NTNU
Norges teknisk-naturvitenskapelige universitet

Fakultet for naturvitenskap og teknologi
Institutt for kjemisk prosessteknologi

SPECIALIZATION PROJECT 2013

TKP 4550

PROJECT TITLE:

Models for on-line control of batch polymerization processes

……

By
Fredrik Gjertsen

……

Supervisor for the project: Sigurd Skogestad, NTNU
Peter Singstad, Cybernetica AS

Date: December 8th, 2013

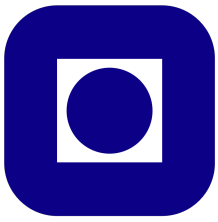
TKP4550 - Specialization project, Process Systems Engineering

Models for on-line control of batch polymerization processes

State and parameter estimation for semi-batch free-radical emulsion copolymerization processes

Author: Fredrik Gjertsen
Supervisors: Prof. Sigurd Skogestad, NTNU
Peter Singstad, Cybernetica AS

Trondheim, December 2013



Department of Chemical Engineering
Norwegian University of Science and Technology

Abstract

This report presents work done as part of the compulsory specialization project in the final year of the study program for the degree of M.Sc. in Chemical Engineering at the Norwegian University of Science and Technology. The project work has been carried out in close collaboration with Cybernetica AS, who proposed the project. The project serves as an extension to a summer internship with Cybernetica AS, and the work is proposed to be extended for a masters thesis during the spring of 2014.

The purpose of the project work is to perform parameter estimation and model validation on an already established model for semi-batch free-radical emulsion copolymerization. The aim is to improve the quality of this model further. Using experimental data from industrial scale reactors or lab-scale test reactors, the established model can be improved to fit reality in a better way. This is important, because the established reactor models are constructed mainly from first principles. Optimal parameter fitting was done using the Cybernetica ModelFit software, which is introduced in this text.

The report aims to give a brief introduction to the established model, which is formulated in the Modelica programming language. The report also includes fundamental theoretical considerations with respect to both emulsion copolymerization and semi-batch reactor modeling, thus providing a sensible base for the main purpose, which is off-line parameter estimation and model validation. In addition to this, an effort has been made to establish the theoretical background for on-line estimation methods, which will be a key feature of an on-line controller implementation in providing methods for on-line state and parameter estimation. The ultimate goal is to design a complete controller for a emulsion copolymerization process, using nonlinear model-based predictive control (NMPC), and this raises the need for a model of high-end quality. The NMPC controller design itself is left for future work, e.g. a masters thesis, while the scope of this work is focused on obtaining a valid model.

I would like to express my sincere gratitude to the employees of Cybernetica AS for providing valuable support along the entire duration of my internship and project work, and for making my stay both pleasant and interesting. I would also like to thank my supervisor at NTNU, Professor Sigurd Skogestad, for his support to the project work.

This work is carried out as a side-track of the COOPOL (Control and Real-time Optimization of Intensive Polymerization Processes) project, which is an EU collaborative research project[1] involving several research institutions and industrial partners across Europe, and I acknowledge the valuable help and guidance given by my fellow contributors of the COOPOL project.

Fredrik Gjertsen
Trondheim, December 8, 2013

[1] Read more about the COOPOL project here: `http://www.coopol.eu/`

Contents

List of Figures

List of Tables

List of Abbreviations

CSTR	Continuously Stirred Tank Reactor
CTA	Chain Transfer Agent
CVs	Controlled variables (sometimes referred to as system outputs)
DAE	Differential Algebraic Equation
DFO	Differentiation-free Optimization
DVs	Disturbance variables
EKF	Extended Kalman Filter
FMI	Functional Mock-up Interface
GUI	Graphical User Interface
HEKF	Hybrid Extended Kalman Filter
HSE	Health, safety and environment
IEKF	Iterated Extended Kalman Filter
KF	Kalman Filter
LKF	Linearized Kalman Filter
LSM	Line-search Methods
LTI	Linear Time Independent (controller)
M1	Monomer type 1
M2	Monomer type 2
MHE	Moving Horizon Estimator
MPC	Model-based Predictive Control
MVs	Manipulated variables (sometimes referred to as system inputs)
NMPC	Nonlinear Model-based Predictive Control
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
PDI	Polydispersity Index
PFR	Plug Flow Reactor
PVC	Poly Vinyl Chloride
QP	Quadratic Programming
SBR	Styrene Butadiene Rubber
SQP	Sequential Quadratic Programming
TRM	Trust Region Methods
UKF	Unscented Kalman Filter
VCM	Vinyl Chloride Monomer

