagen ermittelt und gespeichert (4.2.4.4.1 *Callback-Funktionen der Waagen*).

Der aktuelle RPM-Wert der Diafiltrationspumpe wird mit Hilfe der gleichen Befehle wie bei der Feedpumpe bestimmt und in die globale Hauptdatenvariable `app.dia_pumpRPM` gespeichert.

#### **4.2.4.7.2.3 Prozessdatenanzeige**

In diesem Kapitel wird der Programmcode erklärt, welcher für die Anzeige der Prozessdaten in den Reitern *Process Overview*, *Tables* und *Graphs* zuständig ist. Auch für die Prozessdatenanzeige werden zunächst Kopien aller Hauptdatenvariablen erstellt, um mögliche Fehler durch Überschreibungskomplikationen vorzubeugen.

##### **4.2.4.7.2.3.1 Reiter *Process Overview***

Alle aktuellen Werte der Kopien der Hauptdatenvariablen, außer die des Konzentrationsfaktors und des Diafiltrationsvolumens, werden in dem Reiter *Process Overview* dargestellt. Sie werden in der Haupttimer *Callback-Funktion* mit Hilfe des Befehls `string(Variable)` in die Beschriftungsfelder der jeweiligen Gerätepanels abgebildet. Die Kopie der Hauptdatenvariable `app.t_feedWaage` wird in dem Panel "*Process Time*" verwendet, um die aktuelle Prozesszeit anzuzeigen.

#### 4.2.4.7.2.3.2 Reiter *Tables*

In der Tabelle (*app.TabelleAlles*) des Reiters *Tables* werden alle aktuellen Prozessdaten (Hauptdatenvariablen) angezeigt. Zusätzlich wird auch der Prozessname, der Operatorname sowie Datum und die aktuelle Uhrzeit des jeweiligen Messpunktes mit aufgeführt. Bei einer laufenden Konzentrierung wird der Konzentrationsfaktor in der Spalte “CF/DV“ gelistet, während einer Diafiltration das Diafiltrationsvolumen. Dies wird durch eine *if/esleif*-Anweisung und den jeweiligen Filtrationsmodus-Status-Variablen implementiert.

Die genannten Daten werden dafür in der Haupttimer *Callback*-Funktion durch Auflistung in geschweiften Klammern in der lokalen Variable *nr* als *Cell-Array* definiert (Abbildung 4.42). Die darauffolgende Befehlszeile *app.TabelleAlles.Data = [nr;app.TabelleAlles.Data]* realisiert die Anzeige der aktuellsten Daten in der ersten Zeile der Tabelle. Da die Haupttimer *Callback*-Funktion alle 3 Sekunden ausgeführt wird, wird die Tabelle alle 3 Sekunden mit neuen Prozessdaten aktualisiert.

```

%Tabellen Einlese live
if app.conc_status == 0 || app.conc2_status ==0

    NameundOperator = [app.process_Name, ' ', app.operator_Name];
    nr = {NameundOperator, date,char(timeofday(datetime)),sprintf('%.1f',GewichtFeedWaage_cpy),...
        sprintf('%.1f',GewichtPermeatWaage_cpy),sprintf('%.1f', app.TMP),sprintf('%.1f',permeat_FLux_gefiltert_cpy),...
        sprintf('%.1f', app.flowrate_istWert),sprintf('%.3f',concentration_Factor_cpy)};
    app.TabelleAlles.Data = [nr;app.TabelleAlles.Data];

elseif app.Dia_status == 0

    NameundOperator = [app.process_Name, ' ', app.operator_Name];
    nr = {NameundOperator, date,char(timeofday(datetime)),sprintf('%.1f',GewichtFeedWaage_cpy),...
        sprintf('%.1f',GewichtPermeatWaage_cpy),sprintf('%.1f', app.TMP),sprintf('%.1f',permeat_FLux_gefiltert_cpy),...
        sprintf('%.1f', app.flowrate_istWert),sprintf('%.3f',diafiltrationsVolumen_cpy)};
    app.TabelleAlles.Data = [nr;app.TabelleAlles.Data];

end
  
```

**Abbildung 4.42: Ausschnitt 3 aus dem Programmcode der Timer-Funktion *startDatenAufnahmeundAnzeige(app,~,~)*.** Der Ausschnitt zeigt das Füllen der Tabelle aus dem Reiter *Tables* mit den aktuellen Prozessdaten.

#### 4.2.4.7.2.3.3 Reiter *Graphs*

Die Anzeige der Messdaten des Druckmonitors in den Graphen des Reiters *Graphs* wird in der externen Funktion *trendgrafik\_druck(app)* realisiert, die Anzeige aller anderen graphisch dargestellten Prozessdaten (Waagen Signale, Flussrate, Flux-Wert) dagegen in der externen Funktion *trendgrafik(app)* (Abbildung 4.44). Letztere wird in der Haupttimer *Callback*-Funktion aufgerufen, die Funktion *trendgrafik\_druck(app)* in der *Callback*-Funktion des Druckmonitors. Beide Funktionen besitzen die gleiche Struktur und Funktionalität (Abbildung 4.43), verarbeiten jedoch unterschiedliche Prozessdaten. Im Folgenden wird anhand der Funktion *trendgrafik(app)* die Struktur und Funktionalität der beiden Funktionen erklärt.

In der *Startup*-Funktion wurden zwei Matrizen (*app.trendData* und *app.trendDataTime*) deklariert. Eine Matrix für die Prozessdaten, wobei die Spalten die unterschiedlichen Datentypen trennen und die Zeilen die verschiedenen Messpunkte eines Datentyps. Und eine weitere Matrix für die jeweiligen Prozesszeitpunkte der Prozessdaten. Zu Beginn der Funktion wird durch den Befehl *app.trendData(end+1,:) = [0,0,0,0]* eine neue Zeile, gefüllt mit Nullen, an die Datenmatrix angehängen. Das Gleiche wird bei der Zeitmatrix durchgeführt. Im nächsten Schritt werden jeweils alle Elemente der Matrizen mit Hilfe des Befehls *circshift(matrixName,1)* zeilenweise um eine Position verschoben. Die Matrizen besitzen jetzt in der ersten Zeile die hinzugefügten Nullen. Diese Zeilen werden nun mit den aktuellen Prozessdaten bzw. mit den Prozesszeitpunkten gefüllt. Durch das wiederholte Ausführen dieser Funktion füllen sich nach und nach die Matrizen mit den Prozessdaten. Erreichen die Matrizen eine bestimmte Größe (Druck-Matrizen  $\geq 200$  Zeilen, Matrizen der anderen Prozessdaten  $\geq 400$  Zeilen), wird nach dem Verschieben die letzte Zeile der Matrizen gelöscht. Somit sind die Matrizen immer nur mit den Prozessdaten der letzten 20 Minuten gefüllt. Der letzte Schritt der Funktion besteht im Plotten aller Prozessdaten in den einzelnen Graphen sowie in den Mehrfachgraphen. Dafür wird der Befehl *plot()* verwendet.

Die korrekte Anzeige der einzelnen oder mehreren Graphen wird nach Auswahl des Endbenutzers mit Hilfe von *if*-Anweisungen und der jeweiligen Komponenteneigenschaft *app.UIAxesName.Visible = 'on'* bzw. *'off'* implementiert.

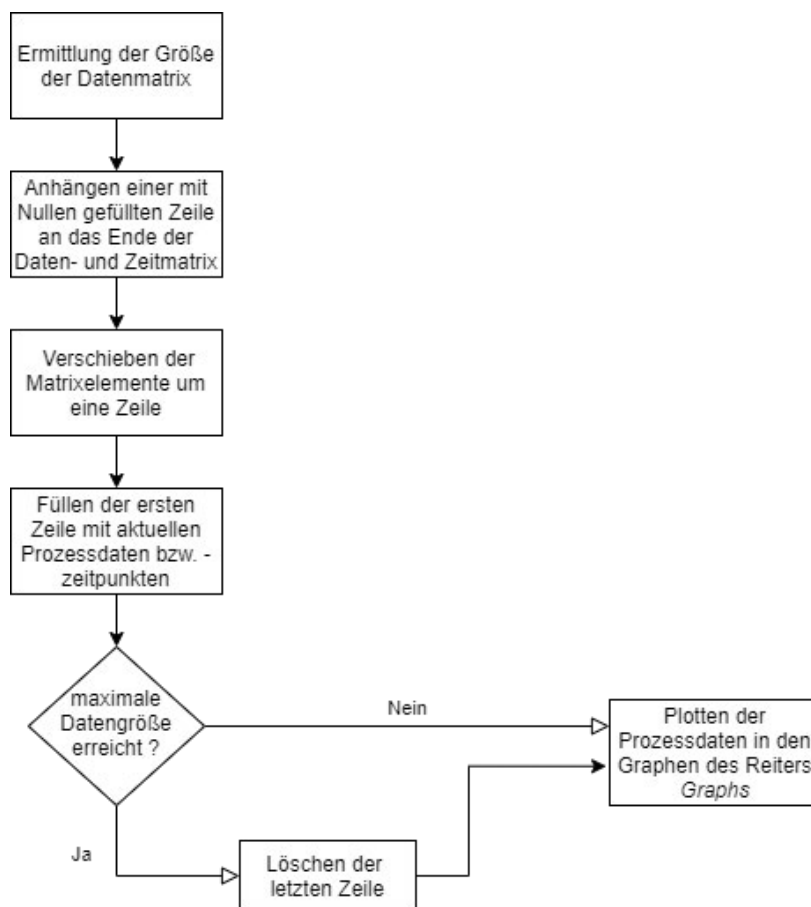
```
function trendgrafik(app)
    % Daten Matrix für die Trendgrafik aktualisieren
    [datenzeilen,~]=size(app.trendData);
    app.trendData(end+1,:) = [0,0,0,0];
    app.trendDataTime(end+1,:) = [0,0,0];

    app.trendData = circshift(app.trendData,1);
    app.trendDataTime = circshift(app.trendDataTime,1);

    app.trendData(1,:) = [app.GewichtFeedWaage, app.GewichtPermeatWaage,...
        app.flowrate_istWert, app.permeat_FLux_gefiltert];
    app.trendDataTime(1,:) = [app.t_feedWaage ,app.t_permeatWaage,app.t_pumpe];

    if (datenzeilen >= 400) % ca 20 Minuten
        app.trendData(end,:) = [];
        app.trendDataTime(end,:) = [];
    end
end
```

**Abbildung 4.44: Ausschnitt aus dem Programmcode der Funktion *trendgrafik(app)*.** Abgebildet ist das Hinzufügen der neuen Matrixzeilen, das zeilenweise Verschieben der Elemente, das Füllen der Matrix mit den aktuellen Prozessdaten sowie das Löschen der letzten Zeile bei Erreichen der maximalen Matrixgröße.



**Abbildung 4.43: Flowchart der Funktion *trendgrafik(app)***

#### 4.2.4.7.2.4 Prozessdatenspeicherung

Wie in Kapitel 4.2.4.6.2 erwähnt, werden die bereits erstellten Dateien für die Prozessdatenspeicherung während des laufenden Prozesses mit den Prozessdaten gefüllt (Abbildung 4.45). Dafür wird bei jedem erneuten Aufruf der Haupttimer *Callback*-Funktion wie schon bereits zum Erstellen der Dateien die externe Funktion *writeDataFile(~,msg,filename)* verwendet. Wie bereits für die Prozessdatenanzeige werden auch für die Prozessdatenspeicherung die Kopie-Variablen der Hauptdatenvariablen genutzt. Ein Datenstring bestehend aus den aktuellen Prozessdaten eines Datentyps (z. B. String der Feedwaagendaten: *sprintf('%6.2f %15.2f',t\_feedWaage\_cpy,GewichtFeedWaage\_cpy)*) wird in eine neue Zeile der jeweils dazugehörigen Datei geschrieben und somit gespeichert. Bei einer laufenden Konzentrierung wird der Konzentrationsfaktor in die Faktorendatei geschrieben. Während einer Diafiltration das Diafiltrationsvolumen. Dies wird mit Hilfe einer *if/elseif*-Anweisung und den jeweiligen Filtrationsmodus-Status-Variablen realisiert.

```

%Faktor Daten
if app.conc_status == 0 || app.conc2_status ==0
    msg=sprintf('%6.2f %17.2f',t_permeatWaage_cpy,concentration_Factor_cpy);
    writeDataFile(app,msg,app.dateiname_autoSave_Faktor);
elseif app.Dia_status == 0
    msg=sprintf('%6.2f %17.2f',t_permeatWaage_cpy,diafiltrationsVolumen_cpy);
    writeDataFile(app,msg,app.dateiname_autoSave_Faktor);
end

```

**Abbildung 4.45: Ausschnitt 4 aus dem Programmcode der Timer-Funktion *startDatenAufnahmeundAnzeige(app,~,~)*.** Der Ausschnitt zeigt das Füllen der Faktordatei mit dem aktuellen Konzentrationsfaktor bzw. dem aktuellen Diafiltrationsvolumen.

#### 4.2.4.7.2.5 Endpunktkontrolle

Der letzte Abschnitt der Haupttimer *Callback*-Funktion *startDatenAufnahmeundAnzeige(app,~,~)* besteht aus der Endpunktkontrolle der geplanten Filtrationsschritte. Mit jeweils drei verschachtelten *if*-Anweisungen wird bei jedem Funktionsaufruf überprüft, ob der eingetragene Endpunktwert des ausgewählten Endpunktyps eines geplanten Filtrationsschrittes erreicht wurde (Abbildung 4.49).

Die erste *if*-Anweisung fragt mit Hilfe der Auswahl der *FiltrationModeButtonGroup* (Nr. 11 in Abbildung 4.13) den ausgewählten Filtrationsmodus ab (z. B. *if app.CDButton.Value == 1*). Diese sind wichtig zu differenzieren, da bei Erreichen eines Endpunktes einer einfachen Konzentrierung oder Diafiltration der Prozess vollständig beendet wird. In einem Multifiltrationsmodus hingegen wird der nächste Filtrationsschritt eingeleitet.

Die zweite *if*-Anweisung überprüft, welcher Endpunktyp im Dropdown-Menü des jeweiligen Filtrationsschrittes vom Endbenutzer ausgewählt wurde. Dies wird durch eine Bedingung wie z. B. “ *if app.EndPointTypeDropDown\_1st\_Concentration.Value == ‘Concentration Factor’* “ festgestellt. Die zweite *if*-Anweisung besitzt eine zusätzliche *if*-Bedingung, die sicherstellt, dass sich der Prozess tatsächlich in dem Filtrationsschritt befindet, für den auch der Endpunktyp ausgewählt wurde. Dieses wird mit Hilfe der Filtrationsmodus-Status-Variablen überprüft.

Sind für den ausgewählten Filtrationsschritt beide *if*-Anweisungen erfüllt, wird die dritte *if*-Anweisungen erreicht. Diese überprüft schließlich, ob der eingetragene Endpunktwert des jeweiligen Filtrationsschrittes erreicht wurde. Zum Beispiel wird im Falle einer Konzentrierung mit dem Endpunktyp “Konzentrationsfaktor“ alle 3 Sekunden kontrolliert, ob der aktuelle Konzentrationsfaktor (*concentration\_Factor\_cpy*) den vom Endbenutzer eingetragenen Endpunktwert (*app.EndPointValueEditField\_1stConcentration.Value*) erreicht oder überschritten hat. Ist dies der Fall, wird der von den *if*-Anweisungen eingerahmte Programmcode ausgeführt.

Handelte es sich bei dem Filtrationsschritt um einen einfachen Filtrationsmodus (C oder D), wird die jeweilige Filtrationsmodus-Status-Variable deaktiviert (Abbildung 4.46) und die Funktion *StopDatenAufnahmeundAnzeige(app)* aufgerufen, welche das Beenden des Prozesses einleitet (4.2.4.7.5 Ende des Prozesses). Ein Dialogfenster informiert den Endbenutzer über das Prozessende.

```

%Concentration (C)
if app.ConcentrationButton.Value == 1
    if app.EndPointTypeDropDown_1st_Concentration.Value == "Concentration Factor " && app.conc_status == 0
        if concentration_Factor_cpy >= app.EndPointValueEditField_1stConcentration.Value

            app.conc_status = 1;

            app.StopDatenAufnahmeundAnzeige;
            msgbox('The desired Concentration Factor was reached, the process ends now !');

        end
    end
end

```

**Abbildung 4.46: Ausschnitt 5 aus dem Programmcode der Timer-Funktion *startDatenAufnahmeundAnzeige(app,~,~)*.** Abgebildet ist die Endpunktkontrolle einer einfachen Konzentrierung mit dem Endpunkttyp "Konzentrationsfaktor".

Handelt es sich um eine Konzentrierung mit anschließender Diafiltration (CD) wird als Erstes die Filtrationsmodus-Status-Variable der ersten Konzentrierung deaktiviert ( $app.conc\_status = 1$ ) und die Filtrationsmodus-Status-Variable der Diafiltration aktiviert ( $app.Dia\_status = 0$ ). Als Nächstes wird die aktuelle Prozesszeit in der globalen Variable  $app.Dia\_startZeit$  und das jetzige Permeatvolumen in der globalen Variable  $app.Dia\_startPermeatGewicht$  gespeichert (Abbildung 4.47). Ebenso wird der Sollwert der Füllstandsregelung während der Diafiltration in der globalen Variable  $app.gewichtSollwert$  definiert. Der Wert entspricht dem von dem Endbenutzer eingetragenen Diafiltrationsstartvolumen. Danach wird das mit Diafiltrationspuffer auszutauschende Volumen berechnet und in der globalen Variable  $app.Dia\_volume\_to\_pump$  gespeichert. Dies wird durch Multiplikation des Startvolumens der Diafiltration mit dem Diafiltrationsvolumen umgesetzt. Die Diafiltrationspumpe wird daraufhin mit dem in den Reglereinstellungen eingegebenen RPM-Startwert gestartet. Schließlich wird der Regelungstimer durch den Befehl  $start(app.dia\_timer)$  aktiviert (4.2.4.7.3 Regelungstimer *Callback-Funktion*), woraufhin dem Endbenutzer ein Informationsdialog erscheint. Dieser bestätigt das Erreichen des Endpunktes und weist auf den folgenden Filtrationsmodus hin. Die Endpunktkontrolle der anschließenden Diafiltration wird nun nach dem gleichen Schema wie bei der bereits beschriebenen einfachen Konzentrierung oder Diafiltration durchgeführt.

```

%Concentration and Diafiltration (CD)
if app.CDButton.Value == 1
    if app.EndPointTypeDropDown_1st_Concentration.Value == "Concentration Factor " && app.conc_status == 0
        if concentration_Factor_cpy >= app.EndPointValueEditField_1stConcentration.Value
            app.conc_status = 1;
            app.Dia_status = 0;

            % Zeit, an dem Umstellen zu Dia passiert, merken
            app.Dia_startZeit = t_feedWaage_cpy;

            % Permeat Gewicht, beim Start von Diafiltration
            app.Dia_startPermeatGewicht = GewichtPermeatWaage_cpy;

            % Zielgewicht der Konzentrierung (=Diafiltrations Start Volumen)
            % als Sollwert für Regelung und zur Berechnung des DV
            app.gewichtSollwert = app.Dia_startVolumen;

            % Dialfiltrations Puffer Volumen berechnen aus DV
            app.Dia_volume_to_pump = app.DV * app.Dia_startVolumen ;

            %Pumpe auf den Anfangspunkt setzen
            commandPumpe= sprintf('1SP%2.2f',round(app.pumpRate_AnfangsPunkt));
            writeline(app.SeriellObjekt_diaPumpe, commandPumpe);
            pause(0.02)
            %Pumpe starten
            commandPumpe= sprintf('1G0');
            writeline(app.SeriellObjekt_diaPumpe, commandPumpe);

            %Start des Regler Timers
            start(app.dia_timer);

            msgbox('The desired Concentration Factor was reached, Diafiltration starts now !');

        end
    end
end

```

**Abbildung 4.47: Ausschnitt 6 aus dem Programmcode der Timer-Funktion *startDatenAufnahmeundAnzeige(app,~,~)*.** Dargestellt ist eine Endpunktkontrolle des Multifiltrationsmodus CD mit dem Endpunkttyp der ersten Konzentrierung "Konzentrationsfaktor".