List of Symbols

Roman letters:

A	System matrix for linear continuous systems, relating states with state changes
A	Matrix used in the equation system for full population balance for radical species modeling
B	System matrix for linear continuous systems, relating inputs with state changes
C	System matrix for linear continuous systems, relating states with measurements
C	Termination term for modeling free-radical species
$c_{p,i}$	Specific heat capacity of component i
c_i	Concentration of component i (usually molar basis)
D	Dead polymer chain (deactivated endgroup)
E	Energy content of a system
$E[\cdots]$	Statistical expectation value of a variable
$E_{A,i}$	Activation energy for species i
F_k	System matrix for linear discrete systems, relating states with state changes
f	General, arbitrary multivariable function
f_I	Efficiency factor of free-radical initiation
G_k	System matrix for linear discrete systems, relating inputs with state changes
g	Gravitational constant
g	General, arbitrary, multivariable function
h	Specific enthalpy
h	General, arbitrary multivariable function
H	Enthalpy for a system
H_k	System matrix for linear discrete systems, relating states with measurements
J	Moment of inertia for pendulum example model
J_k	Cost function for optimal estimation at time t_k
k	Desorption-term for modeling free-radical species
k_I	Rate constant for free-radical initiation
k_{ij}	Propagation rate constant for adding monomer type j to endgroup of type i
k_{ijk}	Propagation rate constant for adding monomer type k to endgroup of type j having a penultimate unit of type i
$k_{f,ij}$	Rate constant for chain transfer between a chain having an endgroup of type i and monomer type j
$k_{f,CTA,i}$	Rate constant for chain transfer from chain endgroup type i to CTA
k_{tc}	Rate constant for polymer chain termination by chain recombination
k_{td}	Rate constant for polymer chain termination by chain disproportiation
k_T	Combinated rate constant for polymer chain termination
$k_{reac,0}$	Initial rate constant for reaction $reac$
$k_{reac,adj}$	Adjustment factor for the rate constant of reaction $reac$
kA	Heat transfer coefficient between reactor and cooling jacket
kA_{SR}	Heat transfer coefficient between surrounding environment and reactor
kA_{SJ}	Heat transfer coefficient between surrounding environment and cooling jacket
$k_{i,p1\rightarrow p2}$	Mass transfer coefficient for component i between phases $p1$ and $p2$
K	Estimator gain, continuous formulation
K_k	Estimator gain at time t_k

M_i	Monomer, type i
L	Pendulum string length for pendulum example model
M_{Mi}	Molar mass, monomer type i
M_n	Molar mass, number average
M_w	Molar mass, weight average
m	Total mass amount
m	Pendulum mass for pendulum example model
m_i	Amount of component i, mass basis
$\dot{m}_i$	Flow rate of component i, mass basis
$\dot{m}_{i,p1\rightarrow p2}$	Mass transfer of component i from phase $p1$ to phase $p2$
n_i	Molar amount of component i
n	Vector of respective molar amounts for particles carrying radicals
$\bar{n}$	Average number of radicals per particle
P	Vector of covariances for a corresponding state vector
$\dot{P}$	Continuous time derivative of covariance vector
P_k^-	Vector of *a priori* covariances for the state estimate at time k
P_k^+	Vector of *a posteriori* covariances for the state estimate at time k
P_i	Living polymer endgroup, type i
p	Pressure of a system
Q_k	Vector of covariances for process noise at time t_k
Q_c	Vector of covariances for process noise, continuous formulation
$\tilde{Q}$	Linearized vector of covariances for process noise
Q	Heat (energy) transfer
q	Extra supporting variable for modelig free-radical species
r_{ij}	Relative reactivity between homopropagation and cross-propagation for endgroup of type i in a copolymerization system
r_i	Reaction rate of component i
R	Universal gas constant
R_k	Vector of covariances for measurement noise at time t_k
R_c	Vector of covariances for measurement noise, continuous formulation
$\tilde{R}$	Linearized vector of covariances for measurement noise
$R(t)$	Molar amount of reactant in a reactor at time t
s	Used to denote length of a polymer chain
T	Temperature
T	Small time interval (Δt) used to derive the continuous linear Kalman filter
t	Time
$Tr\,(A)$	Trace of a matrix A (linear algebra operation)
t_k	Time at point k
u	Vector of inputs for a system
U	Internal energy of a system
v	Measurement noise
v_k	Measurement noise at time t_k
V	Volume
$\dot{V}$	Volumetric flow rate
V_p	Volume of (polymer) particle phase

V_d	Volume of (monomer) droplet phase
V_w	Volume of (aqueous) water phase
w	Process noise
w_k	Process noise at time t_k
w_λ	Process noise affecting weakly varying process parameters
w_i	Mass fraction of component i
W_s	Shaft work applied to a system
X	Degree of conversion
x	Relative monomer composition in system, used in the copolymer equation
x	Vector of states
x_k	Vector of states at time instant t_k
$\dot{x}$	Time derivative of (the vector of) states
$\hat{x}_k^-$	Vector of *a priori* state estimates at time t_k
$\hat{x}_k^+$	Vector of *a posteriori* state estimates at time t_k
$\dot{\hat{x}}$	Time derivative for state estimate vector
y	General, arbitrary, multivariable function
y	Vector of measurements
y_m	Vector of measurements
y_k	Vector of measurements at time t_k
$y_{p,k}$	Vector of predicted outputs (measurements) at time t_k
y	Relative monomer representation in polymer, used in the copolymer equation
Y_i	Molar fraction of monomer type i in polymer

Greek letters:

$\delta(t - t_0)$	Delta impulse function with "spike" at time t_0
Δt_k	Discrete time interval between time t_{k-1} and t_k
ϵ	State estimation error
η	Subset of the vector of parameters (θ) which is unknown
θ	Vector of parameters for a system
μ_{Mi}	Consumption of monomer type i in the copolymerization process
μ_i^j	i'th order moment of component j
ρ	Density
ρ_i	Density of component i
σ	Generation-term for modeling free-radical species
τ	Used as a replacement for time (t) in integrals where time appears both in the integrand and the integration boundaries
ϕ	Extra supporting variable for modelig free-radical species
Φ	Pendulum angle for pendulum example model
ω	Used as a replacement for process noise (w) in integrals where the process noise is "transformed" from a continuous to a discrete formulation
ω	Angular velocity for pendulum example model

1 Introduction

Polymer science and polymer industry are of huge importance for everyday life, in providing the basis for a large variety of products. Examples can range from household products found in the kitchen and furniture, to plastics, rubber for car tires, health care products, agricultural applications, etc. The applications are numerous. Many industrial polymer processes also provide industrial chemicals for use in further processing. To exemplify the importance of polymer industry, the European consumption of plastics in 2011 was approximately 47 million tonnes, which is a 1.1% increase from 2010. A significant portion of this increase is associated with the automotive industry, which experienced an increase of almost 10% in the consumption of plastics. [1]