Hat der Endbenutzer einen CDC-Multifiltrationsmodus geplant, erfolgt die Endpunktkontrolle der ersten Konzentrierung sowie der Übergang in die folgende Diafiltration wie eben im CD-Modus beschrieben. Wird nun der Endpunkt der anschließenden Diafiltration erreicht, wird zunächst die Filtrationsmodus-Status-Variable *app.Dia\_status* deaktiviert und die Filtrationsmodus-Status-Variable der zweiten Konzentrierung aktiviert (*app.conc2\_status = 0*). Die aktuelle Prozesszeit wird in der globalen Variable *app.secondConc\_startZeit* sowie das aktuelle Permeatvolumen in der globalen Variable *app.Conc2\_startPermeatGewicht* gespeichert (Abbildung 4.48). Anschließend wird die Diafiltrationspumpe durch den Befehlsstring '1 ST CR' und der Regelungstimer mittels *stop(app.dia\_timer)* gestoppt. Ein Dialogfenster weist den Endbenutzer auf das Erreichen des Diafiltrationsendpunktes und auf den Start der



zweiten Konzentrierung hin. Die Endpunktkontrolle der anschließenden Konzentrierung wird nach dem gleichen Schema wie bei der bereits beschriebenen einfachen Konzentrierung oder Diafiltration durchgeführt.

```

if app.EndPointTypeDropDown_1st_Diafiltration.Value == "Diafiltration Volume [DV]"...
    && app.Dia_status == 0 && app.conc_status == 1
    if (GewichtPermeatWaage_cpy-app.Dia_startPermeatGewicht) >= app.Dia_volume_to_pump

        app.Dia_status = 1;
        app.conc2_status = 0;

        % Zeit Beim Start der 2. Konzentrierung
        app.secondConc_startZeit = t_feedWaage_cpy;

        % Permeat Gewicht, beim Start von 2. Konzentration
        app.Conc2_startPermeatGewicht = GewichtPermeatWaage_cpy;

        %Regler Timer stoppen
        stop(app.dia_timer);

        %Diafiltrations Pumpe stoppen
        commandPumpe= sprintf('1ST');
        writeline(app.SeriellesObjekt_diaPumpe, commandPumpe);

        msgbox('The desired Diafiltration Volume was reached, the second Concentration starts now !');

    end
end

```

**Abbildung 4.48: Ausschnitt 7 aus dem Programmcode der Timer-Funktion *startDatenAufnahmeundAnzeige(app,~,~)*.** Zu erkennen ist eine Endpunktkontrolle des Multifiltrationsmodus CDC mit dem Endpunkttyp der Diafiltration "Diafiltrationsvolumen".

Hat der Endbenutzer sich durch die *Button Group* im Reiter *Process Setup* für einen manuellen Übergang der Filtrationsschritte entschieden (Nr. 12 in Abbildung 4.13), ist vor jedem Wechsel zum nächsten Filtrationsschritt und vor dem Beenden des Prozesses die Zustimmung des Endbenutzers durch Klicken eines Buttons verpflichtend. Dieser Button erscheint bei Erreichen eines Endpunktes in einem Dialogfenster, in welchem der Endbenutzer gefragt wird, ob der Prozess fortgeföhren werden soll.

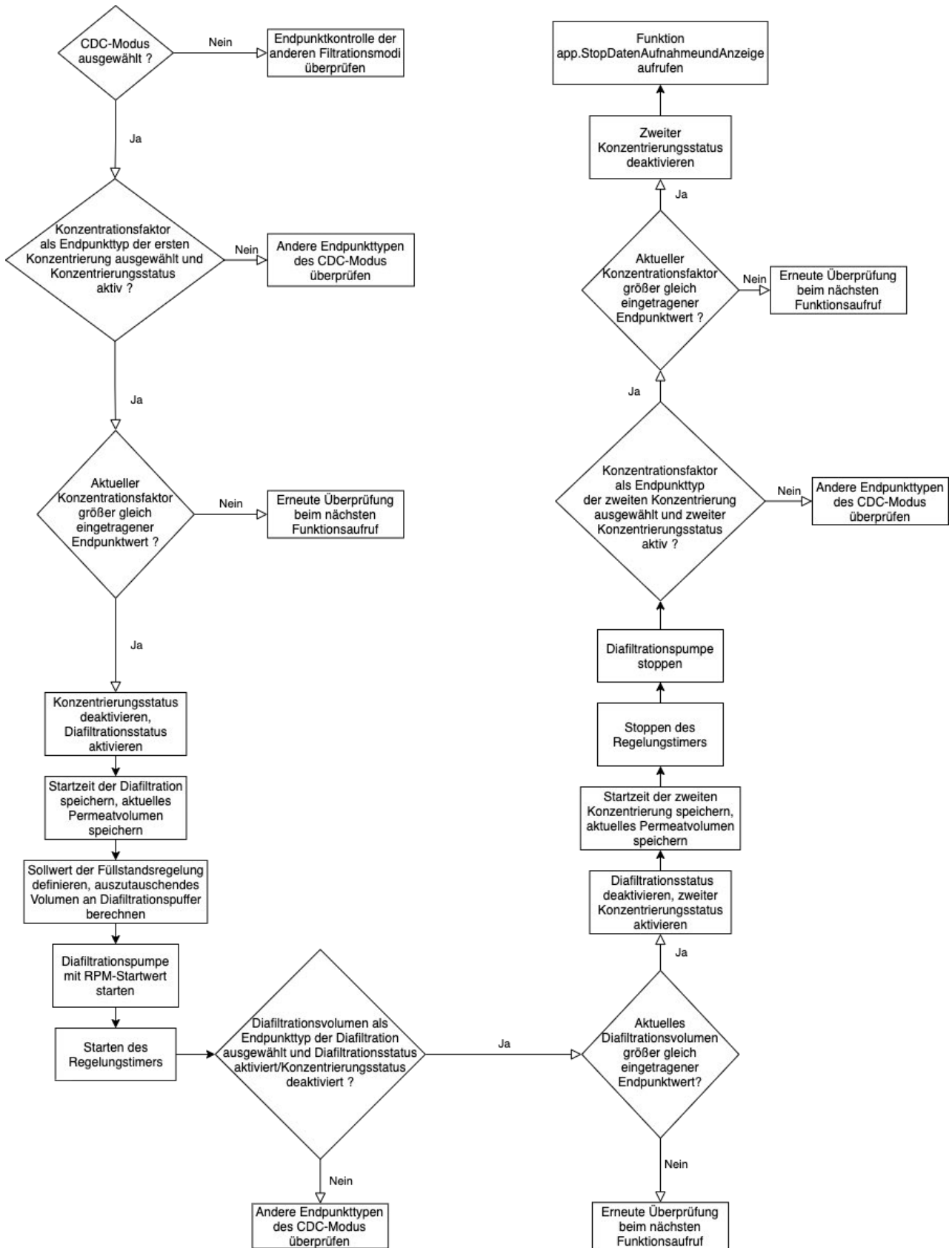


Abbildung 4.49: Beispielhafter Ablauf der Endpunktkontrolle eines CDC-Filtrationsmodus mit den drei Endpunkttypen Konzentrationsfaktor, Diafiltrationsvolumen und Konzentrationsfaktor.

#### 4.2.4.7.3 Regelungstimer *Callback*-Funktion

Der Regelungstimer *app.dia\_timer* wird immer zu Beginn eines Diafiltrationsprozesses gestartet. Seine *Callback*-Funktion *diafiltrations\_Regler\_function (app,~,~)* wird daraufhin jede Sekunde erneut ausgeführt (Abbildung 4.51), die sogenannte Abtastrate des Reglers liegt demnach bei 1 s. Die Funktion ist für die Füllstandsregelung des Feedbehälters während einer Diafiltration zuständig, welche mit Hilfe eines PI-Reglers umgesetzt wird (s. 2.2 Regelungstechnik).

Die vom Endbenutzer im Reiter *Process Setup* eingegebenen Reglerparameter (Nr. 36 und 37 in Abbildung 4.13) werden zu Beginn der Funktion durch Division mit dem eingegebenen maximalen möglichen Fehler der Regelung (Nr. 34 in Abbildung 4.13) normiert und in die globalen Variablen *app.Kp* (Proportionalfaktor) und *app.Ki* (Integrationsfaktor) gespeichert. Mit Hilfe der Matlab<sup>®</sup>-internen Stoppuhrbefehle *tic/toc* wird die vergangene Zeit seit dem letzten Funktionsaufruf in der globalen Variable *app.dt* gespeichert. Durch Berühren der Laborbank oder Anlagenbestandteilen wie z. B. Schläuche kann es zu kurzen Ausreißern im Signal der Feedwaage führen. Da diese nicht direkt eine starke Reaktion des Reglers auslösen sollen, werden die Messwerte der Feedwaage durch einen PT<sub>1</sub>-Filter gefiltert (Formel 2.8) und in der globalen Variable *app.Gewicht\_gefiltert* gespeichert. Dies geschieht jedoch nur, wenn der Endbenutzer im Zahlenfeld Nr. 35 des Reiters *Process Setup* einen Zeitkonstantenwert des Filters eingetragen hat, welches eine *if/else*-Anweisung überprüft.

Im Anschluss wird die Differenz zwischen Sollwert der Regelung (*app.gewichtSollwert*) und gefilterten Ist-Wert des Feedvolumens gebildet und als Regelabweichung in der globalen Variable *app.e* gespeichert (Abbildung 4.50). Auch die Regelabweichung wird durch Division mit dem eingegebenen maximalen möglichen Fehler der Regelung (*app.max\_error*) zu der relativen Regelabweichung (*app.e\_rel*) normiert und gespeichert. Nun werden die einzelnen Anteile der Regler-Gleichung (Formel 2.7) berechnet. Der P-Anteil ergibt sich aus der Multiplikation des Proportionalfaktors *app.Kp* mit der relativen Regelabweichung *app.e\_rel* und wird in der globalen Variable *app.P\_part* gespeichert. Für den I-Anteil wird zunächst bei jedem Funktionsaufruf mit Hilfe der aktuellen und der vorherigen relativen Regelabweichung sowie mit der Trapezmethode (Formel 2.6) eine Summe (*app.summe\_e*) berechnet, welche näherungsweise das Integral der relativen Regelabweichung realisiert. Der I-Anteil (*app.I\_part*) wird durch Multiplikation dieser Summe mit dem Integrationsfaktor *app.Ki* berechnet. Zwei

*if/elseif*-Anweisungen beschränken beide Regler Anteile auf einen Bereich von -10 bis 10. Als nächster Schritt wird die relative Stellgröße in Prozent durch die Befehlszeile  $app.y\_rel = (app.P\_part + app.I\_part) * 100$  ermittelt. Diese wird durch den Befehl  $((app.pumpRate\_AnfangsPunkt/app.RPM\_max)*100) + app.y\_rel$  mit dem Arbeitspunkt der Diafiltrationspumpe verrechnet und das Ergebnis in der globalen Variable *app.pumpRate\_SollWert* gespeichert. Durch einen weiteren *if/elseif*-Block wird der Wert dieser Variable auf einen Bereich von 0 bis 100 beschränkt. Die anschließende Division durch 100 und die Multiplikation mit dem maximalen RPM-Wert der Pumpe berechnet die absolute Stellgröße, den RPM-Wert *app.pumpRate\_SollWert\_send*. Dieser wird in den Befehlsstring zur Änderung der Pumpleistung eingebaut und an die Diafiltrationspumpe gesendet. Als letzter Schritt werden jeweils die aktuellen Werte *app.Gewicht\_gefiltert* und *app.e\_rel* den Variablen *app.Gewicht\_gefiltert\_last* und *app.e\_rel\_last* für den nächsten Funktionsaufruf zugewiesen.

```

%Regelabweichung berechnen
% Regler-Regeldifferenz E = W - X ;
app.e = app.gewichtSollwert - app.Gewicht_gefiltert;

% Normierte Regler-Regeldifferenz e = (w-x) / (w_max - w_min) ; [-1,+1] ohne Einheit
app.e_rel = app.e / app.max_error ;

%PI Regler

%app.summe_e=0;
app.summe_e = app.summe_e + (app.dt * ((app.e_rel + app.e_rel_last)/2));

app.P_part = app.Kp * app.e_rel;
app.I_part = app.Ki * app.summe_e;

%Beschränkung P Part
if app.P_part > 10
    app.P_part = 10;
elseif app.P_part < -10
    app.P_part = -10;
end

%Beschränkung I Part
if app.I_part > 10
    app.I_part = 10;
elseif app.I_part < -10
    app.I_part = -10;
end

%Regler Gleichung
app.y_rel = (app.P_part + app.I_part) *100; % In Prozent

```

**Abbildung 4.50: Ausschnitt aus dem Programmcode der Funktion *diafiltrations\_Regler\_function* (*app,~,~*).** Abgebildet ist die Berechnung der absoluten und relativen Regelabweichung, der Integral-Summe, der Regler Anteile und der relativen Stellgröße. Ebenso zu sehen ist die Beschränkung der Regler Anteile.

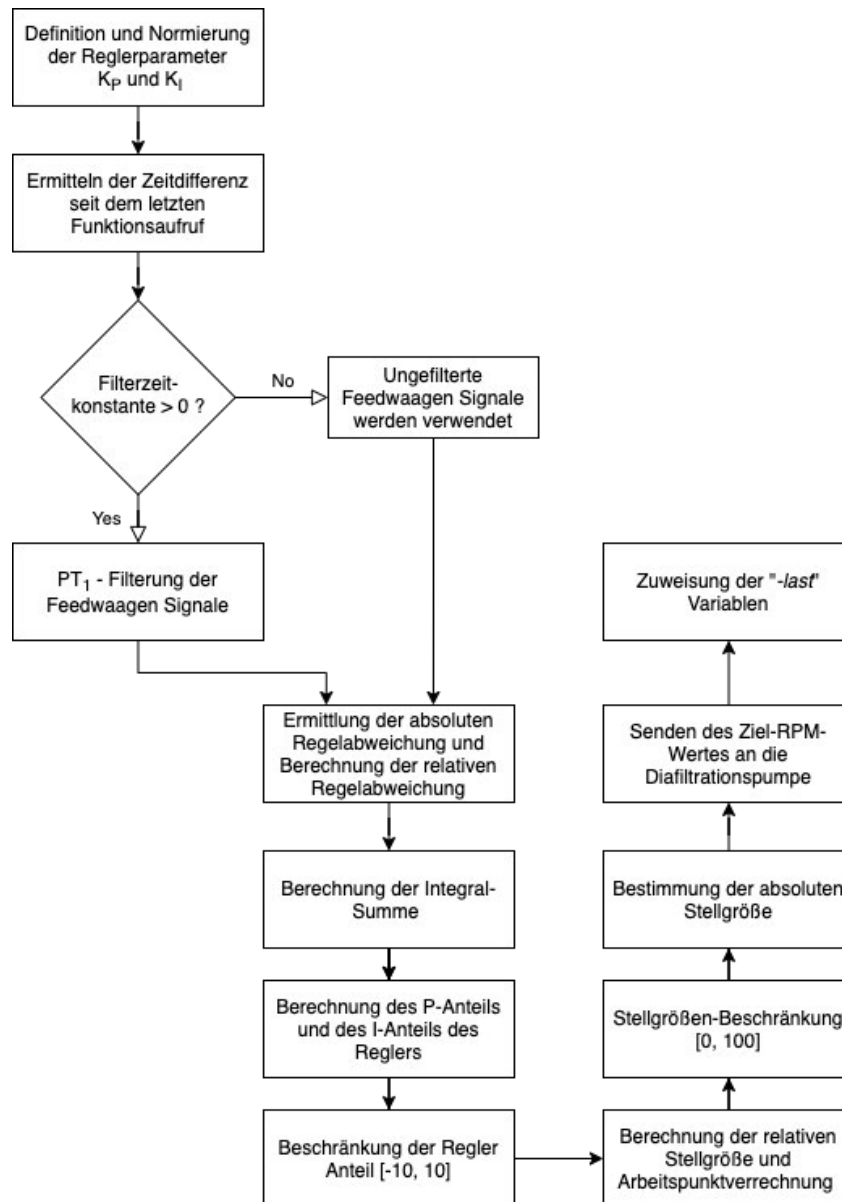


Abbildung 4.51: Flowchart der Regelungstimer *Callback*-Funktion *diafiltrations\_Regler\_funktion (app,~,~)*.

#### 4.2.4.7.3.1 PI-Reglerparameter

Die in der Anwendung vorgegebenen PI-Reglerparameter der Füllstandsregelung wurden experimentell bestimmt und als Wert der Zahlenfelder 36 und 37 (in Abbildung 4.13) gespeichert. Der Endbenutzer hat die Wahl die vorgegebenen Werte zu verwenden oder die Werte nach eigener Wahl zu verändern. Im Folgenden wird der Prozess der experimentellen Bestimmung der vorgegebenen Reglerparameter erläutert. Ziel der Parametrisierung war eine zügige Einregelung des Füllstandes mit akzeptablem Über- bzw. Unterschwingen der Regelgröße.

Für die experimentelle Bestimmung der PI-Reglerparameter wird die TFF-Anlage mit VE-Wasser betrieben. Der Arbeitspunkt des Feedbehälters liegt bei einem Füllstand von 500 ml, die Flussrate wird durch die Feedpumpe auf eine Schergeschwindigkeit von  $6000 \text{ s}^{-1}$  (entspricht 424 ml/min) eingestellt. Die permeatseitige Schlauchklemme ist vollständig geöffnet. Durch leichte Drosselung der retentatseitigen Schlauchklemme nach Prozessstart wird ein TPM von 100 mbar eingestellt. Der Füllstand des Feedbehälters soll durch die Diafiltrationspumpe, welche an den mit VE-Wasser gefüllten Diafiltrationsbehälter angeschlossen ist, geregelt werden. Dafür wird in der Anwendung eine einfache Diafiltration geplant. Der maximale mögliche Fehler der Regelung ( $e_{\max}$ ) wird auf 100 ml, die maximale Pumpleistung auf 400 RPM und der Zeitkonstanten-Wert des Filters auf 5 s eingestellt. Der RPM-Startwert der Diafiltrationspumpe liegt bei 80, da dieser Wert bei dem gegebenen Arbeitspunkt in vorherigen Versuchen eine direkte Einregelung des Füllstandes erzielte. Der Prozess wird über das Programm gestartet. Im eingeregelten Zustand wird durch weiteres Drosseln oder Öffnen der retentatseitigen Schlauchklemme der TMP und somit auch der Permeatfluss durch den Filter (Störgröße) verändert. Dies simuliert einen Sprung der Störgröße. Die Güte der darauffolgenden Regelung des Füllstandes wird anhand der Geschwindigkeit der Anregelung und des maximalen Über- bzw. Unterschwingers (und allgemeine Schwingung des Reglers) bewertet. Dabei werden optimale Reglerparameter durch die folgende Herangehensweise bestimmt:

Zunächst wird nur ein optimaler Proportionalfaktor  $K_P$  ermittelt. Dafür wird der I-Anteil des Reglers deaktiviert. Es wird bei einem Startwert von  $K_P = 10$  begonnen und das eben beschriebene Regelungsexperiment durchgeführt. Nähert sich der Istwert dem Sollwert zu langsam an, wird  $K_P$  verdoppelt und ein weiteres Experiment gestartet. Schwingt der Istwert zu stark und zu lange wird  $K_P$  dagegen verkleinert. Die Proportionalfaktor-Experimente ergaben, dass ein  $K_P$  von 565 ein zügiges Annähern an den Sollwert mit einem einmaligen kurzen Überschwingen ermöglicht. Daraufhin wird der I-Anteil des Reglers wieder aktiviert. Nun wird das gleiche Verfahren auch mit dem Integrationsfaktor  $K_I$  durchgeführt. Es wird bei einem niedrigen Wert begonnen. Erreicht der Istwert nur langsam den Sollwert, wird  $K_I$  erhöht. Reagiert der Regler zu schnell, sodass er stark schwingt oder sogar instabil wird, verringert man  $K_I$  erneut. Die Integrationsfaktor-Experimente ergaben, dass ein  $K_I$  von 300 und ein  $K_P$  von 565 ein schnelles Erreichen des Sollwertes (ca. 4 min) mit geringem Überschwingen (maximale Regelabweichung 2,1 ml) realisiert.

## 4.2.4.7.4 Prozessunterbrechungen

### 4.2.4.7.4.1 Pause

Die Filtration kann während eines laufenden Prozesses durch Klicken des Buttons "PAUSE" (Nr. 1 in Abbildung 4.15) pausiert werden. Die *ButtonPushed-Callback* Aktion des Buttons ruft die Funktion *PauseDatenAufnahmeundAnzeige(app)* auf. In ihr wird zunächst durch den Matlab®-Befehl *tic* ein Zeitstempel gesetzt. Daraufhin überprüft der Befehl *get()* und eine *if*-Anweisung, ob der Regelungstimer aktiv ist. Ist dies der Fall, wird die Füllstandsregelung durch Stoppen des Regelungstimers mittels *stop(app.dia\_timer)* pausiert und die Diafiltrationspumpe durch Senden des Befehlsstrings '1 ST CR' angehalten (Abbildung 4.52). Durch das Zuweisen der Status-Variable *app.dia\_timer\_wurdeGestoppt = 1* wird das Stoppen des Regelungstimers gespeichert. Des Weiteren wird durch Deaktivieren des Haupttimers und der Status-Variable *app.DruckCallback\_Variable* die Prozessdatenaufnahme und -anzeige unterbrochen. Als letzter Schritt wird auch die Feedpumpe durch den Befehlsstring '1 ST CR' gestoppt.

```

%Regler Timer wird gestoppt, wenn er läuft
R = get(app.dia_timer, 'Running');
if isequal(R, 'on')
    stop(app.dia_timer);

%Diafiltrations Pumpe stoppen
commandPumpe= sprintf('1ST');
writeline(app.SeriellesObjekt_diaPumpe, commandPumpe);

app.dia_timer_wurdeGestoppt = 1;
end

%Haupt-Timer wird gestoppt
stop(app.t);

%DruckCallback Variable auf Pause
app.DruckCallback_Variable = 0 ;

%Stop Befehl der Pumpe
commandPumpe= sprintf('1ST');
writeline(app.SeriellesObjektPumpe, commandPumpe);

```

**Abbildung 4.52: Ausschnitt aus dem Programmcode der Funktion *PauseDatenAufnahmeundAnzeige(app)*.** Zu sehen ist die Überprüfung und das eventuelle Stoppen des Regelungstimers, das Deaktivieren des Haupttimers und der Druck-Callback Status-Variable sowie das Stoppen der Feedpumpe.

#### 4.2.4.7.4.2 Fortfahren

Durch Klicken des Buttons “*CONTINUE*” (Nr. 1 in Abbildung 4.15) wird der pausierte Prozess erneut gestartet. Dabei wird die *ButtonPushed-Callback-Funktion ContinueDatenAufnahmeundAnzeige(app)* ausgeführt (Abbildung 4.53). Durch die Befehlszeile  $app.pausenzeit = app.pausenzeit + toc/60$  wird die Zeitperiode der Prozessunterbrechungen gespeichert. Wurde die Status-Variable *app.dia\_timer\_wurdeGestoppt* in der *Callback-Funktion PauseDatenAufnahmeundAnzeige(app)* aktiviert, wird hier die Füllstandsregelung durch Starten der Diafiltrationspumpe (Befehlsstring: ‘1 GO CR’) und des Regelungstimers fortgeführt. Die Status-Variable *app.dia\_timer\_wurdeGestoppt* wird daraufhin deaktiviert. Ebenso wird die Prozessdatenaufnahme und -anzeige durch Aktivieren des Haupttimers und der Status-Variable *app.DruckCallback\_Variable* fortgesetzt. Die Feedpumpe wird nun mit einem sehr niedrigen RPM-Wert ( $app.speed = 1$ ) gestartet. Wie schon beim Anlaufen der Feedpumpe wird die globale Hilfsvariable *app.speed* verwendet, um ein stufenweises Steigern der Flussrate zu erzeugen. In einer *while*-Schleife wird diese Variable in den Befehlsstring zum Ändern der Pumpleistung eingebaut, dieser String an die Feedpumpe gesendet und anschließend der Wert der Variable um zwei RPM erhöht. Der Programmcode der *while*-Schleife wird so lange wiederholt, bis der RPM-Wert der Hilfsvariable den vom Endbenutzer vorgegebenen RPM-Wert überschreitet. Daraufhin wird der vorgegebene RPM-Wert an die Feedpumpe gesendet.



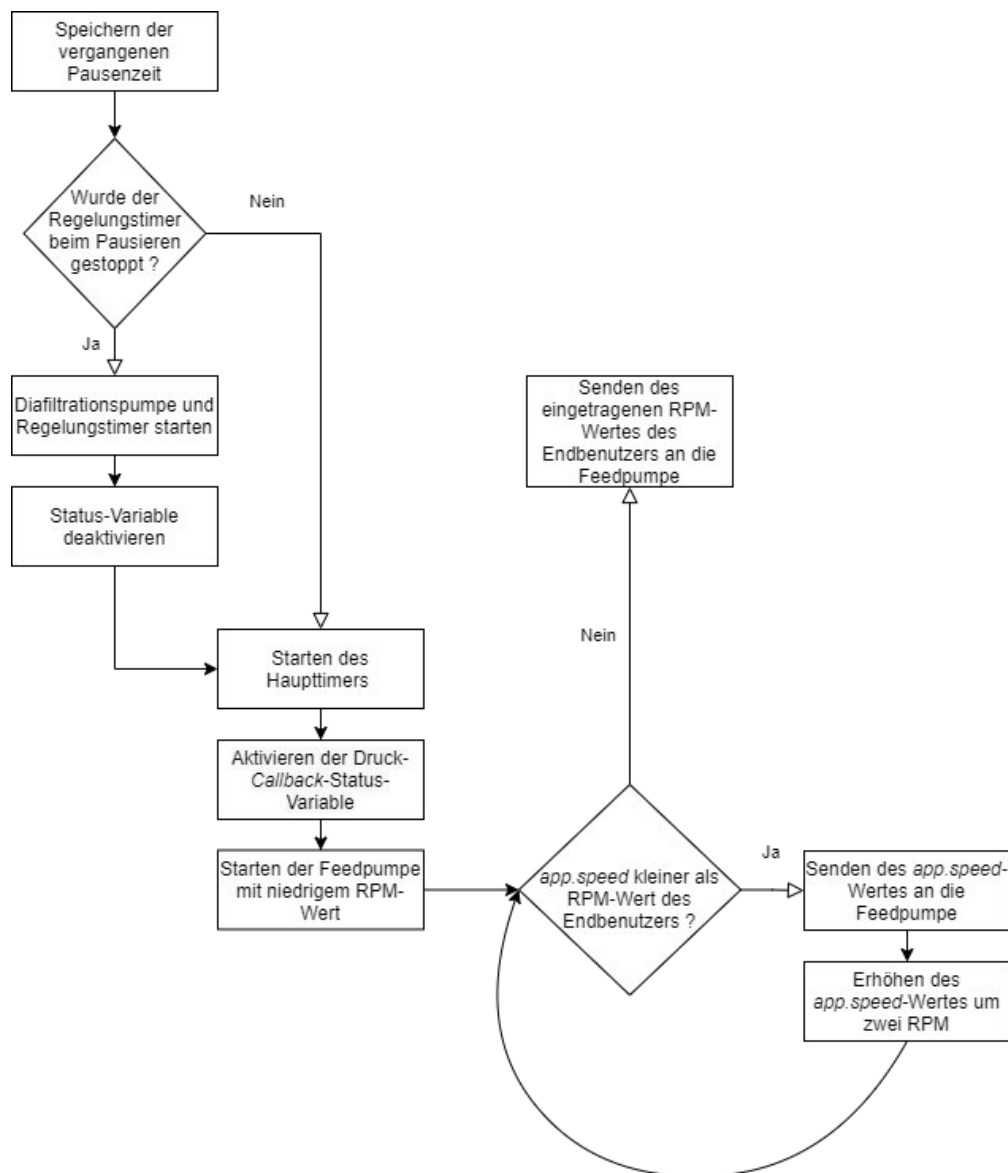


Abbildung 4.53: Flowchart der *Callback-Funktion ContinueDatenAufnahmeundAnzeige(app)*.

#### 4.2.4.7.5 Ende des Prozesses

Wurde der Endpunkt des letzten Filtrationsschrittes erreicht, wird die Funktion *StopDatenAufnahmeundAnzeige(app)* aufgerufen (Abbildung 4.54). Hier werden beide Pumpen durch Senden des Befehlsstrings '1 ST CR' gestoppt. Ebenso wird der Haupttimer und falls aktiv auch der Regelungstimer gestoppt. Als letztes werden die Verbindungen zu den ansteuerbaren Geräten getrennt und die seriellen Objekte gelöscht.

```
% Löschen und säubern des Seriellen Objektes Druckmonitor
delete(app.SeriellesObjektdruckMonitor);
clear app.SeriellesObjektdruckMonitor;

% Löschen und säubern des Seriellen Objektes Feed Waage
delete(app.SeriellesObjektFeedWaage);
clear app.SeriellesObjektFeedWaage;
```

**Abbildung 4.54: Ausschnitt aus dem Programmcode der Funktion *StopDatenAufnahmeundAnzeige(app)*.** Zu erkennen ist das Trennen der Verbindung und das Löschen der seriellen Objekte der ansteuerbaren Geräte.

#### 4.2.4.8 Speicherung der Tabellen Daten

Im Reiter *Save Data* wird durch Klicken des Buttons “Save Table Data“ die *Callback*-Funktion *SaveTableDataButtonPushed(app, event)* ausgeführt. Diese sorgt für die separate Speicherung der Tabelle aus dem Reiter *Tables*. Der Matlab®-Befehl *uiputfile* öffnet ein Dialogfenster zur Pfadauswahl der zu speichernden Datei. Dieser Pfad wird zusammen mit dem String “*Process Data*“ und dem aktuellen Datum zu der lokalen Variable *dateiname* kombiniert. Die Befehlszeile *writecell(app.TabelleAlles.Data,dateiname)* speichert letztlich alle Daten aus der Tabelle in einer Datei, welche unter dem ausgewählten Pfad zu finden ist.

#### 4.2.4.9 *UIFigureClose*-Funktion

Wird die Anwendung durch Klicken der Schaltfläche “X“ in der oberen rechten Ecke des Fensters beendet, wird automatisch die *Callback*-Funktion *UIFigureCloseRequest(app, event)* ausgeführt. In dieser wird zu Beginn die externe Zustandsspeicherungsfunktion *saveState(app)* (4.2.4.3 Funktion zur Zustandsspeicherung und -ladung) aufgerufen, um die eingegebenen Werte dieses Programmaufrufes zu speichern. Um zu gewährleisten, dass alle Timer der Anwendung gestoppt sind, werden diese durch den Matlab®-Befehl *timerfind* gefunden und daraufhin deaktiviert. Ebenso muss sichergestellt werden, dass die Verbindung aller Geräte getrennt wird, da es sonst bei einem erneuten Programmaufruf zu Verbindungs-Fehlern kommen würde. Dies wird durch den MATLAB-Befehl *instrreset* implementiert. Die darauffolgende Zeile *delete(app)* schließt die Anwendung.

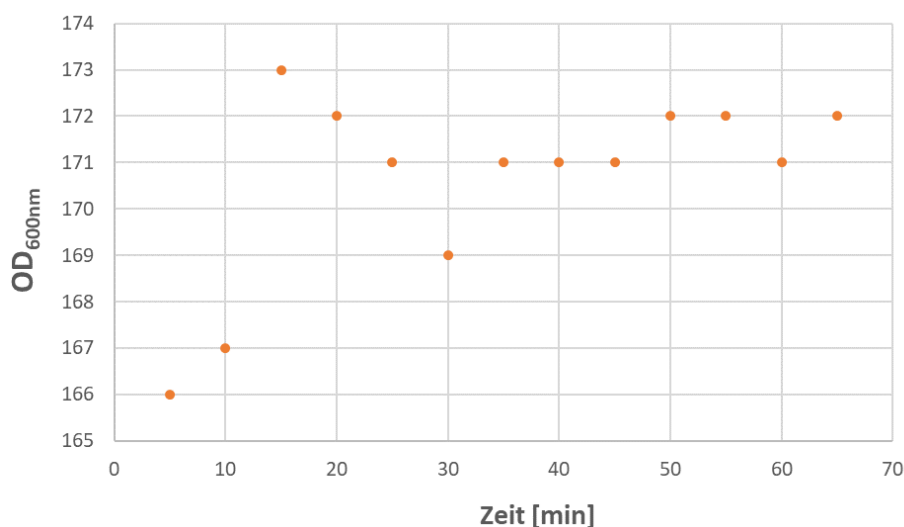
## 5 Praktische Anwendung der TFF-Anlage

### 5.1 Vorversuche

In diesem Abschnitt werden die Ergebnisse der Vorversuche kurz dargelegt und diskutiert.

#### 5.1.1 Aktivierungszeit der suspendierten *Saccharomyces cerevisiae*

Nach der in Kapitel 3.2.1.1 beschriebenen Versuchsdurchführung wurde die *Saccharomyces cerevisiae* Lösung im 5-Minuten-Takt auf ihre optische Dichte untersucht. Die Messergebnisse können dem folgenden Diagramm entnommen werden.

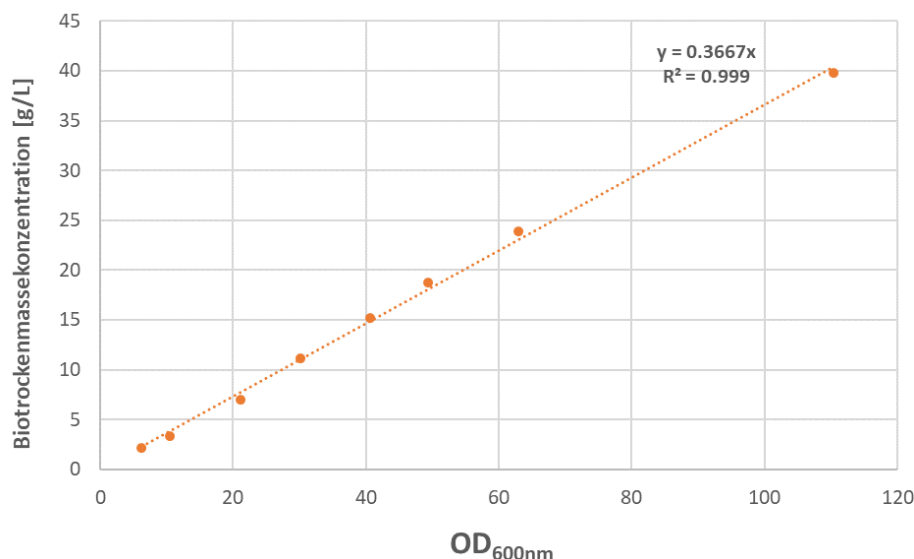


**Abbildung 5.1: Optische Dichte der Hefesuspension nach dem Resuspendieren der getrockneten *Saccharomyces cerevisiae* in Abhängigkeit von der Zeit.**

Abbildung 5.1 zeigt, dass die optische Dichte der Suspension in den ersten 15 min von 166 auf 173 steigt und in weiteren 15 min erneut auf 169 absinkt. Nach ca. 40 min pendelt die optische Dichte bis zum Ende der Messung zwischen den Werten 171 und 172. Aus diesen nahezu statischen Bedingungen lässt sich ableiten, dass die *Saccharomyces cerevisiae* Zellen der Suspension nach ca. 40 min vollständig aktiviert sind.

## 5.1.2 Ermittlung des Korrelationsfaktors zwischen Biotrockenmassekonzentration und $OD_{600nm}$

Acht *Saccharomyces cerevisiae* Suspensionen mit verschiedenen Konzentrationen wurden auf ihre optische Dichte untersucht und einer gravimetrischen Bestimmung der Biotrockenmassekonzentration unterzogen (s. Kapitel 3.2.1.2). Die dabei entstandenen Datensätze sind im Anhang in Tabelle 8.2 abgebildet. Durch Auftragen der ermittelten Biotrockenmassekonzentrationen gegen die dazugehörigen  $OD_{600nm}$  Werte und der Erstellung einer Regressionsgeraden kann der Korrelationsfaktor dieser Parameter ermittelt werden.



**Abbildung 5.2:** Auftragung der Biotrockenmassekonzentrationen gegen die dazugehörigen  $OD_{600nm}$  Werte.

Der in Abbildung 5.2 ermittelte Korrelationsfaktor zwischen der Biotrockenmassekonzentration und der optischen Dichte der Suspensionen liegt bei  $0,3667 \frac{g_{BTM} * L^{-1}}{AU}$ . Ein Bestimmtheitsmaß von 0,999 bestätigt die Güte der Regression. Vergleicht man das eingewogene Gewicht der getrockneten *S. cerevisiae* pro Liter VE-Wasser zum Resuspendieren (g/L) mit der ermittelten Biotrockenmassekonzentration der jeweiligen Suspension (s. Tabelle 8.2) zeigt sich, dass Letzteres jeweils einen kleineren Wert annimmt. Ein möglicher Grund für diesen Unterschied stellt der in der Trockenhefe enthaltene Emulgator dar, welcher durch das Abzentrifugieren der Proben vor der gravimetrischen Bestimmung der Biotrockenmassekonzentration entfernt wird, beim

Einwiegen der getrockneten Hefe jedoch mit gemessen wird. Dieser Umrechnungsfaktor wurde ebenfalls durch Auftragung der beiden Messdaten (s. Abbildung 5.3) ermittelt und liegt bei  $0,7821 \frac{\text{gBTM} \cdot \text{L}^{-1}}{\text{gTrockenhefe} \cdot \text{L}^{-1}}$ . Auch hier bestätigt ein Bestimmtheitsmaß von 0,999 die Güte der Regression.