Emulsion polymerization processes in particular, which are the focus of this work, are used for several different applications worldwide. The most significant ones are probably the industrial production processes leading to adhesives (glues) and various coatings. The preparation and use of adhesives is ancient, perhaps nearly as old as civilization itself. Speaking about modern industrial production, however, many of the industrial technologies were first established close to a hundred years ago, and they are still subject to research and improvements [2]. New ways to achieve desired reactor cooling, new monomers, new combination patterns of monomers in copolymerization, alternative reactor structures, etc. are just a few examples of factors that bring great variety and new possibilities to the science of emulsion (co)polymerization, aswell as polymerization in general. To exemplify the importance of these types of processes, the world consumption of adhesives was approximately 12 million tonnes in 2010, having an average annual growth of 3.5%. The annual growth is predicted to be as high as 5% by 2015. [3]

In the pursuit of the best possible performance of chemical reactors used in polymerization processes, with the ultimate goal being the most profitable operation under safe conditions, there are several challenges. One example of such a challenge is the fact that many polymerization systems are highly sensitive to temperature changes, with the chemical reactions of the system being both exothermic and kinetically favorable at higher temperatures. This can lead to so-called run-away situations, and raises the need for good solutions for temperature control for the reactor system. In a run-away situation the heat developed in the reactor system acts directly to speed up the chemical reaction rate. This may proceed in an uncontrollable manner, quickly establishing a vicious circle leading to plant failure if it is not counteracted. Such a scenario would endanger the employees of the plant, the ability to keep up the production as intended and possibly also the nearby environment. Because of this, temperature control is a crucial issue for most polymer reactors, and the use of sophisticated controllers can, with respect to this concern, contribute to yield safe operation of the plant. Using model-based controllers, critical points in the operation of the reactor can be predicted in advance and be actively treated according to the objectives of the plant.

Another example of a typical challenge in polymerization processes is the time duration associated with making each batch of polymer product in a (semi-)batch reactor. For many industrial applications, the batch time may amount to several hours. Under given constraints for product quality, considerations with respect to health, safety and environment (HSE), etc., the batch time could be subject to optimization, with the goal being a larger production capacity in total as a consequence of shorter batch times. The reduction of batch time is a complex issue demanding extensive knowledge of all the aspects affecting the behavior of the reactor system. A suggestion to overcome the time demand in batch reactor applications is to explore the use of continuous reactors for polymer applications, for instance the use of tubular plug flow reactors (PFR). This is

a less mature technology than batch reactor applications when it comes to polymer industry, and presents with challenges of its own. Flow patterns, mixing issues and cooling systems are among the challenges associated with continuous flow reactors. Common for both batch reactors and continuous flow reactors is that the evaluation and control of these systems require process models of high-end quality and extensive process understanding for the respective systems, and these systems are often complex and time consuming to comprehend, since they are dynamic and theoretically advanced. On the other hand, having an advanced controller is crucial for the success of such a reactor system.

Once a mathematical process model is developed for a system, the potential for model-based control is present. This does, however, require the model to be of high-end quality, and the model must be adjusted to fit reality in a satisfying way, i.e. the respective parameters of the model must be estimated such that the model agrees with experimental data. Parameter estimation for a model is, in other words, a preliminary task with respect to controller design, in the sense that the model must be verified and improved off-line before it is used in on-line applications. The off-line validation of the model could also include estimation of initial states for the system, in the case where certain states are initially unknown or known with low certainty. In addition to this, estimation is also an ongoing continuous task in the controller implementation, in the sense that both states and parameters for the respective system can be estimated on-line by the estimator unit in the controller implementation.

The purpose of this project work is to perform off-line parameter estimation with experimental data to fit some of the physical parameters for a specific reactor system for semi-batch free-radical emulsion copolymerization to experimental data. On-line state estimation and parameter estimation has also been explored, and algorithms for Kalman filter estimators has been derived. For the purpose of this work, the development of the various estimator equations is an entirely theoretical achievement without immediate implications for the results of this specific report. This will, however, provide the basis for further considerations of the system, e.g. a masters thesis, for which the design of a model-based predictive controller is proposed. In a complete controller implementation, the estimation algorithms for the system play a key part alongside the controller algorithm iself, and so does the process model.