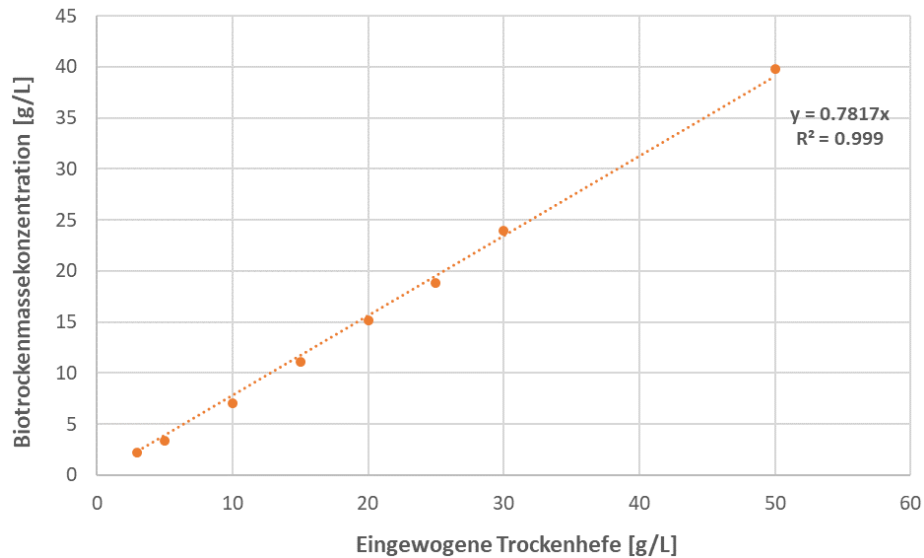
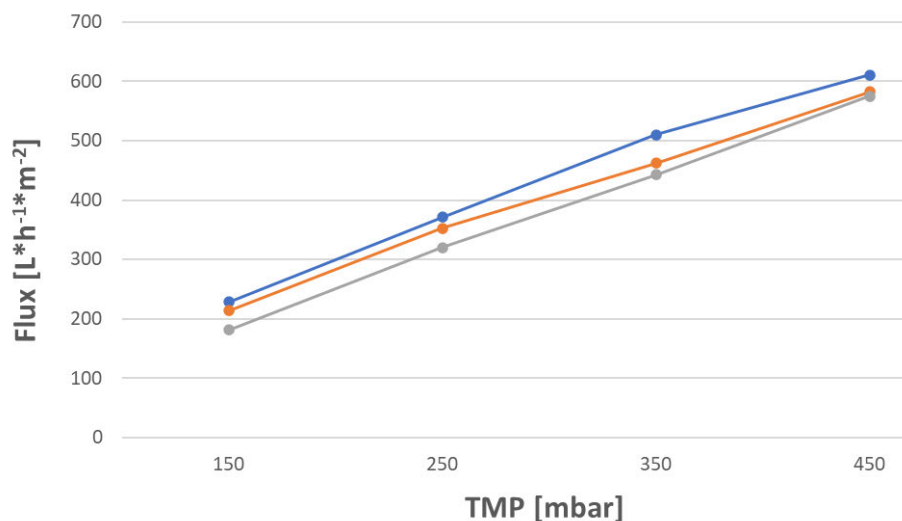


Abbildung 5.3: Auftragung der Biotrockenmassekonzentrationen in g/L gegen die eingewogene Trockenhefe pro L VE-Wasser zum Resuspendieren in g/L.

### 5.1.3 Wasserwert-Versuche des Hohlfasermoduls

Das Hohlfasermodul wird wie in Kapitel 3.2.2 beschrieben mit VE-Wasser bei den Schergeschwindigkeiten 6000, 8000 und 10000 1/s bei den Transmembrandrücken 150, 250, 350 und 450 mbar eingefahren. Die Schergeschwindigkeiten entsprechen respektiv den Überströmungsgeschwindigkeiten  $0,75 \text{ m s}^{-1}$ ,  $1 \text{ m s}^{-1}$  und  $1,25 \text{ m s}^{-1}$ . Abbildung 5.4 zeigt die Ergebnisse dieser Versuche. Der Datensatz zur Erstellung des folgenden Diagramms kann dem Anhang unter Tabelle 8.3 entnommen werden.



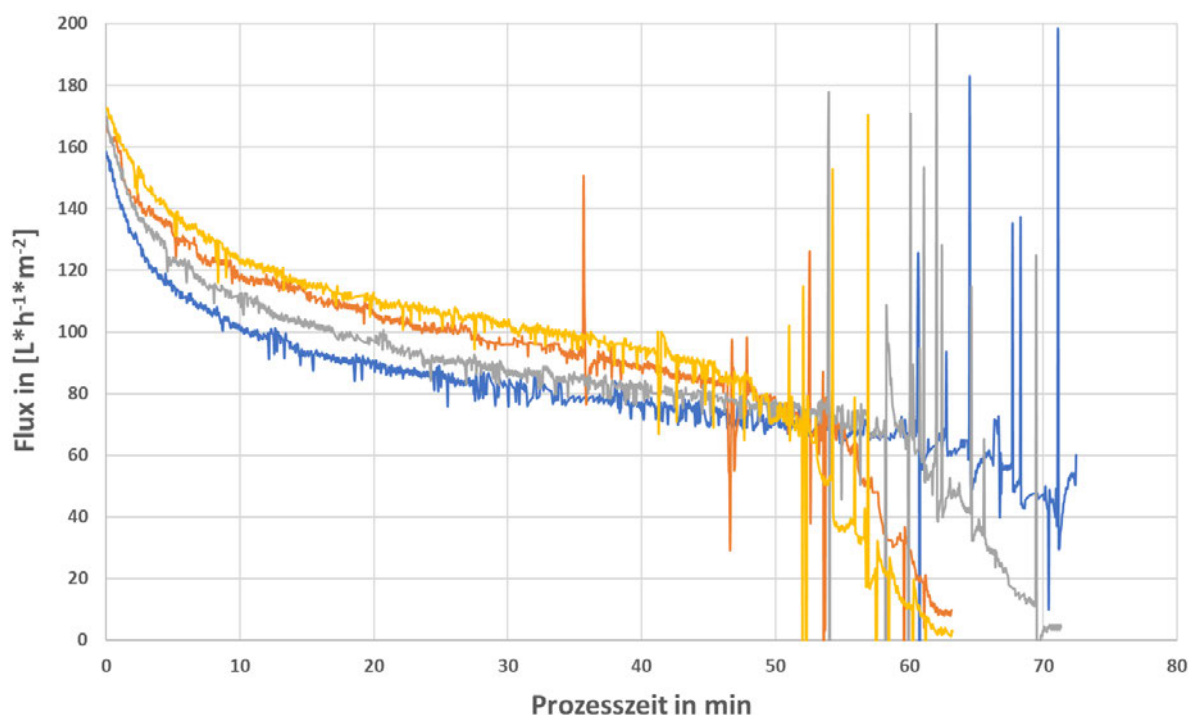
**Abbildung 5.4:** Wasserwerte des Hohlfasermoduls in Abhängigkeit von dem Transmembrandruck bei den Überströmungsgeschwindigkeiten  $0,75 \text{ m s}^{-1}$ ,  $1 \text{ m s}^{-1}$  und  $1,25 \text{ m s}^{-1}$ .

Die Kurvenschar aus dem obigen Diagramm zeigt, dass der Flux unabhängig von der Überströmungsgeschwindigkeit durch eine Erhöhung des Transmembrandruckes nahezu linear ansteigt. Dieses Ergebnis wird von der Literatur bestätigt, da der Transmembrandruck die Triebkraft des Permeatflusses darstellt. Der Flux folgt dem Gesetz von Hagen-Poiseuille, welches aussagt, dass unter idealen Bedingungen (z. B. kein Fouling, keine Deckschichtbildung) der Durchtritt eines Fluids durch eine Membran proportional zum Transmembrandruck ist (Ahmed *et al.*, 2017). Die Ergebnisse zeigen ebenso, dass bei einer höheren Überströmungsgeschwindigkeit der Flux jeweils bei gleichen Transmembrandrücken einen geringeren Wert annimmt. Dies steht jedoch im Widerspruch mit der Theorie. Anhand dieser (s. 2.1.2.2.3 Deckschichtbildung und Fouling) würde bei einem zellfreien Fluid eine Unabhängigkeit der Überströmungsgeschwindigkeit in Bezug auf den Flux erwartet werden, da es zu keiner Deckschichtbildung kommt, welche durch die Überströmungsgeschwindigkeit beeinflusst wird. Ein möglicher Erklärungsansatz für diese Abweichung könnte eine

deutliche Variabilität der Überströmungsgeschwindigkeit darstellen. Da die TFF-Anlage keinen Durchflussmesser besitzt, wird die Überströmungsgeschwindigkeit nur durch die Feedpumpe gesteuert und nicht geregelt. Fehlende Überprüfungen der Pumpenkalibrierung zwischen und nach den Wasserwert-Versuchen in Verbindung mit möglichen Effekten wie z. B. die Abnutzung des Pumpschlauches können somit einen nicht identifizierten Einfluss auf die Überströmungsgeschwindigkeit gehabt haben. Zudem wurden die Wasserwert-Versuche nicht willkürlich, sondern systematisch nacheinander (von kleinen zu großen Überströmungsgeschwindigkeit und TMP) durchgeführt. Durch dieses Vorgehen sind systematische Fehler ebenso nicht auszuschließen.

## 5.2 Konzentrationsversuche der *Saccharomyces cerevisiae* Zellsuspensionen

Die vier Tangentialflussfiltrationen einer *Saccharomyces cerevisiae* Suspension werden nach der in Kapitel 3.2.3 beschriebenen Versuchsdurchführung mit einem Ausgangsvolumen von 1000 ml bei einer Schergeschwindigkeit von 8000 1/s und einem Transmembrandruck von 250 mbar durchgeführt. Die Schergeschwindigkeit entspricht einer Flussrate von 565 ml/min bzw. einer Überströmungsgeschwindigkeit in den Fasern von  $1 \text{ m s}^{-1}$ . Dabei liegt ein Feedvolumen zu Filterflächen-Verhältnis von  $133 \text{ L m}^{-2}$  vor. Die Filtrationen werden nach dem die Zielüberströmung erreicht und der TMP eingestellt ist, durch Öffnen der permeatseitigen Schlauchklemme gestartet. In dem folgenden Diagramm sind die spezifischen Permeatflüsse (Flux) der einzelnen Filtrationen dargestellt. Der Flux-Peak bei ca. 35 min (Konzentrationsversuch 2, orange) ist auf ein Stoßen gegen die Laborbank zurückzuführen, welches das Permeatwaagensignal beeinflusst hat. Alle anderen starken Einbrüche oder Peaks des Flux entstanden durch das manuelle Regeln des TMP mittels der permeatseitigen Schlauchklemme, welches ebenso das Permeatwaagensignal kurzfristig deutlich beeinflusst hat.



**Abbildung 5.5:** Filtrationen einer *Saccharomyces cerevisiae* Suspension mit einer BTM von 50 g/L bei einer Überströmungsgeschwindigkeit von  $1 \text{ m s}^{-1}$  und einem TMP von 250 mbar. Zu sehen sind die spezifischen Permeatflüsse (Flux) der vier Konzentrationsversuche (1,2,3,4) in Abhängigkeit von der Prozesszeit.



Zu Beginn der Filtrationen ergeben sich nach Öffnen der permeatseitigen Schlauchklemme Flux-Werte im Bereich  $158 - 172 \text{ L h}^{-1} \text{ m}^{-2}$ . Es ist zu erkennen, dass in den ersten 15 Minuten (bis  $t = 15 \text{ min}$ ) die Flux-Werte bei allen Versuchen deutlich sinken. Betrachtet man die Mittelwerte des Flux der vier Versuchsreihen, sinkt der spezifische Permeatfluss in dieser Zeitspanne im Durchschnitt um  $61,3 \text{ L h}^{-1} \text{ m}^{-2}$  auf den Bereich  $93 - 111 \text{ L h}^{-1} \text{ m}^{-2}$ . Diese Abnahme lässt sich mit der Ausbildung einer Deckschicht auf der Membranoberfläche zu Beginn der Filtrationen begründen. Im weiteren Verlauf der Filtrationen ist bei allen Versuchen eine geringere, aber kontinuierliche Abnahme des Flux zu beobachten. Von  $t = 15 \text{ min}$  bis  $t = 45 \text{ min}$  sinkt der Flux im Durchschnitt um  $25,3 \text{ L h}^{-1} \text{ m}^{-2}$  auf den Bereich  $70 - 84 \text{ L h}^{-1} \text{ m}^{-2}$ . Dies kann auf die durch das Abführen des Permeats bedingte Aufkonzentrierung des Retentats zurückgeführt werden. Diese bewirkt eine stetige Zunahme der Deckschichtdicke, welches ein zusätzliches Permeationshindernis darstellt und den Flux über die Prozesszeit verringert. Ebenso verändern sich durch diese Konzentrationszunahme die Eigenschaften der Zellsuspension wie z. B. Dichte, Oberflächenspannung oder Viskosität. Dabei könnte die steigende Viskosität die Überströmung der Membran beeinflussen. Zudem können auch Fouling Phänomene als Grund für die kontinuierliche Abnahme des spezifischen Permeatflusses nicht ausgeschlossen werden.

Ab ca.  $t = 45 \text{ min}$  ( $\text{BTM}_{\text{Retentat}} = 155,28 \text{ g/L}$ ) kann eine drastische Verminderung des Flux des ersten Konzentrationsversuches (gelb), ab ca.  $t = 50 \text{ min}$  ( $\text{BTM}_{\text{Retentat}} = 164,6 \text{ g/L}$ ) des zweiten Konzentrationsversuches (orange), ab ca.  $t = 55 \text{ min}$  ( $\text{BTM}_{\text{Retentat}} = 156,9 \text{ g/L}$ ) des dritten Konzentrationsversuches (grau) und ab ca.  $t = 63 \text{ min}$  ( $\text{BTM}_{\text{Retentat}} = 161 \text{ g/L}$ ) des vierten Konzentrationsversuches (blau) festgestellt werden. Daraufhin werden die Filtrationen beendet. Als Grund für diese deutliche Flux-Abnahme am Ende der Filtrationen lässt sich erneut die über den Filtrationsprozess steigende Retentatkonzentration sowie die damit verbundene Verdickung der Deckschicht anführen. Es kann eine Retentatkonzentration erreicht werden, bei der der Strömungswiderstand der ausgebildeten Deckschicht stark ansteigt. In Verbindung mit einer möglichen, durch die Viskositätszunahme bedingte Abnahme der Überströmung, welche die Deckschicht nicht mehr hinreichend abtragen kann, kommt es zum Einbruch des spezifischen Permeatflusses.

Bei allen Konzentrationsversuchen wird während des gesamten Filtrationsprozesses ein klares Permeat beobachtet. Demzufolge ist davon auszugehen, dass die Zellen vollständig von der Membran zurückgehalten werden.

Das obige Diagramm zeigt, dass es unter den vier Konzentrationsversuchen ab  $t = 45$  min zu deutlichen Zeitunterschieden kommt, wann der Flux-Einbruch einsetzt. Aufgrund dieser starken Variabilität und den vielen Ausreißern der Flux-Werte durch die TMP Regelung am Ende der Filtrationen wurden die Messreihen der vier Versuche nur bis Minute 45 statistisch ausgewertet. Flux-Ausreißer in diesem Bereich wurden mit Hilfe der Funktion `boxplot.stats()` der Entwicklungsumgebung "RStudio" des Herstellers *RStudio, Inc.* identifiziert und für die Analyse ausgeschlossen. Es wurden zu jedem Messzeitpunkt (Abtastrate von 3 s, s. 4.2.4.7.2 Haupttimer *Callback*-Funktion) die Flux-Werte der einzelnen Versuche auf ihre Streubreite (Variationskoeffizient) untersucht. Durch Bildung des quadratischen Mittelwertes der Variationskoeffizienten aller Messzeitpunkte entsteht die Gesamtstreuung der vier Versuche. Die relative Variabilität der Konzentrationsversuche der *Saccharomyces cerevisiae* Zellsuspensionen liegt bei einem Gesamt-Variationskoeffizienten von 9,27 % (Standardabweichung dieser Mittelwertbildung = 1,55 %).

Die Forscher um Kendall *et al.* (2002) erzielten bei Tangentialflussfiltrationen von *Escherichia coli* Lysaten eine relative Variabilität des spezifischen Permeatflusses von ca. 6 %. Wiederholungsversuche der Forscher um McDonogh *et al.* (1989) ergaben bei Ultrafiltrationen (TFF) von monodispersen Silica-Partikeln eine Reproduzierbarkeit des Flux von ca. 5 % (Variationskoeffizient). Die im Vergleich höhere Variabilität der Konzentrationsversuche dieser Masterarbeit könnte auf verschiedene Ursachen zurückzuführen sein. Ein möglicher Grund könnte die in Kapitel 5.1.3 (Ergebnisse Wasserwert-Versuche des Hohlfasermoduls) bereits beschriebene Variabilität der Überströmungsgeschwindigkeit aufgrund der fehlenden Flussratenregelung darstellen. Eine weitere Ursache könnte in dem manuellen Regeln des Transmembrandruckes mit Hilfe der Schlauchklemmen liegen. Zum einen erfolgte das manuelle Regeln in zeitlich ungleichmäßigen Abständen, wobei der TMP mal mehr und mal weniger lange und stark vom Sollwert abgewichen ist. Zum anderen hat die Sensitivität der TMP-Einstellung mittels der Schlauchklemmen aufgrund der Viskositätszunahme im Verlauf der Filtration immens zu genommen. Minimale Veränderungen der Schlauch-

klemmen führten vor allem gegen Ende des Filtrationsprozesses zu starken ruckartigen Überschwingern und Unterschwingern des TMP. Insbesondere beim Gegenregeln mit Hilfe der permeatseitigen Schlauchklemme war es dadurch nahezu unmöglich, den TMP-Sollwert exakt einzustellen. Dies könnte auch die Tendenz der besonders starken Variabilität am Ende der Filtrationen (ab  $t = 45$ ) begründen.

Dabei muss erwähnt werden, dass die statistische Auswertung lediglich auf vier Versuchen beruht. Um eine aussagekräftige Varianz von Filtrationen der in dieser Masterarbeit entwickelten Tangentialflussfiltrationsanlage ermitteln zu können, müssen weitere Reproduzierbarkeitsversuche mit der Anlage durchgeführt werden. Dabei könnten durch weitere Anpassungen der TFF-Anlage und des Experimentdesigns Gründe für die bislang hohe Variabilität eingegrenzt oder exakt identifiziert werden. Hierbei sollten weitere Parametersätze aus Schergeschwindigkeit und TMP untersucht werden, um umfangreichere Aussagen über die Reproduzierbarkeit treffen zu können und im Allgemeinen die Etablierung der TFF-Anlage weiter zu forcieren.

## 6 Fazit und Ausblick

Das Ziel dieser Arbeit war die Entwicklung einer Tangentialflussfiltrationsanlage (TFF-Anlage) sowie einer Computer-Anwendung zur Steuerung der Filtrationen über einen externen Rechner. Die graphische Benutzeroberfläche (GUI) sollte dabei eine einfache Planung und Automatisierung von Konzentrations- oder Diafiltrationsprozessen und eine überschaubare, aber ausreichend detaillierte Überwachung und Messdatendokumentation der durchzuführenden Prozesse ermöglichen. Um die Funktionalität der TFF-Anlage sowie der Computer-Anwendung zu überprüfen, sollte als sekundäres Ziel ein Konzentrations- und Diafiltrationsprozess einer *Saccharomyces cerevisiae* Suspension etabliert werden.

Die praktische Anwendung der TFF-Anlage zeigte, dass sie in der Lage ist, eine Zellsuspension unter vollständigem Zellrückhalt zu filtrieren und eine Aufkonzentrierung des Retentats zu erzielen. Dabei realisiert die entwickelte Computer-Anwendung erfolgreich die Feedpumpen-Kalibrierung sowie die vollständige Planung eines Filtrationsprozesses. Diese Planung beinhaltet das Erstellen eines Prozessablaufs durch die Auswahl mehrerer Filtrationsschritte inklusive Endpunktwerte und der Einstellung von Prozessparametern wie Schergeschwindigkeit oder Flussrate. Abgesehen von der manuellen Regelung der Schlauchklemmen ermöglicht die graphische Benutzeroberfläche das automatisierte Durchführen der erstellten Filtrationsprozesse von einem externen Rechner. Tabellarische, graphische sowie schematische Live-Anzeigen von prozessrelevanten Messdaten machen die Überwachung des Prozesses variabel, benutzerfreundlich und zuverlässig. Die Messdatendokumentation erfolgt ebenso automatisiert im laufenden Prozess und steht dem Endbenutzer für weitere Analysen in dem selbst ausgewählten Pfad zur Verfügung. Robustheit gegenüber Programmfehlern oder einer fehlerhaften Bedienung des Endbenutzers konnte während der Konzentrationsversuche bestätigt werden. Die Funktionalität einer Konzentrierung mit anschließender Diafiltration einer zellhaltigen Suspension konnte im zeitlichen Rahmen dieser Abschlussarbeit jedoch nicht gezeigt werden.

Ein zukünftiger Gesichtspunkt der softwareseitigen Weiterentwicklung könnte das Programmieren von auditiven und visuellen Warnungen beim Erreichen von voreingestellten Grenzwerten (z. B. Eingangsdruck des Hohlfasermoduls) sein, die viele kommerziell erhältliche TFF-Anlagen besitzen.

Die Filtrationen der *Saccharomyces cerevisiae* Zellsuspensionen haben gezeigt, dass die Konzentrationsversuche mit einem Variationskoeffizienten von knapp 10 % eine relativ hohe Variabilität besitzen. Zum Ende der Filtrationen nimmt die Streuung der einzelnen Versuche nochmals deutlich zu. Gründe für die hohe Variabilität konnten im Zuge dieser Masterarbeit nicht genau identifiziert werden. Als mögliche Ursachen können die fehlende Überprüfung der Überströmungsgeschwindigkeit und das manuelle Regeln des Transmembrandruckes mittels der retentat- und permeatseitigen Schlauchklemme angeführt werden. Im Hinblick darauf könnten weiterführende Entwicklungen der TFF-Anlage den Einbau eines Durchflussmessers realisieren. Dies würde mit Hilfe der Feedpumpe eine Regelung der Überströmungsgeschwindigkeit ermöglichen. Zudem könnten, wie in einigen kommerziell erhältlichen TFF-Anlagen, ansteuerbare Ventile verbaut werden, um eine zuverlässige und robuste TMP-Regelung zu erlauben. Ebenfalls werden zusätzliche Untersuchungen von weiteren Parametersätzen aus Schergeschwindigkeit und TMP als sinnvoll erachtet.

In nachfolgenden Arbeiten gilt es, die genannten Defizite der TFF-Anlage zu beseitigen und das Experimentdesign zukünftiger Reproduzierbarkeitsversuche zu optimieren und auszuweiten. Damit kann die Etablierung von Konzentrations- und Diafiltrationsprozessen vorangetrieben werden, um das in dieser Masterarbeit zu Grunde gelegte Potential der Tangentialflussfiltrationsanlage und der Computer-Anwendung voll ausschöpfen zu können.

## 7 Literaturverzeichnis

**Ahmed, I.; Balkhair, K. S.; Albeiruttye, M. H.; Shaiban, A. Ahmed Jamil (2017):** Importance and Significance of UF/MF Membrane Systems in Desalination Water Treatment. In: Taner Yonar (Hg.): Desalination: InTech.

**Anspach, B. (2018):** Aufarbeitungs- und Reinigungsverfahren. Vorlesungsskript. Hochschule für Angewandte Wissenschaften Hamburg.

**Arbenz, P. (2008):** Einführung in Matlab. Eidgenössische Technische Hochschule Zürich. Online verfügbar unter <http://people.inf.ethz.ch/arbenz/MatlabKurs/matlabintro.pdf>, zuletzt geprüft am 21.12.2021.

**Baker, R. W. (2004):** Membrane technology and applications. 2. ed. Chichester: Wiley. Online verfügbar unter <http://www.loc.gov/catdir/description/wiley041/2003021354.html>.

**Busch, P. (2012):** Elementare Regelungstechnik. 8., überarbeitete Auflage. Würzburg: Vogel Buchverlag (Vogel-Fachbuch).

**Cornelissen, G. (2020):** Bioprocess Automation. Vorlesungsskript. Hochschule für Angewandte Wissenschaften Hamburg.

**Curran, B. P. G. and Bugeja, V. (2006):** Basic investigations in *Saccharomyces cerevisiae*. In: *Methods in molecular biology (Clifton, N.J.)* 313, S. 1–13. DOI: 10.1385/1-59259-958-3:001.

**Cytiva (2021):** ÄKTA flux tangential flow filtration system. Global Life Sciences Solutions USA LLC. Online verfügbar unter <https://www.cytivalifesciences.com/en/us/shop/bioprocessing-filtration/tangential-flow-filtration/filtration-systems/akta-flux-tangential-flow-filtration-system-p-05755#tech-spec-table>, zuletzt geprüft am 20.12.2021.

**Gasper, H.; Oechsle, D.; Pongratz, E. (2000):** Handbuch der industriellen Fest-/Flüssig-Filtration: Wiley.

**Goedecke, R. (2006):** Fluidverfahrenstechnik. Grundlagen, Methodik, Technik, Praxis: Wiley.

**Goffeau, A.; Barrell, B. G.; Bussey, H.; Davis, R. W.; Dujon, B.; Feldmann, H. et al. (1996):** Life with 6000 genes. In: *Science (New York, N. Y.)* 274 (5287), 546, 563-7. DOI: 10.1126/science.274.5287.546.

**Hartwell, L. H. (1974):** *Saccharomyces cerevisiae* cell cycle. In: *Bacteriological reviews* 38 (2), S. 164–198. DOI: 10.1128/br.38.2.164-198.1974.

**Heinrich, B. (2021):** Grundlagen Regelungstechnik. Einfache Übungen, praktische Beispiele und komplexe Aufgaben. 6., überarbeitete und erweiterte Auflage. Wiesbaden: Springer Vieweg (Lehrbuch). Online verfügbar unter <http://www.springer.com/>.

**Howell, John A. (Hg.) (1993):** Membranes in bioprocessing: theory and applications. 1. ed. London: Chapman & Hall (Elsevier applied biotechnology series).

**Ji, Q.; Mai, J.; Ding, Y.; Wei, Y.; Ledesma-Amaro, R.; Ji, X.-J. (2020):** Improving the homologous recombination efficiency of *Yarrowia lipolytica* by grafting heterologous component from *Saccharomyces cerevisiae*. In: *Metabolic engineering communications* 11, e00152. DOI: 10.1016/j.mec.2020.e00152.

**Karathia, H.; Vilaprinyo, E.; Sorribas, A.; Alves, R. (2011):** *Saccharomyces cerevisiae* as a model organism: a comparative study. In: *PloS one* 6 (2), e16015. DOI: 10.1371/journal.pone.0016015.

**Kayser, O. (2002):** Grundwissen Pharmazeutische Biotechnologie. Wiesbaden: Vieweg+Teubner Verlag.

**Kendall, D.; Lye, G. J.; Levy, M. S. (2002):** Purification of plasmid DNA by an integrated operation comprising tangential flow filtration and nitrocellulose adsorption. In: *Biotechnology and bioengineering* 79 (7), S. 816–822. DOI: 10.1002/bit.10325.

**Köly, C. (2010):** Evaluierung verschiedener Hohlfasermembranen in Bezug auf ihre Eigenschaften zur Zellabtrennung von *Pichia pastoris* und Per.C6@EPCAM aus Zellsuspensionen unter Berücksichtigung der Proteinausbeute. Diplomarbeit. Hochschule für Angewandte Wissenschaften, Hamburg.

**Kraume, M. (2020):** Filtration und druckgetriebene Membranverfahren. In: Matthias Kraume (Hg.): Transportvorgänge in der Verfahrenstechnik. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 277–306.

- Loewe, D.; Dieken, H.; Grein, T. A.; Salzig, D.; Czermak, P. (2022):** A Combined Ultrafiltration/Diafiltration Process for the Purification of Oncolytic Measles Virus. In: *Membranes* 12 (2), S. 105. DOI: 10.3390/membranes12020105.
- Lozano, J. E. (2003):** SEPARATION AND CLARIFICATION. In: *Encyclopedia of Food Sciences and Nutrition*: Elsevier, S. 5187–5196.
- Mcdonogh, R. M.; Fane, A. G.; Fell, C. J.D. (1989):** Charge effects in the cross-flow filtration of colloids and particulates. In: *Journal of Membrane Science* 43 (1), S. 69–85. DOI: 10.1016/S0376-7388(00)82354-6.
- mts Apic (2020):** Funktionsprinzip bei Filterelementen mit Tiefenfiltration. MTS & APIC Filter GmbH & Co.KG. Online verfügbar unter [https://www.mts-apic-filter.de/Produkte/Industrie-Filterelemente/Tiefenfiltration.html#\\_](https://www.mts-apic-filter.de/Produkte/Industrie-Filterelemente/Tiefenfiltration.html#_), zuletzt geprüft am 02.12.2021.
- Murtey, M. DasandRamasamy, P. (2016):** Sample Preparations for Scanning Electron Microscopy – Life Sciences. In: Milos Janecek und Robert Kral (Hg.): *Modern Electron Microscopy in Physical and Life Sciences*: InTech.
- Nikolay, A.; Grooth, J. de; Genzel, Y.; Wood, J. A.; Reichl, U. (2020):** Virus harvesting in perfusion culture: Choosing the right type of hollow fiber membrane. In: *Biotechnology and bioengineering* 117 (10), S. 3040–3052. DOI: 10.1002/bit.27470.
- Ohlrogge, K.; Ebert, K. (2006):** Membranen. Grundlagen, Verfahren und industrielle Anwendungen. Weinheim: Wiley-VCH. Online verfügbar unter <https://onlinelibrary.wiley.com/doi/book/10.1002/3527609008>.
- Parker Hannifin Corporation (2021):** SciLog® SciPres®Pressure Monitor & Sensor. domnick hunter - Process Filtration. Online verfügbar unter [www.parker.com/dhsingleuse](http://www.parker.com/dhsingleuse), zuletzt geprüft am 14.11.2021.
- Rautenbach, R.; Melin, T. (2007):** Membranverfahren. Berlin, Heidelberg: Springer Berlin Heidelberg.
- Razanskiene, A.; Schmidt, J.; Geldmacher, A.; Ritzi, A.; Niedrig, M.; Lundkvist, A. et al. (2004):** High yields of stable and highly pure nucleocapsid proteins of different hantaviruses can be generated in the yeast *Saccharomyces cerevisiae*. In: *Journal of biotechnology* 111 (3), S. 319–333. DOI: 10.1016/j.jbiotec.2004.04.010.



**Repligen (2021):** KrosFlo® KR2i TFF System. Repligen Corporation. Online verfügbar unter <https://www.repligen.com/technologies/krosflo-tff/lab/kr2i>, zuletzt geprüft am 20.12.2021.

**Ripperger, S. (1992):** Mikrofiltration mit Membranen. In: *Starch/Stärke* 47 (4). DOI: 10.1002/star.19950470413.

**Sartorius AG (2021):** Sartoflow® Smart. Small Scale Benchtop System. Sartorius AG. Online verfügbar unter <https://www.sartorius.com/en/products/process-filtration/tangential-flow-filtration/tff-systems/sartoflow-smart>, zuletzt geprüft am 20.12.2021.

**Singh, R. and Purkait, M. K. (2019):** Microfiltration Membranes. In: Randeep Singh und Mihir Kumar Purkait (Hg.): *Membrane Separation Principles and Applications*: Elsevier, S. 111–146.

**Stein, U. (2017):** Programmieren mit MATLAB. Programmiersprache, grafische Benutzeroberflächen, Anwendungen. 6., neu bearbeitete Auflage. München: Fachbuchverlag Leipzig im Carl Hanser Verlag.

**Stieß, M. (1997):** Mechanische Verfahrenstechnik. Berlin, Heidelberg: Springer Berlin Heidelberg.

**Sutherland, K.; Chase, G. (2008):** *Filters and Filtration Handbook*: Elsevier.

**Svaricek, F. (2012):** Digitale Regelung. Vorlesungsskript. Universität der Bundeswehr München.

**The MathWorks, Inc. (2021):** MATLAB - Mathematik. Grafiken. Programmierung. The MathWorks, Inc. Online verfügbar unter [https://de.mathworks.com/products/matlab.html?s\\_tid=hp\\_products\\_matlab](https://de.mathworks.com/products/matlab.html?s_tid=hp_products_matlab), zuletzt geprüft am 21.12.2021.

**Tien, C. (2012):** *Principles of Filtration*: Elsevier.

**Unbehauen, H. (2008):** Klassische Verfahren zur Analyse und Synthese linearer kontinuierlicher Regelsysteme, Fuzzy-Regelsysteme. 15., überarbeitete und erweiterte Auflage. Wiesbaden: Vieweg + Teubner (Studium Automatisierungstechnik, 1).

## 8 Anhang

### 8.1 Abbildungsverzeichnis

<b>Abbildung 2.1: Darstellungen der drei grundsätzlichen Filtrationsarten Kuchenfiltration, Tiefenfiltration und Tangentialflussfiltration (Querstromfiltration).</b> Zu sehen sind die Richtungen der Suspensionsströme und der Filtratströme, das Filtermedium sowie die Filtermittelhilfsschicht (Anspach, 2018). .....	4
<b>Abbildung 2.2: (A) Darstellung einer Kuchenfiltration inklusive Aufbau des Filterkuchens. (B) Abhängigkeit der Schichtdicke des Filterkuchens auf den spezifischen Filtratfluss</b> (verändert nach Kraume, 2020). .....	5
<b>Abbildung 2.3: Illustration einer Kuchenfiltration.</b> Abgebildet sind die mechanischen (rechts) und adsorptiven (links) Abtrennungsmechanismen des Filtermediums (mts Apic, 2020). .....	6
<b>Abbildung 2.4: Arbeitsbereiche der Membranverfahren.</b> Zu erkennen sind die genannten Membranverfahren in Abhängigkeit von der Partikel- bzw. Molekülgröße der abzutrennenden Substanzen sowie bei druckgetriebenen Membranfiltrationen (graue Kästchen) von den typischen Druckdifferenzen. Die mit Pfeil dargestellten Membranverfahren sind durch andere Potentiale getrieben (Anspach, 2018). .....	8
<b>Abbildung 2.5: Modulkonstruktion einer dynamischen Membranfiltration.</b> Zu sehen sind die Ein- bzw. Ausgänge des Moduls für Feed, Retentat und Permeat (verändert nach Lozano, 2003). .....	9
<b>Abbildung 2.6: (A) Darstellung einer statischen Membranfiltration inklusive Aufbau der Deckschicht. (B) Abhängigkeit der Schichtdicke auf den spezifischen Permeatfluss</b> (verändert nach Ripperger, 1992). .....	11
<b>Abbildung 2.7: (A) Darstellung einer dynamischen Membranfiltration inklusive Aufbau der Deckschicht. (B) Abhängigkeit der Schichtdicke auf den spezifischen Permeatfluss</b> (Kraume, 2020). .....	13
<b>Abbildung 2.8: Schematische Darstellung einer symmetrischen und einer asymmetrischen Membran</b> (verändert nach Rautenbach und Melin, 2007). .....	16
<b>Abbildung 2.9: Porengrößenverteilung einer Mikrofiltrationsmembran mit einem nominalen Porendurchmesser von 0,13 <math>\mu\text{m}</math></b> (Rautenbach und Melin, 2007). .....	17
<b>Abbildung 2.10: Bauklassen und -arten von Membrankonstruktionen.</b> Zusätzlich abgebildet sind schematische Darstellungen der Module sowie typische Packungsdichten (Anspach, 2018). .....	19
<b>Abbildung 2.11: Struktureller Aufbau und Betriebsweise von Kapillar- und Hohlfasermusername.</b> Dargestellt sind die Flussrichtungen der Stoffströme sowie die einzelnen Bestandteile des Moduls (Rautenbach und Melin, 2007). .....	20
<b>Abbildung 2.12: Vielfältige Ursachen des Foulings bei porösen Membranen.</b> Abgebildet sind die Mechanismen der irreversiblen Deckschichtbildung, der sterischen Porenverblockung, der inneren Adsorption und des Biofoulings (Rautenbach und Melin, 2007). .....	23

<b>Abbildung 2.13: (A) Mechanismus der sterischen Porenverblockung in Abhängigkeit von der Porengröße. (B) Der Permeatfluss in Abhängigkeit von der Filtrationszeit von großen und kleinen Poren (Anspach, 2018).....</b>	<b>24</b>
<b>Abbildung 2.14: Die Äkta Flux™ S (links) und die Äkta Flux™ 6 (rechts) (Cytiva).....</b>	<b>28</b>
<b>Abbildung 2.15: Das Tangentialflussfiltrations-System KrosFlo® KR2i (Repligen, 2021).....</b>	<b>29</b>
<b>Abbildung 2.16: Das System Sartoflow® Smart (Sartorius AG, 2021).....</b>	<b>30</b>
<b>Abbildung 2.17: Blockbilddarstellung eines Regelkreises. Dargestellt sind die einzelnen Komponenten und Größen eines Regelkreises sowie ihre Wirkung aufeinander.....</b>	<b>31</b>
<b>Abbildung 2.18: Grundstruktur der Timer-Logik. ....</b>	<b>35</b>
<b>Abbildung 2.19: Saccharomyces cerevisiae. Zu sehen ist eine Rasterelektronenmikroskop- Aufnahme von Saccharomyces cerevisiae Zellen (Murtey und Ramasamy, 2016).....</b>	<b>38</b>
<b>Abbildung 3.1: Flowchart von dem Vorbereitungsablauf eines Filtrationsprozesses.....</b>	<b>47</b>
<b>Abbildung 4.1: Schematische Darstellung der TFF-Anlage (verändert nach Loewe et al., 2022). Die gestrichelten Linien stellen eine Anbindung an den externen Computer da.....</b>	<b>50</b>
<b>Abbildung 4.2: Schematische Darstellung der Systemkommunikation zwischen dem externen Laptop und den einzelnen Geräten. P1, P2 und P3 stellen die mobilen Drucksensoren des Moduleingangs, Modulausgangs und des Permeatausganges dar. ....</b>	<b>51</b>
<b>Abbildung 4.3: Nahaufnahme des Feedbehälters. Zu sehen sind die 3 Stützen des Schraubverschlusses des Feedbehälters, sowie der Auslassstützen am Boden der Flasche. Stützen 1 ist auf diesem Bild abweichend vom Text nicht mit der Diafiltrationspumpe verbunden, da hier eine einfache Konzentrierung stattgefunden hat.....</b>	<b>52</b>
<b>Abbildung 4.4: Abbildung des Druckmonitors der mobilen Drucksensoren. Links ist die Vorderseite inklusive Bildschirm dargestellt. Rechts zu erkennen sind die Anschlüsse auf der Rückseite des Monitors (Parker Hannifin Corporation, 2021).....</b>	<b>55</b>
<b>Abbildung 4.5: Nahaufnahme des verbauten Hohlfasermoduls und der mobilen Drucksensoren. Die Abbildung zeigt, wie das Hohlfasermodul über Luer-Lock Schlauchadapter an das Schlauchsystem der Anlage verbunden und mit Hilfe von Stativklammern fest positioniert ist. Ebenso sind die mobilen Drucksensoren am Filtermoduleingang, -ausgang und Permeatausgang sowie ihre Schlauchverbindungen zu erkennen.....</b>	<b>56</b>
<b>Abbildung 4.6: Nahaufnahme der Feedpumpe. Die Abbildung zeigt die Feedpumpe mit beiden Pumpenköpfen und die Anbindung an das Schlauchsystem. ....</b>	<b>57</b>
<b>Abbildung 4.7: Pin-Belegung der seriellen Schnittstelle und des notwendigen seriellen Kabels der Watson Marlow Pumpen. ....</b>	<b>58</b>
<b>Abbildung 4.8: Nahaufnahme der Feedwaage und des darauf platzierten Magnetrührers. ....</b>	<b>59</b>