2 Theoretical concepts

The scope of this section is to provide a brief introduction to the theoretical concepts involved in the project work. The specific system considered in the project work is a semi-batch reactor case with free-radical emulsion copolymerization. The models involved are mainly constructed using first principles[1], and the fundamental theoretical concepts for performing the modeling are introduced. In Sec. 2.1, the chemistry of emulsion copolymerization systems is described, while Sec. 2.2 considers modeling of semi-batch reactors. Then, in Secs. 2.3 & 2.4, theory of optimization and estimation for dynamic systems is introduced.

Altogether, this will establish the necessary basis for the purpose of this text. In addition, it will provide several important considerations which will prove important for the proposed next step, which is the design and tuning of an on-line nonlinear model-based predictive controller structure including an on-line estimator.

2.1 Fundamentals of free-radical emulsion copolymerization

This section aims to give an introduction to some of the basic theoretical concepts of copolymerization, specifically for emulsion systems, i.e. the typical chemical behavior of such systems. The motivation for this is to establish a proper starting point for discussing the modeling of emulsion copolymerization reactors. The chemical features of the copolymerization reaction system, including possible reaction patterns, are discussed throughout this section. A discussion on the phase behavior of copolymerization systems and the modeling of free-radical species both included, and some quality parameters for polymer products are given at the end of the section. For more details on emulsion (co-)polymerization, the reader is referred to more extensive publications on this subject. [2][4][5]

The course of a free-radical polymerization reaction process can be separated/classified into several different stages. These are most commonly referred to as initiation, propagation and termination. In addition, so-called chain transfer processes will occur, from activated polymer chains to both free monomer units and CTA[2]. In free-radical polymerization, the growing polymer chains are often described as "living" (active) or "dead" (inactive/terminated) depending on whether they carry radicals on their endgroups, and the chemical behavior of the system is largely dependent on this. An analogy to the different stages of the free-radical polymerization process, which helps to support the active/terminated classification of polymer chains, is presented in Fig. 2.1. Here, the behavior of a rapidly growing biological system of cells in a closed-off environment with limited nutrition is introduced, as seen in Fig. 2.1a. Initially, the cell population experiences excess of nutrition, and will multiply in the birth-dominated region until the births of cells balance the deaths of cells in the stationary region. When the source of nutrition is nearly depleted, the deaths will outweigh the births until the cell culture is extinct. The same behavior is found for the active polymer chains in a free-radical polymerization system. In Fig. 2.1b, the total polymer concentration, i.e. concentration of active and terminated chains combined, is shown. The concentration of activated chains is not illustrated, but it would be (qualitatively) similar to the living cell concentration in Fig. 2.1a. In this analogy, the monomers of the polymer system, which act to increase the polymer concentration, are analogous to the nutrition for the cell culture. Once the nutrition (monomer) is

[1] First principles (*ab initio*) in physics refer to calculations starting directly from established laws, largely avoiding assumptions based on empirical evidence and fitted parameters.

[2] CTA: Chain Transfer Agent

drawing near to depletion, the cells (active polymer chains) are more likely to die (deactivate) than to keep on growing. As an ending remark to this analogy, it is emphasized that the deactivation of polymer is not as fatal as the analogous term (death) suggests, given that they have already grown to a satisfying size.