<b>Abbildung 4.9: Tangentialflussfiltrationsanlage.</b> Zu erkennen sind die einzelnen Baubestandteile: Permeatwaage, Permeatbehälter, Hohlfasermodule, Drucksensoren, Druckmonitor, retentatseitige und permeatseitige Schlauchklemme, 3-Wege-Ventil, Feedpumpe, Feedwaage, Magnetrührer, Feedbehälter und die Diafiltrationspumpe. Die Diafiltrationspumpe sowie der Diafiltrationsbehälter (nicht abgebildet) sind in diesem Bild abweichend vom Text nicht angeschlossen, da dieses Bild während einer laufenden Konzentrierung gemacht wurde. Der Laptop der Anlage befindet sich auf einem Schreibtisch gegenüber von der hier abgebildeten TFF-Anlage. ....	60
<b>Abbildung 4.10: Ausschnitt des Reiters Hardware Setup.</b> Die relevanten Softwarekomponenten (Tabelle 4.2) des Reiters sind mit roten Zahlen markiert und nummeriert. ....	63
<b>Abbildung 4.11: Bildschirmaufnahme 1 des Reiters Calibration.</b> Die relevanten Softwarekomponenten des Reiters (Tabelle 4.3) sind mit roten Zahlen markiert und nummeriert.....	65
<b>Abbildung 4.12: Bildschirmaufnahme 2 des Reiters Calibration.</b> Die relevanten Softwarekomponenten des Reiters (Tabelle 4.3) sind mit roten Zahlen markiert und nummeriert.....	66
<b>Abbildung 4.13: Bildschirmaufnahme des Reiters Process Setup.</b> Die relevanten Softwarekomponenten des Reiters (Tabelle 4.4) sind mit roten Zahlen markiert und nummeriert.....	69
<b>Abbildung 4.14: Bildschirmaufnahme des Reiters Process Overview.</b> Die relevanten Softwarekomponenten des Reiters (Tabelle 4.5) sind mit roten Zahlen markiert und nummeriert.....	72
<b>Abbildung 4.15: Bildschirmaufnahme des Reiters Tables.</b> Die relevanten Softwarekomponenten des Reiters (Tabelle 4.6) sind mit roten Zahlen markiert und nummeriert. ....	74
<b>Abbildung 4.16: Bildschirmaufnahme 1 des Reiters Graphs.</b> Die relevanten Softwarekomponenten des Reiters (Tabelle 4.7) sind mit roten Zahlen markiert und nummeriert. ....	76
<b>Abbildung 4.17: Bildschirmaufnahme 2 des Reiters Graphs.</b> Die relevanten Softwarekomponenten des Reiters (Tabelle 4.7) sind mit roten Zahlen markiert und nummeriert. ....	77
<b>Abbildung 4.18: Ausschnitt des Reiters Save Data.</b> Die relevanten Softwarekomponenten des Reiters (Tabelle 4.8) sind mit roten Zahlen markiert und nummeriert. ....	79
<b>Abbildung 4.19: Auszug aus dem Programmcode der Funktion startupFcn(app).</b> Dargestellt sind einige Initialwert-Zuweisungen der globalen Variablen. ....	82
<b>Abbildung 4.20: Auszug aus dem Programmcode der Funktion loadState(app).</b> Abgebildet sind das Laden und Speichern der state-Variablen. ....	83
<b>Abbildung 4.21: Ausschnitt aus dem Programmcode der Funktion ComPortInitialisierungen(app).</b> Abgebildet ist die serielle Verbindungsherstellung der Feedwaage, ihre Kommunikationsüberprüfung und die Deklaration ihrer Callback-Funktion.....	84
<b>Abbildung 4.22: Flowchart der ButtonPushed-Callback-Funktion ComPortInitialisierungen(app).</b> .....	85

<b>Abbildung 4.23: Ausschnitt aus dem Programmcode der Callback-Funktion readSerialDataFeedWaage(app,ser,~).</b> Zu sehen ist das Einlesen der Feedwaagenmessdaten in die globalen Hauptdatenvariablen. ....	87
<b>Abbildung 4.24: Ausschnitt aus dem Programmcode der Callback-Funktion readSerialDataDruck(app,ser,~).</b> Zu sehen ist das Einlesen der Druckmessdaten in die globalen Hauptdatenvariablen.....	88
<b>Abbildung 4.25: Auszug aus dem Programmcode der Funktion GetCalibrationCurveButtonPushed(app, event).</b> Zu sehen ist das Transponieren der Daten-Vektoren, das Lösen des Gleichungssystems mittels Backslash-Operator und das Berechnen der Werte der Regressionsgeraden. ....	90
<b>Abbildung 4.26: Struktureller Ablauf der Feedpumpenkalibrierung.</b> .....	91
<b>Abbildung 4.27: Programmcode der Funktion Set_FlowRate_EditFieldValueChanged(app, event).</b> Abgebildet ist die Berechnung und anschließende Anzeige der Schergeschwindigkeit nach eingegebener Flussrate. ....	93
<b>Abbildung 4.28: Programmcode der Funktion Set_Shear_EditField_2ValueChanged(app, event).</b> Dargestellt ist die Berechnung und Anzeige der Flussrate in ml/min sowie des RPM- oder Pumpleistungswertes.....	94
<b>Abbildung 4.29: Flowchart der ValueChanged-Callback-Funktion Set_Shear_EditField_2ValueChanged(app, event).</b> .....	95
<b>Abbildung 4.31: Ausschnitt aus dem Programmcode der Funktion SearchpathButtonPushed2(app, event).</b> Abgebildet ist das Öffnen des Pfadialoges, die Erstellung der Dateinamen und das Füllen der Dateien mit Hilfe der Funktion writeDataFile.....	97
<b>Abbildung 4.30: Flowchart der Funktion SearchpathButtonPushed2(app, event).</b> .....	98
<b>Abbildung 4.32: Programmcode der Funktion AutoTareButtonPushed(app, event).</b> Zu sehen ist das Senden des Waagenbefehlsstrings, die Berechnung und Anzeige des Taragewichtes und die Aktivierung des Tariersstatus.....	100
<b>Abbildung 4.33: Callback-Funktionen der Totvolumen-Berechnung.</b> Abgebildet sind die beiden Funktionen StartFillUpButtonPushed2(app, event) und StopFillUpButtonPushed2(app, event) und der dazugehörige Programmcode. ....	100
<b>Abbildung 4.34: Ausschnitt aus dem Programmcode der Funktion STARTButtonPushed(app, event).</b> Dargestellt sind die verschachtelten if-Bedingungen, die Speicherung des Startzeitpunktes, die Aktivierung einzelner Status-Variablen sowie das Starten der Feedpumpe. ....	103
<b>Abbildung 4.35: Flowchart der Funktion STARTButtonPushed(app, event)...</b>	104
<b>Abbildung 4.36: Ausschnitt 1 aus dem Programmcode der Timer-Funktion startDatenAufnahmeundAnzeige(app,~,~).</b> Abgebildet ist das Senden des RPM-Wertes der Hilfsvariable app.speed an die Feedpumpe sowie das Erhöhen des Wertes dieser Variable.....	106
<b>Abbildung 4.37: Flowchart des Anlaufens der Feedpumpe</b> .....	106

<b>Abbildung 4.38: Ausschnitt 2 aus dem Programmcode der Timer-Funktion startDatenAufnahmeundAnzeige(app,~,~).</b> Der Ausschnitt zeigt das Senden des Befehlsstrings zum Auslösen der Waagen Callback-Funktionen.....	107
<b>Abbildung 4.39: Programmcode der Funktion f_permeatFlux(app).</b> Dargestellt sind die Berechnung und Filterung des aktuellen Flux-Wertes.....	109
<b>Abbildung 4.40: Ausschnitt aus dem Programmcode der Funktion f_volumeConcentrationFactor(app).</b> Zu erkennen sind die zwei verschiedenen Berechnungsformeln des Konzentrationsfaktors. ....	110
<b>Abbildung 4.41: Programmcode der Funktion f_diafiltrationsVolumen(app).</b> Zu sehen ist die Berechnung des Diafiltrationsvolumens. ....	111
<b>Abbildung 4.42: Ausschnitt 3 aus dem Programmcode der Timer-Funktion startDatenAufnahmeundAnzeige(app,~,~).</b> Der Ausschnitt zeigt das Füllen der Tabelle aus dem Reiter Tables mit den aktuellen Prozessdaten. ....	113
<b>Abbildung 4.43: Flowchart der Funktion trendgrafik(app)</b> .....	115
<b>Abbildung 4.44: Ausschnitt aus dem Programmcode der Funktion trendgrafik(app).</b> Abgebildet ist das Hinzufügen der neuen Matrizenzeilen, das zeilenweise Verschieben der Elemente, das Füllen der Matrix mit den aktuellen Prozessdaten sowie das Löschen der letzten Zeile bei Erreichen der maximalen Matrixgröße. ....	115
<b>Abbildung 4.45: Ausschnitt 4 aus dem Programmcode der Timer-Funktion startDatenAufnahmeundAnzeige(app,~,~).</b> Der Ausschnitt zeigt das Füllen der Faktordatei mit dem aktuellen Konzentrationsfaktor bzw. dem aktuellen Diafiltrationsvolumen. ....	116
<b>Abbildung 4.46: Ausschnitt 5 aus dem Programmcode der Timer-Funktion startDatenAufnahmeundAnzeige(app,~,~).</b> Abgebildet ist die Endpunktkontrolle einer einfachen Konzentrierung mit dem Endpunktyp "Konzentrationsfaktor". ....	118
<b>Abbildung 4.47: Ausschnitt 6 aus dem Programmcode der Timer-Funktion startDatenAufnahmeundAnzeige(app,~,~).</b> Dargestellt ist eine Endpunktkontrolle des Multifiltrationsmodus CD mit dem Endpunktyp der ersten Konzentrierung "Konzentrationsfaktor". ....	119
<b>Abbildung 4.48: Ausschnitt 7 aus dem Programmcode der Timer-Funktion startDatenAufnahmeundAnzeige(app,~,~).</b> Zu erkennen ist eine Endpunktkontrolle des Multifiltrationsmodus CDC mit dem Endpunktyp der Diafiltration "Diafiltrationsvolumen". ....	120
<b>Abbildung 4.49: Beispielhafter Ablauf der Endpunktkontrolle eines CDC-Filtrationsmodus mit den drei Endpunktypen Konzentrationsfaktor, Diafiltrationsvolumen und Konzentrationsfaktor</b> .....	121
<b>Abbildung 4.50: Ausschnitt aus dem Programmcode der Funktion diafiltrations_Regler_function (app,~,~).</b> Abgebildet ist die Berechnung der absoluten und relativen Regelabweichung, der Integral-Summe, der Regler Anteile und der relativen Stellgröße. Ebenso zu sehen ist die Beschränkung der Regler Anteile. ....	123
<b>Abbildung 4.51: Flowchart der Regelungstimer Callback-Funktion diafiltrations_Regler_function (app,~,~).</b> .....	124

<b>Abbildung 4.52: Ausschnitt aus dem Programmcode der Funktion PauseDatenAufnahmeundAnzeige(app).</b> Zu sehen ist die Überprüfung und das eventuelle Stoppen des Regelungstimers, das Deaktivieren des Haupttimers und der Druck-Callback Status-Variable sowie das Stoppen der Feedpumpe. ....	126
<b>Abbildung 4.53: Flowchart der Callback-Funktion ContinueDatenAufnahmeundAnzeige(app).</b> .....	128
<b>Abbildung 4.54: Ausschnitt aus dem Programmcode der Funktion StopDatenAufnahmeundAnzeige(app).</b> Zu erkennen ist das Trennen der Verbindung und das Löschen der seriellen Objekte der ansteuerbaren Geräte. ....	129
<b>Abbildung 5.1: Optische Dichte der Hefesuspension nach dem Resuspendieren der getrockneten Saccharomyces cerevisiae in Abhängigkeit von der Zeit.</b> .....	130
<b>Abbildung 5.2: Auftragung der Biotrockenmassekonzentrationen gegen die dazugehörigen OD<sub>600nm</sub> Werte.</b> .....	131
<b>Abbildung 5.3: Auftragung der Biotrockenmassekonzentrationen in g/L gegen die eingewogene Trockenhefe pro L VE-Wasser zum Resuspendieren in g/L.</b> .....	132
<b>Abbildung 5.4: Wasserwerte des Hohlfasermoduls in Abhängigkeit von dem Transmembrandruck bei den Überströmungsgeschwindigkeiten 0,75 m s<sup>-1</sup>, 1 m s<sup>-1</sup> und 1,25 m s<sup>-1</sup>.</b> .....	133
<b>Abbildung 5.5: Filtrationen einer Saccharomyces cerevisiae Suspension mit einer BTM von 50 g/L bei einer Überströmungsgeschwindigkeit von 1 m s<sup>-1</sup> und einem TMP von 250 mbar.</b> Zu sehen sind die spezifischen Permeatflüsse (Flux) der vier Konzentrationsversuche (1,2,3,4) in Abhängigkeit von der Prozesszeit. ....	135
<b>Abbildung 8.1: Schergeschwindigkeits-Graph des Hohlfasermoduls.</b> Graphisch und tabellarisch zu sehen sind die benötigten Flussraten durch das Modul, um gewünschte Schergeschwindigkeiten zu erzeugen.....	x

## 8.2 Tabellenverzeichnis

<b>Tabelle 2.1: Callback-Aktionen der in dieser Masterarbeit am meisten vorkommenden UI-Komponenten.</b> .....	37
<b>Tabelle 3.1: Verwendete Geräte.</b> .....	39
<b>Tabelle 3.2: Verwendete Verbrauchsmaterialien.</b> .....	40
<b>Tabelle 3.3: Verwendete Chemikalien und Agenzien.</b> .....	41
<b>Tabelle 3.4: Verwendete Lösungen.</b> .....	41
<b>Tabelle 4.1: Technische Daten des verwendeten Hohlfasermoduls.</b> .....	54
<b>Tabelle 4.2: Softwarekomponenten des Reiters Hardware Setup.</b> Dazugehörige Callback Aktionen und Nummern der Bildschirmaufnahme sind mit aufgelistet. ....	64
<b>Tabelle 4.3: Softwarekomponenten des Reiters Calibration.</b> Dazugehörige Callback Aktionen und Nummern der Bildschirmaufnahme sind mit aufgelistet. ....	66
<b>Tabelle 4.4: Softwarekomponenten des Reiters Process Setup.</b> Dazugehörige Callback Aktionen und Nummern der Bildschirmaufnahme sind mit aufgelistet. ....	70
<b>Tabelle 4.5: Softwarekomponenten des Reiters Process Overview.</b> Dazugehörige Callback Aktionen und Nummern der Bildschirmaufnahme sind mit aufgelistet. ....	73
<b>Tabelle 4.6: Softwarekomponenten des Reiters Tables.</b> Dazugehörige Callback Aktionen und Nummern der Bildschirmaufnahme sind mit aufgelistet. ....	75
<b>Tabelle 4.7: Softwarekomponenten des Reiters Graphs.</b> Dazugehörige Callback Aktionen und Nummern der Bildschirmaufnahme sind mit aufgelistet. ....	78
<b>Tabelle 4.8: Softwarekomponenten des Reiters Save Data.</b> Dazugehörige Callback Aktionen und Nummern der Bildschirmaufnahme sind mit aufgelistet. ....	79
<b>Tabelle 4.9: Serielle Kommunikationseinstellungen und -protokolle von den Geräten der TFF-Anlage.</b> Das X in dem Befehlscode stellt ein beliebigen Befehlsbuchstabe dar. CR steht für Carriage return (deu.: Wagenrücklauf), LF für Line feed (deu.: Zeilenvorschub) und Esc für Escape-Taste. ....	80
<b>Tabelle 8.1: Einwaagen und Verdünnungen zur Erstellung der Hefesuspensionen.</b> Alle Suspensionen wurden auf ein Volumen von 20 ml eingestellt. ....	viii
<b>Tabelle 8.2: Messergebnisse der optischen Dichte-Bestimmung und der gravimetrischen Bestimmung der Biotrockenmassekonzentration.</b> .....	viii
<b>Tabelle 8.3: Messdaten der Wasserwert-Bestimmung des Hohlfasermoduls.</b> .....	ix
<b>Tabelle 8.4: Liste der Namen und Beschreibungen der globalen Variablen des Programmcodes.</b> .....	xi



## 8.3 Rohdaten

### 8.3.1 Vorversuche

#### 8.3.1.1 Bestimmung des Korrelationsfaktors $c_x/OD_{600nm}$

Tabelle 8.1: Einwaagen und Verdünnungen zur Erstellung der Hefesuspensionen. Alle Suspensionen wurden auf ein Volumen von 20 ml eingestellt.

Konzentration [g/L]	Einwaagen [g] bzw. Verdünnungen
3	1:10 Verd. aus 30 g/L
5	1:2 Verd. aus 10 g/L
10	1:2 Verd. aus 20 g/L
15	1:2 Verd. aus 30 g/L
20	0.4
25	1:2 Verd. aus 50 g/L
30	0.6
50	1

Tabelle 8.2: Messergebnisse der optischen Dichte-Bestimmung und der gravimetrischen Bestimmung der Biotrockenmassekonzentration.

Proben ein- gewogen	OD <sub>600</sub> Messung	BTM 1			BTM 2			Mittel- wert BTM
		$m_{leer}$	$m_{Pellet}$	$c_x$ , BTM, 1	$m_{leer}$	$m_{Pellet}$	$c_x$ , BTM, 2	
g/l	-	g	g	g/l	g	g	g/l	g/l
3	6.240	1.0207	1.0229	2.2	1.0177	1.0199	2.2	2.2
5	10.500	1.0177	1.0210	3.3	1.0166	1.0200	3.4	3.3
10	21.100	1.0258	1.0328	7.0	1.0228	1.0298	7.0	7.0
15	30.100	1.0179	1.0289	11.0	1.0143	1.0255	11.2	11.1
20	40.700	1.0195	1.0345	15.0	1.0229	1.0382	15.3	15.2
25	49.300	1.0166	1.0352	18.6	1.0241	1.0431	19.0	18.8
30	63.000	1.0305	1.0541	23.6	1.0164	1.0406	24.2	23.9
50	110.400	1.0186	1.0579	39.3	1.0177	1.0580	40.3	39.8

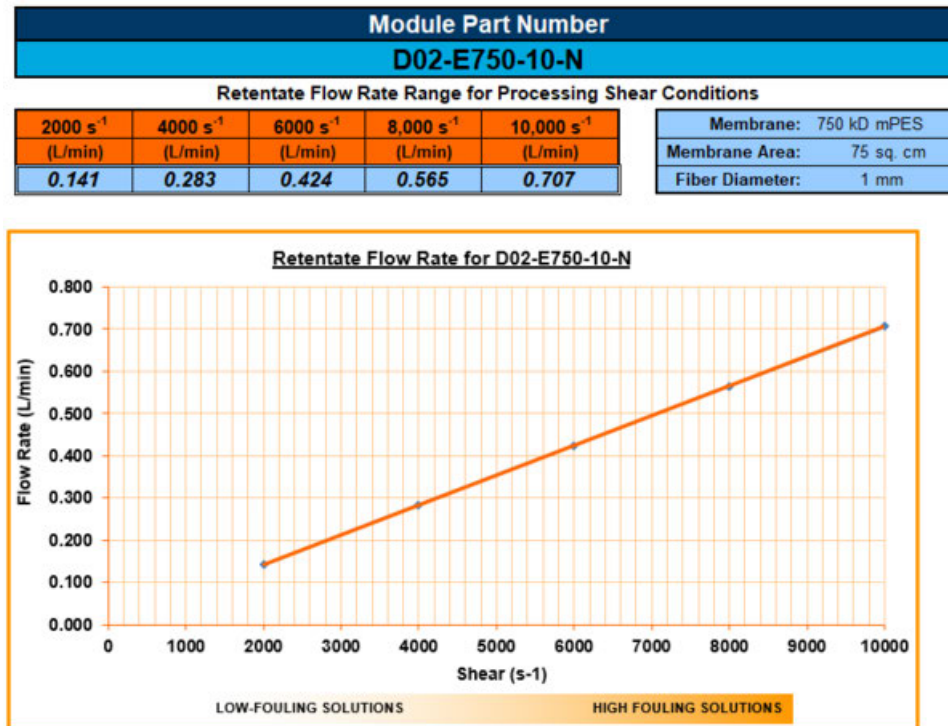
### 8.3.1.2 Wasserwert-Bestimmung des Hohlfasermoduls

Tabelle 8.3: Messdaten der Wasserwert-Bestimmung des Hohlfasermoduls.