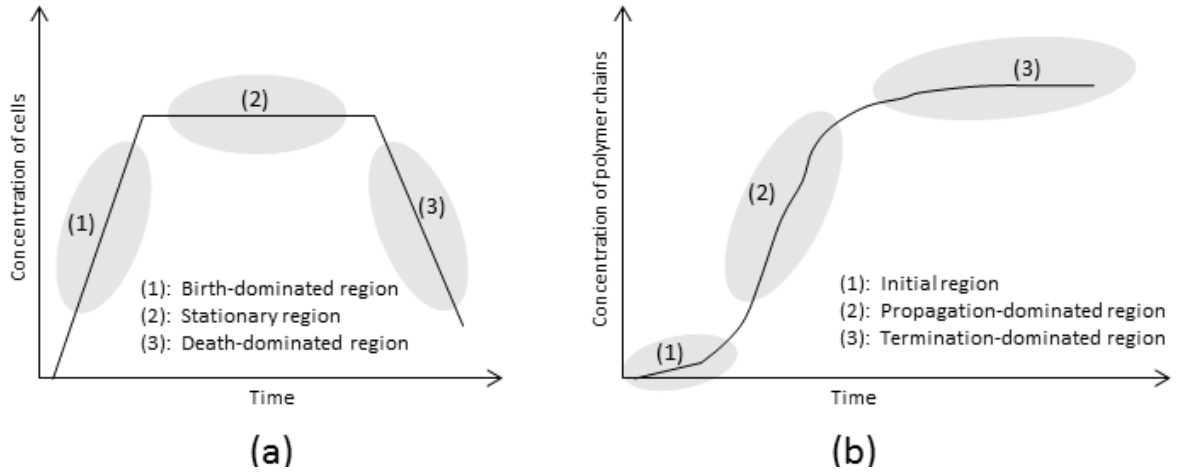


Figure 2.1: Qualitative analogy for free-radical polymerization, using the lifetime of a typical biological cell culture in a closed-off environment (a) to illustrate the "life" of a typical system of growing free-radical polymer chains (b).

Initiation

Initiation of radical species is a crucial step in a free-radical polymerization reaction system. Although the propagation reactions are the reactions that actually create and elongate the polymer chains, these reactions would never proceed in the first place without the activation/initiation of the radical species in the system. The modeling of the radical distribution in the system is itself challenging, and this is discussed briefly later in this section. Various peroxides are most commonly used in free-radical initiation processes, as peroxides easily decompose to form free radicals under the right conditions. In such situations, the peroxide is the *initator* of the system. Fig. 2.2 is an illustration of the initatior activiation for an arbitrary peroxide compound. Once the peroxide is activated it can start to "attack" the double bond(s) of the monomer units. This mechanism[3] is illustrated in Fig. 2.3. Notice that after the attack, the free radical, which originally resided at the oxygen of the peroxide, is actually carried by one of the carbon atoms originally situated in the double bond, enabling this carbon atom to attack a new double bond. In other words, the reaction proceeds towards propagation after this initial activation, and the polymer chain starts to grow in a "snowball"-manner[4]. A polymer chain is generally referred to as living or dead, depending on whether the end group of the chain is activated by a radical species or not. In this sense, termination reactions, which are discussed later, tend to deactivate ("kill") living chains by

[3]In figures showing reaction mechanisms, the thin, curved arrows indicate the migration of electrons or radicals.

[4]Much like a snowball grows rapidly once it starts to roll downhill, a polymer chain grows rapidly once initiated/activated by a radical species. This is crucial to the polymerization process, yet important to control.

neutralizing/removing the free-radical state of the growing polymer chain.

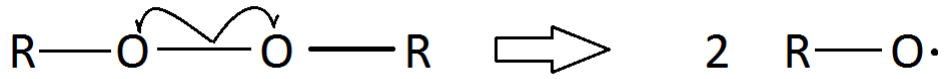


Figure 2.2: Mechanistic idea for initiator activation.

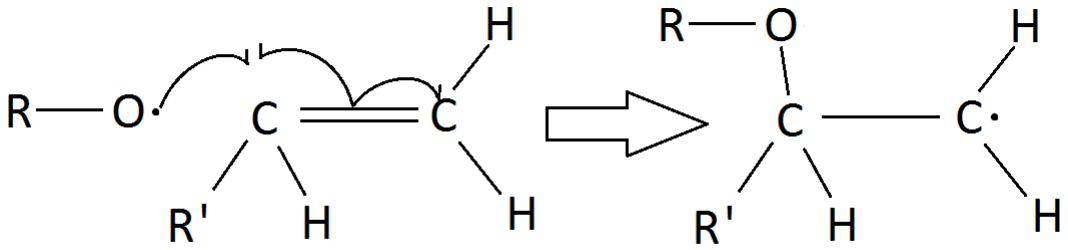


Figure 2.3: Mechanistic idea for initiator attack on a monomer double bond.