Schergeschwindigkeit [1/s]	Flussrate [ml/min]	Überströmungsgeschwindigkeit [m/s]	TMP [mbar]	Flux [l/h*m <sup>2</sup> ]
6000	424	0.75	150	228.5
6000	424	0.75	250	371.3
6000	424	0.75	350	509.9
6000	424	0.75	450	610.9
8000	565	1.00	150	213.4
8000	565	1.00	250	352.3
8000	565	1.00	350	462.2
8000	565	1.00	450	582.7
10000	707	1.25	150	180.9
10000	707	1.25	250	320.2
10000	707	1.25	350	443.0
10000	707	1.25	450	575.5

## 8.4 Hardware-Informationsblätter

### 8.4.1 Hohlfasermodul



**Abbildung 8.1: Schergeschwindigkeits-Graph des Hohlfasermoduls.** Graphisch und tabellarisch zu sehen sind die benötigten Flussraten durch das Modul, um gewünschte Schergeschwindigkeiten zu erzeugen.

## 8.5 Graphische Benutzeroberfläche

### 8.5.1 Globale Variablen

Tabelle 8.4: Liste der Namen und Beschreibungen der globalen Variablen des Programmcodes.

Name	Beschreibung
autoSave_status	Variable welche aussagt, ob Autosave ausgewählt wurde oder nicht.
AutoTaraGewicht	Gewicht welches beim internen Trieren abgezogen wird
berechneteFlowrate_vonOperator_Vorgabe_Prozent_RPM	Flussrate die durch Regressionsgerade in RPM Flussrate für die Pumpe ungerechnet wurde
comPort_diaPump	Portstring der Diafiltrationspumpe
comPortDruckmonitor	Portstring des Druckmonitors
comPortFeedWaage	Portstring der Feedwaage
comPortPermeatWaage	Portstring der Permeatwaage
comPortPump	Portstring der Feedpumpe
conc_status	Variable welche 1 wird, wenn erster Konzentrationsendpunkt erreicht wurde (In CD-Modus)
Conc2_startPermeatGewicht	Permeatgewicht an dem die 2. Konzentration gestartet wird
conc2_status	Variable welche 1 wird, wenn zweiter Konzentrationsendpunkt erreicht wurde (In CD-Modus)
concentration_Factor	Konzentrationsfaktor im Feedbehälter
datasizeDruck	Datenmengen des Druck-Plots
datasizeWaagen	Datenmengen der Waagen-Plots
dateiname_autoSave_Druck	Kompletter Name der Druck Daten Files für die automatische Speicherung der Daten
dateiname_autoSave_Faktor	Kompletter Name der Faktor Daten Files für die automatische Speicherung der Daten
dateiname_autoSave_Faktor	Kompletter Name der Faktor Daten Files für die automatische Speicherung der Daten
dateiname_autoSave_FeedGewicht	Kompletter Name des Feed Daten Files für die automatische Speicherung der Daten
dateiname_autoSave_Flowrate	Kompletter Name der FlowrateDaten Files für die automatische Speicherung der Daten
dateiname_autoSave_Flowrate	Kompletter Name der FlowrateDaten Files für die automatische Speicherung der Daten

Name	Beschreibung
dateiname_auto-Save_Flux	Kompletter Name der Flux Daten Files für die automatische Speicherung der Daten
dateiname_auto-Save_PermeatGewicht	Kompletter Name des Permeat Daten Files für die automatische Speicherung der Daten
dead_Volume	Totvolumen der Anlage
dia_pumpRPM	Pumprate der Diafiltrationspumpe
Dia_startPermeatGewicht	Permeatgewicht an dem die Diafiltration gestartet wird
Dia_startVolumen	Volumen des Feedbehälters beim Start der Diafiltration
Dia_startZeit	Zeit an dem die Diafiltration gestartet wird
Dia_status	Variable welche 1 wird, wenn Diafiltrationsendpunkt erreicht wurde (In CDC Modus)
dia_timer	Timer des Reglers für die Diafiltration
dia_timer_wurdeGestoppt	Variable welche 1 ist, wenn Dia_Timer lief und durch Pause Button gestoppt wurde
Dia_volume_to_pump	Volumen, welches von Diafiltrationspumpe gepumpt werden muss (entspricht Umrechnung mit Diafiltrationsvolumen als Endpunkt)
diafiltrationsVolumen	Das jetzige Diafiltrationsvolumen
DruckCallback_Variable	Variable zum Starten oder Stoppen des Druck-Callbacks
DruckT	Zeitdaten des Druckes
dt	Zeitintervall zwischen den Reglerschritten
dt0	t0 für das Zeitintervall welches der Regler benötigt
DV	Eingegebenes Diafiltrationsvolumen
e	Absolute Regelabweichung
e_rel	Relative Regelabweichung
e_rel_last	Letzte relative Regelabweichung
endpoint_Concentration_value	Endpunkt der ersten Konzentrierung
fiberRadius_cm	Faserradius in cm
FillUpStartVolumen	Gewicht der Feedwaage vor dem Fill Up
FillUpStopVolumen	Gewicht der Feedwaage nach dem Fill Up
filterTimeConstant	Zeitkonstante des Filters der Feedwaagen Signale
flowrate_istWert	Abgefragter Ist-Wert der umgerechneten Flussrate der Feedpumpe
gesetzteFlowrate_in_mlpermin	Flussrate des Operators in ml/min

Name	Beschreibung
Gewicht_gefiltert	Gefiltertes Feedgewicht
Gewicht_gefiltert_last	Letztes gefiltertes Feedgewicht
GewichtFeedWaage	Daten der Feedwaage
GewichtFeedWaageHilfe	Hilfsvariable die in dem Funktion-Callback das Feedgewicht als Double Variable speichert
GewichtPermeatWaage	Daten der Permeatwaage
GewichtPermeat-Waage_cpy_last	Letzter Permeatgewichts-Wert
GewichtPermeatWaage-Hilfe	Hilfsvariable die in dem Funktion-Callback das Permeatgewicht als Double Variable speichert
gewichtSollwert	Sollwert des Feedgewichts für den Regler
I_part	Integral Part der Reglergleichung
k	Variable, welche einmaligen Aufruf von If-Funktion erlaubt, bei welcher der Feedgewicht-Ist-Wert als letzten Gefilterten Wert gesetzt wird
Ki	Intergrationsfaktor für Stellgrößenberechnung
ki	Ki welches der Operator im GUI eingibt
kp	Kp welches der Operator im GUI eingibt
Kp	Proportionalfaktor für Stellgrößenberechnung
max_error	Normierungswert der Regler Parameter (maximal möglicher Fehler)
operator_Name	Name des Operators
P_part	Porportional Part der Reglergleichung
P1	Daten des Feed-Druckes
P2	Daten des Retentat-Druckes
P3	Daten des Permeat-Druckes
pausenzeit	Zeit der Pausen in min
permeat_FLux_gefiltert	Permeat Flux-Wert gefiltert
permeat_FLux_gefiltert_last	Letzter gefilterte Permeat Flux-Wert
permeateFlowrate	Permeat Massenstrom in l/h
permeateMassflowML-proMIN	Permeat Massenstrom in ml/min
permeatFlux	Permeat Flux-Wert
process_Comment	Kommentar zum Prozess
process_Name	Name des Prozesses
Process_status	Statusvariable die anzeigt, ob der Prozess gestartet wurde

Name	Beschreibung
pumpCali_AchsenAbschnitt	y-Achsenabschnitt der linearen Pumpenregression
pumpCali_steigung	Steigungskonstante der linearen Pumpenregression
pumpe_DrehRichtung	Drehrichtung der Pumpe
pumpRate_Anfangspunkt	Anfangspunkt der Pumpe in RPM
pumprate_IstWert	Abgefragter Ist-Wert der Pumpleistung der Feedpumpe
PumpRate_mlpermin	Pumprate der Pumpe in ml/min
PumpRate_prozent	Pumprate der Pumpe in RPM nach Umrechnung von Operator Eingabe
PumpRate_prozentForCali	Pumprate der Pumpe in RPM für Kalibriergerade
pumpRate_SollWert	Relativer Sollwert der Pumpe nach Stellgrößenberechnung und Verrechnung mit dem Arbeitspunkt
pumpRate_SollWert_send	Absoluter Sollwert der Pumpe nach Stellgrößenberechnung und Verrechnung mit dem Arbeitspunkt
R2	Bestimmtheitsmaß der Regression
ramp_up	Variable damit nur einmal im Timer die Flussrate auf den Sollwert gesetzt wird
RPM_max	Maximale RPM der Diafiltrations Pumpe zur Normierung
secondConc_startZeit	Zeit an dem die zweite Konzentrierung gestartet wird
SeriellObjekt_diaPumpe	Serielle Schnittstelle der Diafiltrationspumpe
SeriellObjektdruckMonitor	Serielle Schnittstelle des Druckmonitors
SeriellObjekt_FeedWaage	Serielle Schnittstelle der Feedwaage
SeriellObjektPermeatWaage	Serielle Schnittstelle der Permeatwaage
SeriellObjektPumpe	Serielle Schnittstelle der Pumpe
shearRate	Scherkraft
speed	Variable für Ramp up der Flussrate zum Start
startZeitpunkt	Startzeitpunkt des Prozesses (Start Button)
summe_e	Summe Integral für I Part
t	Objekt des Daten-Timers
t_feedWaage	Zeitvektor der Feedwaage
t_permeatWaage	Zeitvektor der Permeatwaage

<b>Name</b>	<b>Beschreibung</b>
t_permeat- Waage_cpy_last	Letzter Permeatzeit-Wert
t_pumpe	Zeitvektor der Feedpumpe
Tarier_status	Variable welche 1 wird, wenn automatisch tariert wurde
TMP	Daten des Transmembrandruckes
trendData	Feste Matrix für Diagramme der Live-Werte (alle außer Druck)
trendDataDruck	Feste Matrix Für Diagramme der Druck Werte
trendDataTime	Feste Matrix für die Zeitwerte der Live-Daten (alle außer Druck)
trendDataTimeDruck	Feste Matrix für die Zeitwerte der Druck Daten
y_rel	Relative Stellgröße



## 8.5.2 Kurzanleitung der Computer-Anwendung



### Kurzanleitung

Planung, Durchführung und Überwachung  
von Konzentrations- und Diafiltrationsprozesses  
mit XFLOW

Version 2022-01-20

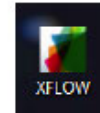
Fabian Weirauch

## Inhaltsverzeichnis

1	Starten des Programmes .....	1
2	Verbindung der Geräte .....	1
3	Kalibrierung der Feedpumpe .....	2
4	Planung eines Filtrationsprozesses .....	3
5	Starten eines Filtrationsprozesses .....	5
6	Überwachung eines Filtrationsprozesses .....	7
7	Manuelles Speichern der Daten aus dem Reiter <i>Tables</i> .....	8

## 1 Starten des Programmes

- 1.1. Geben Sie in der Windows-Suche den Namen des Programmes „XFLOW“ ein und klicken Sie die Anwendung an.
- 1.2. Es kann einen Moment dauern, bis das Programm vollständig gestartet ist.



## 2 Verbindung der Geräte

- 2.1. Klicken Sie auf den Reiter „*Hardware Setup*“
- 2.2. Geben Sie dort die **COM-Ports** der angeschlossenen Geräte in die dazugehörigen Textfelder ein (s. Abbildung).
- 2.3. Klicken Sie auf „*Initialization of Com Ports*“ und warte Sie bis jedes Textfeld einen grünen Haken aufweist und die „Wait!“ Beschriftung zu „Ready!“ wechselt.
- 2.4. Kommt es zu einer Fehlermeldung, befolgen Sie die darauf beschriebenen Anweisungen.

**Hardware Connection**

Com Port Feed Pump:	<input type="text" value="COM1"/>
Com Port Pressure Monitor:	<input type="text"/>
Com Port Feed Scale:	<input type="text"/>
Com Port Permeat Scale:	<input type="text"/>
Com Port Diafiltration Pump:	<input type="text"/>

Ensure that every component has a green check!

Kurzanleitung XFLOW

1

### 3 Kalibrierung der Feedpumpe

3.1. Klicken Sie auf den Reiter „*Calibration*“

3.2. Möchten Sie eine neue Feedpumpenkalibrierung durchführen befolgen Sie die folgenden Schritte.

3.2.1. Verbinden Sie die Feedpumpe auf der einen Seite mit einem Wasser-Reservoir und auf der anderen Seite mit einem Messzylinder.

3.2.2. Geben Sie in dem Zahlenfeld „*Calibration Point*“ (1) jeweils als Erstes die Nummer des Kalibrierpunktes ein.

3.2.3. Im darunterliegenden Auswahl-Menü können Sie entscheiden, ob Sie die Pumpleistung der Feedpumpe in RPM oder Prozent eingeben.

3.2.4. Geben Sie im Zahlenfeld 2 die Pumpleistung ihres derzeitigen Kalibrierpunktes ein.



1: Calibration Point :

Pump Rate in % ▼ 2:

3.2.5. Starten Sie durch Klicken des Buttons „*Start Pumping*“ die Feedpumpe.

3.2.6. Lassen Sie für 60 s Wasser aus dem Reservoir in den Messzylinder fördern und stoppen Sie anschließend durch Klicken des Buttons „*Stop Pumping*“ die Feedpumpe.

3.2.7. Geben Sie im Zahlenfeld 3 das **geförderte Volumen** in ml ein.

3.2.8. Speichern Sie durch Klicken des Buttons „*Save Calibration Point*“ den durchgeführten Kalibrierpunkt.

3.2.9. Wiederholen Sie Schritt 3.2.2. bis 3.2.8 für jeden weiteren Kalibrierpunkt.

3.2.10. Zum Darstellen der Kalibrierkurve und der Regressionsgleichung klicken Sie den Button „*Get Calibration Curve*“



3.3. Möchten Sie die Kalibrierkurve der zuletzt durchgeführten Kalibrierung laden klicken sie den Button „*Load Calibration Curve*“

## 4 Planung eines Filtrationsprozesses

- 4.1. Klicken Sie auf den Reiter „*Process Setup*“ und geben Sie in den obersten 3 Textfeldern den **Prozessnamen**, ihr Operatorkürzel und eine Beschreibung des Prozesses ein.
- 4.2. Scrollen Sie zu dem Bereich „*Create Trial Recipe*“.
- 4.3. Mit Hilfe der Auswahlkästchen können Sie sich für einen **Filtrationsmodus** entscheiden und auswählen ob die Übergänge bei einem Multifiltrationsmodus automatisch oder nach Bestätigung des Endbenutzers erfolgen sollen.
- 4.4. Auf der rechten Seite kann der **Endpunkttyp** (Auswahl-Menü „*End Point Type*“) und der Endpunktwert (Zahlenfeld „*End Point Value*“) der Filtrationsschritte definiert werden.
- 4.5. Geben Sie in dem Zahlenfeld „*Start Volume Feed*“ das **Anfangsvolumen des Feedbehälters** des ausgewählten Filtrationsschrittes an.

1st Concentration	End Point Type Concentration Factor	End Point Value 0	Start volume Feed [ml] 0
1st Dialfiltration	End Point Type Dialfiltration Volume [DV]	End Point Value 0	0
2nd Concentration	End Point Type Concentration Factor	End Point Value 0	

Every field must be filled out!

4.6. In dem darunterliegenden Abschnitt „*Filter Setup*“ können Sie die Daten (Faseranzahl, Faserinnendurchmesser, Filterfläche) ihres Hohlfasermoduls eingeben.

4.7. Im nächsten Abschnitt „*Feed Pump Setup*“ können Sie durch die Auswahlkästchen entscheiden, ob Sie die **Flussrate** oder die **Schergeschwindigkeit** für die Pumpensteuerung vorgeben wollen. Der exakte Wert kann in das Zahlenfeld jeweilige Zahlenfeld eingetragen werden.

4.8. Durch Eingabe eines neuen Wertes und Klicken des Buttons „*Change Flowrate at process*“ können Sie den Wert der Pumpensteuerung im laufenden Prozess ändern.

4.9. Des Weiteren können Sie in diesem Abschnitt die **Drehrichtung der Feedpumpe** jederzeit ändern und den Innendurchmesser des Pumpenschlauches (Zahlenfeld „*Tubing Size*“) in mm eingeben.

<b>Feed Pump Setup</b>	Flow Rate [ml/min]	Shear Rate [1/s]
<input type="radio"/> Set Flowrate	Label	0
<input checked="" type="radio"/> Set Shear	↔	

Every field must be filled out!

[Change Flowrate at process](#)

4.10. Im folgenden Abschnitt „Diafiltration Pump/Controller Setup“ können Einstellungen zu der Füllstandsregelung während einer Diafiltration festgelegt werden.  $K_p$  und  $K_i$  geben dabei die Reglerparameter, das Zahlenfeld „*Time Constant Scale*“ die Zeitkonstante der Filterung der Feedwaagensignale und das Zahlenfeld  $e_{\max}$  den maximal möglichen Regelungsfehler an. Zudem kann der Innendurchmesser des Pumpenschlauches in mm in dem Zahlenfeld „*Tubing Size*“ und der Startwert der Diafiltrationspumpe (Zahlenfeld „*Starting point*“) definiert werden

Starting Point [rpm]	Tubing Size (ID in mm)	max. Pump Rate:	<input type="text" value="400"/> [rpm]		
<input type="text" value="30"/>	<input type="text" value="0"/>	max. possible Error (emax):	<input type="text" value="100"/> [ml]	$K_p$ :	<input type="text" value="565"/>
		Time Constant Scale:	<input type="text" value="5"/> [s]	$K_i$ :	<input type="text" value="300"/>

## 5 Starten eines Filtrationsprozesses

- 5.1. Klicken Sie auf den Reiter „*Process Setup*“ und scrollen Sie zu dem Bereich „*Auto-Save!*“.
- 5.2. Durch Klicken des Buttons „*Search path*“ können Sie einen Pfad für die automatische Messdatendokumentation auswählen.
- 5.3. Scrollen Sie hoch zu dem Abschnitt „*Plant Preparation*“.
- 5.4. Versichern Sie sich, dass Ihre Anlage bereit für den Start des Filtrationsprozesses ist (zu filtrierende Flüssigkeit im Feedbehälter, retentatseitige Schlauchklemme vollständig auf, permeatseitige Schlauchklemme vollständig geschlossen)
- 5.5. Möchten Sie ein **internes Trieren der Feedwaage** (1) durchführen müssen Sie das Startvolumen im Feedbehälter in das Zahlenfeld „*Start volume Feed*“ in ml eingeben und anschließend den Button „*Auto Tare*“ klicken.

1. Belated Taring:

Start volume Feed [ml]

Offset Value

- [g]

5.6. Anschließend folgt das **Fill Up der Anlage** für die Berechnung des Totvolumens. Geben Sie hierfür zunächst einen niedrigen RPM-Wert in das Zahlenfeld „*Pump Rate*“ ein und starten Sie durch Klicken des Buttons „**Start Fill Up**“ die Feedpumpe. Durch Eingabe eines höheren RMP-Wertes und Bestätigen durch Enter wird die Pumpleistung der Feedpumpe automatisch angepasst. Bleibt das angezeigte Gewicht der Feedwaage konstant kann durch Klicken des Buttons „**Stop Fill Up**“ die Feedpumpe gestoppt und das Totvolumen berechnet und angezeigt werden.

5.7. Das manuell ermittelte **permeatseitige Totvolumen** kann im nebenstehenden Zahlenfeld „*Dead volume permeate site*“ in ml definiert werden.

2. System Fill Up:

Pump Rate [RPM]:

- [ml]

Dead volume permeate site  [ml]

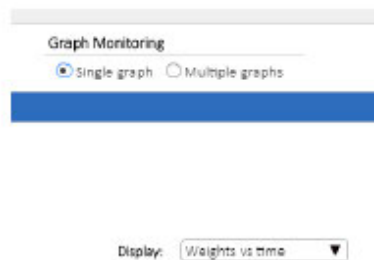
5.8. Gehen Sie in den Reiter „**Process Overview**“ und Klicken Sie zum Starten des Filtrationsprozesses den Button „**START**“.

Process Overview	Tables	Graphs	Save Data
<input type="button" value="START"/>	<input type="button" value="PAUSE"/>	<input type="button" value="STOP"/>	



## 6 Überwachung eines Filtrationsprozesses

- 6.1. Im Reiter „*Process Overview*“ ist eine schematische Darstellung der Filtrationsanlage abgebildet. Hier sind die einzelnen Komponenten der Anlage und prozessrelevanten Daten dargestellt. Ebenso kann diesem Reiter die aktuelle Prozesszeit entnommen werden. Die Buttons „*Tare*“ ermöglichen ein **Tarieren** der Feed- oder Permeatwaage.
- 6.2. Im Reiter „*Tables*“ werden mit einer Abtastrate von 3 s sämtliche Messdaten des Prozesses aufgelistet.
- 6.3. Der Reiter „*Graphs*“ ermöglicht eine graphische Überwachung einiger Messdaten des Filtrationsprozesses. Mit Hilfe der Auswahlkästchen „*Graph Monitoring*“ können Sie zwischen der Anzeige mehrerer kleiner Graphen oder eines großen Graphens entscheiden. Bei Letzterem entscheiden Sie über das Auswahl-Menü „*Display*“ welcher Parameter angezeigt werden soll.



- 6.4. Im oberen Bereich der genannten Reiter können Sie durch Klicken des Buttons „*PAUSE*“ den Filtrationsprozess unterbrechen und durch Klicken des Buttons „*STOP*“ ihn beenden.



## 7 Manuelles Speichern der Daten aus dem Reiter *Tables*

7.1. Klicken Sie auf den Reiter „*Save Data*“ und klicken Sie den Button „*Save Table Data*“.

Es erscheint ein Fenster, indem sie den Pfad für das manuelle Speichern des Datensatzes aus der Überwachungstabelle auswählen können.

To save the data from the Tables Tab:



Save Table Data

(Name is generated automatically)

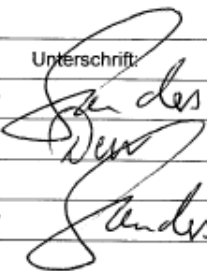
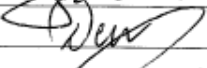
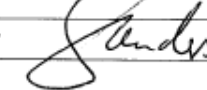
## 8.6 Standardarbeitsanweisungen

### 8.6.1 OD<sub>600nm</sub> - Messung

#### Standardarbeitsanweisung

<b>Bestimmung der Optischen Dichte bei 600 nm</b>	SOP-Nr.: 320101-04
	Gültig ab: 01.07.2013
	Seite: 1 von 4

Standardarbeitsanweisung Nr.:	<b>320101-04</b>		
Titel:	<b>Bestimmung der Optischen Dichte bei 600 nm</b>		
Gültig ab:	01.07.2013		
Geltungsbereich:	Unternehmen:	Hochschule für Angewandte Wissenschaften	
	Abteilung:	Fakultät Life Sciences	
	Standort:	Hamburg / Campus Bergedorf	
	Gebäude:	Laborgebäude Biotechnologie	
	Abteilung:	Bioverfahrenstechnik	
	Raum:	- entfällt -	
Betroffene Produkte:	- entfällt -		
Änderungsverwaltung:	Änderungsfassung		

	Name:	Abteilung:	Datum:	Unterschrift:
Verfasst:	Prof. Dr. Ernst A. Sanders	Laborleiter BVT	29.05.13	
Geprüft:	Dipl.-Ing. Petra Derr	BVT	29.05.13	
Mitzeichnung:	entfällt			
Freigegeben:	Prof. Dr. Ernst A. Sanders	Laborleiter BVT	29.05.13	

Schlüsselworte:

Vergebene Schlüsselworte:	Offline Analytik, OD-Messung
---------------------------	------------------------------

## Standardarbeitsanweisung

<b>Bestimmung der Optischen Dichte bei 600 nm</b>	<b>SOP-Nr.:</b> 320101-04
	<b>Gültig ab:</b> 01.07.2013
	<b>Seite:</b> 2 von 4

Änderungsverwaltung:

SOP ersetzt SOP Nr. / vom:	320101-01 vom 01.04.2008
Grund der Änderung:	Das Pipettieren kleiner Volumina bei hohen Zelldichten führt zu nicht reproduzierbaren Ergebnissen
Änderungen:	Es wurde ein dritter Verdünnungsschritt für höhere Zelldichten eingefügt, so dass die zu pipettierenden Volumina größer sind

SOP ersetzt SOP Nr. / vom:	320101-02 vom 09.11.2011
Grund der Änderung:	Bezeichnung $OD_0$ führte zu Verwechslungen mit dem Wert zu Versuchsbeginn ( $t = 0$ )
Änderungen:	Variable in $OD_{med}$ geändert sowie kleine redaktionelle Änderungen

SOP ersetzt SOP Nr. / vom:	320101-03 vom 15.02.2012
Grund der Änderung:	Redaktionelle Änderungen
Änderungen:	Bezeichnung Standardarbeitsanweisung

32010104 ODMessung 600nm.docx

## Standardarbeitsanweisung

<b>Bestimmung der Optischen Dichte bei 600 nm</b>	<b>SOP-Nr.:</b> 320101-04
	<b>Gültig ab:</b> 01.07.2013
	<b>Seite:</b> 3 von 4

### 1 Einleitung und Zielsetzung

Diese Anweisung beschreibt das Vorgehen bei der Bestimmung der Optischen Dichte  $\Delta OD$ . Wegen der Geräteabhängigkeit muss das verwendete Photometer dokumentiert werden und darf innerhalb einer Versuchsserie nicht gewechselt werden. Die Verdünnungsstufen sind so gewählt, dass der ablesbare Extinktionswert zwischen 0,12 und 0,6 liegt, um eine Proportionalität zwischen  $\Delta OD$  und der Biotrockenmassekonzentration zu erhalten.

### 2 Abkürzungen und Definitionen

$OD_{susp}$	Optische Dichte einer Suspensionsprobe (Extinktionsmessung am Photometer)
$OD_{med}$	Optische Dichte einer Medienprobe
$\Delta OD$	Differenz der Optischen Dichten von Suspension und zellfreiem Überstand
PK	Physiologische Kochsalzlösung, 0,9% w/v NaCl in vollentsalztem Wasser

### 3 Material und Geräte

- Photometer mit Einstellmöglichkeit oder Filter für die vorgegebene Wellenlänge
- Kolbenhubpipette 10 - 100  $\mu$ l
- Kolbenhubpipette 100 (200) – 1.000  $\mu$ l
- Zentrifuge für Mikroreaktionsgefäße, Drehzahl  $\geq 5.000 \text{ min}^{-1}$
- Halbmikroküvetten
- Mikroreaktionsgefäße mit ca. 1,5 ml
- Parafilm
- Physiologische Kochsalzlösung (0,9% w/v NaCl in vollentsalztem Wasser)

### 4 Durchführung

#### 4.1 Vorbereitung

Einschalten des Photometers und bei 600 nm ohne Küvette im Strahlengang auf Null abgleichen (Set Ref).

#### 4.2 Durchführung

1 ml unverdünnte Probe in eine Halbmikroküvette pipettieren. Nullabgleich wiederholen, wenn bei leerem Strahlengang nicht 0 angezeigt wird. Optische Dichte der Suspension messen. Ist  $OD_{susp} > 0,6$  muss verdünnt (siehe 4.3) und erneut gemessen werden.

Nach der Messung wird die Probe oder, wenn verdünnt wurde, die verdünnte Suspension in einem Mikroreaktionsgefäß  $5 \pm 2 \text{ min}$  bei  $\geq 5.000 \text{ min}^{-1}$  zentrifugiert. Der Überstand wird sehr vorsichtig (es dürfen mit dem Auge keine Schlieren erkennbar sein) in eine zweite Küvette dekantiert oder mit einer Pipette übertragen. Im selben Photometer wird bei unveränderten Einstellungen die optische Dichte des Überstands  $OD_{med}$  bestimmt.

## Standardarbeitsanweisung

<b>Bestimmung der Optischen Dichte bei 600 nm</b>	<b>SOP-Nr.:</b> 320101-04
	<b>Gültig ab:</b> 01.07.2013
	<b>Seite:</b> 4 von 4

### 4.3 Verdünnung

Die Verdünnungsstufen lassen sich mit zwei variablen Pipetten ohne Änderung der Volumeneinstellung mit physiologischer Kochsalzlösung herstellen. Erst wenn wegen Änderung der Zelldichte eine andere Verdünnungsstufe zu wählen ist, werden die Volumina an den Pipetten neu eingestellt.

$\Delta OD$	Verdünnung	F	Pipette 1 Probe $\mu\text{l}$	Pipette 2 PK $\mu\text{l}$	Pipette 1 2. Verd. $\mu\text{l}$	Pipette 2 PK $\mu\text{l}$	Pipette 1 3. Verd. $\mu\text{l}$	Pipette 2 PK $\mu\text{l}$
0 bis 0,6	keine	1	1000	-	-	-	-	-
0,6 bis 3	1 : 5	5	200	800	-	-	-	-
3 bis 15	1 : 25	25	40	960	-	-	-	-
15 bis 75	1 : 123	123	90	910	90	910		
75 bis 300	1 : 500	500	200	800	100	900	100	900
> 300	1:1.000	1.000	100	900	100	900	100	900

Die Verdünnungen 1:5 und 1:25 erfolgen direkt in der Küvette, die mit Parafilm abgedeckt über einem Abfallbehälter (Becherglas, Abfallbeutel o. ä.) mehrfach zur Durchmischung umgeschwenkt wird. Bei den Verdünnungen 1:123, 1:500 und 1:1.000 erfolgen der/die ersten Verdünnungsschritte in einem Mikroreaktionsgefäß, der letzte direkt in der Küvette.

### 4.4 Berechnung von $\Delta OD$

Die Differenz der Optischen Dichte der Probe und des Überstands wird mit dem Verdünnungsfaktor  $F$  multipliziert.  $OD_{\text{susp}}$  und  $OD_{\text{med}}$  stehen für die Extinktionswerte der verdünnten oder unverdünnten Proben, wobei beide der gleichen Verdünnungsstufe entstammen.

$$\Delta OD = F(OD_{\text{susp}} - OD_{\text{med}}) \quad (1)$$

## 5 Mitgeltende Unterlagen

- Bedienungsanleitungen der Photometer -

## 6 Anlagen

- keine -

## 8.6.2 BTM-Bestimmung

### Standard Arbeitsanweisung

Gravimetrische Bestimmung der Biotrockenmasse- konzentration in Mikroreaktionsgefäßen	SOP-Nr.: 320001-03
	Gültig ab: 23.04.2013
	Seite: 1 von 4

Standard Arbeitsanweisung Nr.:	<b>320001-03</b>		
Titel:	<b>Gravimetrische Bestimmung der Bio- trockenmassekonzentration in Mikroreaktionsgefäßen</b>		
Gültig ab:	23.04.2013		
Geltungsbereich:	Unternehmen:	Hochschule für Angewandte Wissenschaften	
	Abteilung:	Fakultät Life Sciences	
	Standort:	Hamburg / Campus Bergedorf	
	Gebäude:	Laborgebäude Biotechnologie	
	Abteilung:	Bioverfahrenstechnik	
Raum:	- entfällt -		
Betroffene Produkte:	- entfällt -		
Änderungsverwaltung:	Änderungsfassung		

	Name:	Abteilung:	Datum:	Unterschrift:
Verfasst:	Prof. Dr. Ernst A. Sanders	Laborleiter BVT	16.04.2013	
Geprüft:	Diana Pohle	WiMi BVT	22.04.2013	
Mitzeichnung:	entfällt			
Freigegeben:	Prof. Dr. Ernst A. Sanders	Laborleiter BVT	22.04.2013	

### Schlüsselworte:

Vergebene Schlüsselworte:	Offline Analytik, Biotrockenmassekonzentration, BTM, CDM
---------------------------	--

Ausgabe und Verteilung erfolgt durch den Laborleiter. Handschriftliche Änderungen in der SOP sind nicht zulässig.  
 32000103 BTM Konzentration mit Eppis

### Standard Arbeitsanweisung

Gravimetrische Bestimmung der Biotrockenmasse- konzentration in Mikroreaktionsgefäßen	SOP-Nr.:	320001-03
	Gültig ab:	23.04.2013
	Seite:	2 von 4

#### Änderungsverwaltung:

SOP ersetzt SOP Nr. / vom:	320001-01, die nicht gültig wurde.
Grund der Änderung:	Prüfung auf Notwendigkeit eines Waschschriffs
Änderungen:	Waschschritt gelöscht

SOP ersetzt SOP Nr. / vom:	320001-02 vom 01.12.2011
Grund der Änderung:	Redaktionelle Änderungen und Einführen neuer Bezeichnungen für Variablen
Änderungen:	Angabe von Berechnungsgleichungen

Ausgabe und Verteilung erfolgt durch den Laborleiter. Handschriftliche Änderungen in der SOP sind nicht zulässig.

32000103 BTM Konzentration mit Eppis



## Standard Arbeitsanweisung

Gravimetrische Bestimmung der Biotrockenmasse- konzentration in Mikroreaktionsgefäßen	SOP-Nr.:	320001-03
	Gültig ab:	23.04.2013
	Seite:	3 von 4

### 1 Einleitung und Zielsetzung

Die SOP beschreibt die Bestimmung der Biotrockenmassekonzentration durch Auswiegen der in 1 ml Suspension enthaltenen Biomasse. Verwendet werden Mikroreaktionsgefäße.

### 2 Abkürzungen und Definitionen

BTM Biotrockenmasse

### 3 Material und Geräte

#### 3.1 Material

- 1,5 ml Mikroreaktionsgefäße
- temperaturbeständiger Träger für Mikroreaktionsgefäße
- 1000 µl Pipettenspitzen
- vorbereitete Tabelle

#### 3.2 Geräte

- Trockenschrank
- Exsikkator
- Pinzette
- Analysenwaage
- Kolbenhubpipette 1000 µl
- Zentrifuge für Mikroreaktionsgefäße, Drehzahl  $\geq 10000 \text{ min}^{-1}$

### 4 Durchführung

#### 4.1 Vorbereitung

Die Zahl der für den kompletten Prozess gewünschten Proben wird definiert und eine entsprechende Anzahl an Mikroreaktionsgefäßen in einem Träger mit Aufkleber (Experimentator, Datum, Mikroorganismus) bereitgestellt. Die Mikroreaktionsgefäße werden auf dem Deckel nummeriert.

Wird zu jeder OD-Messung eine BTM-Bestimmung durchgeführt, sollen die gleichen Probennummern verwendet werden.

Da die Massen der Zellpellets sehr klein sind, muss ausgeschlossen sein, dass an den Mikroreaktionsgefäßen befindliche Feuchtigkeit mitgewogen wird. Die beschrifteten, geöffneten Mikroreaktionsgefäße werden dazu im Trockenschrank bei 105 °C über Nacht getrocknet. Es muss darauf geachtet werden, dass keine Feuchtigkeit aus anderen Proben im Trockenschrank die Feuchte erhöht, ggf. mehrfach die Tür kurz öffnen und für Luftaustausch sorgen.

Die in einem Exsikkator abgekühlten, trockenen Mikroreaktionsgefäße werden darin zur Analysenwaage gebracht und ausgewogen.

Ausgabe und Verteilung erfolgt durch den Laborleiter. Handschriftliche Änderungen in der SOP sind nicht zulässig.

32000103 BTM Konzentration mit Eppis

## Standard Arbeitsanweisung

Gravimetrische Bestimmung der Biotrockenmasse- konzentration in Mikroreaktionsgefäßen	SOP-Nr.: 320001-03
	Gültig ab: 23.04.2013
	Seite: 4 von 4

An der Waage ist eine Tabelle etwa folgenden Formats zu nutzen:

Proben Nr.	Datum, Zeit o. Kennzeichng. dd.mm.yy hh:mm	Prozesszeit o. Kennz. h	eingesetztes Volumen ml	Tara- masse, $m_{R,0}$ g	Brutto- masse, $m_{R,X}$ g	Zellmassen- konz., $c_X$ g l <sup>-1</sup>
xx						
:						

Die Mikroreaktionsgefäße werden einzeln mit einer Pinzette aus dem Exsikkator genommen und auf die Waagschale gelegt. Der Exsikkator und die Analysenwaage werden wieder geschlossen. Bei stabiler Anzeige wird abgelesen und der Wert in die Tabelle übernommen.

Nach der Wägung werden die Mikroreaktionsgefäße in aufsteigender Reihenfolge wieder in einen Träger gestellt und dort platziert, wo die Proben später eingefüllt werden.

### 4.2 Durchführung

Zum Befüllen der Mikroreaktionsgefäße eine 1000 µl-Pipette, Pipettenspitzen sowie ein Abfallgefäß bereitstellen.

Von der gut gemischten Probe werden 1000 µl mit der Pipette abgenommen und in das entsprechende Mikroreaktionsgefäß gegeben. Dieses wird verschlossen und in der Zentrifuge mit einem Gegengewicht (zweite Probe oder ein mit gleicher Menge Wasser gefülltes Mikroreaktionsgefäß) 3 min bei  $\geq 10000 \text{ min}^{-1}$  zentrifugiert. Nach dem Dekantieren des Überstands wird die restliche Flüssigkeit, die sich auf dem Zellpellet befindet, vorsichtig mit einer Pipette abgesaugt, ohne das Pellet zu berühren. Das Mikroreaktionsgefäß wird geschlossen in den Träger zurückgestellt.

### 4.3 Trocknung

Zur Trocknung werden die Gefäße geöffnet und in einem temperaturbeständigen Träger in den Trockenschrank bei 105 °C gestellt, so dass die Feuchtigkeit entweichen kann. Es muss darauf geachtet werden, dass keine Feuchtigkeit aus anderen Proben im Trockenschrank die Feuchte erhöht, ggf. mehrfach die Tür kurz öffnen und für Luftaustausch sorgen.

### 4.4 Auswertung

Nach Abkühlen der trockenen Mikroreaktionsgefäße mit den Pellets im Exsikkator werden diese mit einer Pinzette entnommen und auf der Analysenwaage gewogen. Wenn die Anzeige einen stabilen Wert anzeigt, wird dieser in die Tabelle eingetragen.

Die Zellmassenkonzentration ergibt sich aus

$$c_X = \frac{m_{R,X} - m_{R,0}}{V_L}$$

## 5 Mitgeltende Unterlagen

- keine -

## 6 Anlagen

- keine -

Ausgabe und Verteilung erfolgt durch den Laborleiter. Handschriftliche Änderungen in der SOP sind nicht zulässig.

32000103 BTM Konzentration mit Eppis

## 8.7 Erklärung zu verwendeten Hilfsmitteln

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig verfasst und dabei keine anderen als die angegebenen Hilfsmittel benutzt habe. Sämtliche Stellen der Arbeit, die im Wortlaut oder dem Sinn nach Publikationen oder Vorträgen anderer Autoren entnommen sind, habe ich als solche kenntlich gemacht. Ich bin damit einverstanden, dass die Masterarbeit veröffentlicht wird.

---

Ort, Datum Unterschrift

---

Unterschrift: Fabian Weirauch