The rate of change for the amount of initiator compound is usually expressed as indicated in Eq. 2.1, where both initiator decomposition and external mass tranfer of initiator are considered. In this expression, n_I is the amount of initiator in the reaction vessel and $\dot{n}_I$ is the net external transport of initiator to the system, while f_I is the efficiency factor for the activation process. The kinetic rate constant (k_I) is calculated using a standard temperature dependent Arrhenius-expression starting from an initial value for the rate constant ($k_{I,0}$), as shown in Eq. 2.2. Here, $E_{A,I}$ is the activation energy for the initiator decomposition, while R and T denote the universial gas constant and the system temperature, respectively. For reasons related to parameter estimation, which will be discussed later, a fictitious adjustment factor ($k_{I,adj}$) is also included in this expression to ease parameter modifications in the future.

$$\frac{dn_I}{dt} = -f_I k_I n_I + \dot{n_I} \tag{2.1}$$

$$k_I = \frac{k_{I,0}\ exp\left(-\frac{E_{A,I}}{RT}\right)}{k_{I,adj}} \tag{2.2}$$

Propagation

The propagation reactions represent the connecting of monomer units to growing (living) polymer chains. This is the main part of the copolymerization reaction, and these are the dominant reactions in the propagation-dominated phase (2), as can be seen in Fig. 2.1b. Generally speaking, the system can consist of any number of various monomer types, which will affect the complexity of the system, and also the properties of the polymer product. The most usual cases are, however, one-monomer polymerization (homopolymerization[5]) and copolymerization with two monomers[6]. The latter is

[5]A typical example of homopolymerization: The production of polyvinyl chloride (PVC) uses vinyl chloride monomer (VCM), also known as chloroethene, to yield a very applicable plastic polymer. This polymer is mainly produced using chemical systems for free-radical suspension polymerization.

[6]A typical example of two-monomer copolymerization: The production of styrene-butadiene rubber (SBR) combines styrene monomer and butadiene monomer to yield a polymer product with desirable properties for rubber used

the case for the process studied in this work, and hence this will govern the focus of this theoretical introduction.

The notation used below, which is also used throughout the entire text, represents the various species of the chemical system as follows:

- M_i represents a monomer unit of type i. For a two-monomer copolymerization case, i can be either 1 or 2.
- $M_i(s)$ represents a polymer chain body consisting of a combination of s monomer units, which could be any combination of type 1 and type 2.
- P_i represents a living (active) endgroup of a polymer chain, which originally was a monomer unit of type i. This active endgroup is a key component in the free-radical reactions involved in the chemical reaction system. In the cell culture analogy introduced in Fig. 2.1, the cells are analogous to these living polymer chains with active endgroups.
- D_i represents a dead (deactivated) endgroup of a polymer chain, which originially was a monomer unit of type i.
- k_{ii}, k_{iji}, etc. are reaction rate constants for the various reactions involved in the propagation reactions in the polymerization process. For instance, k_{12} is the kinetic rate constant for the reaction where a monomer type 2 adds to a growing polymer chain having a living endgroup of type 1, k_{212} is the rate constant for the reaction where monomer type 2 adds to a growing polymer chain having an active endgroup of type 1, with a unit of type 2 neighboring the endgroup, and so on.

The monomers of the system can interact in various patterns, depending on the respective monomers reactive affinity for the other monomers in the system. This can be described by different modeling approaches, using varying degree of complexity. The Endgroup Kinetic Model, also known as The Terminal Model, for copolymerization involves four[7] possible propagation reactions, as indicated in Eqs. 2.3 - 2.6. In this sense, the reactivity of the growing polymer chain is determined solely by the characteristics of the terminal group.

$$M_i(s) - P_1 \quad + \quad M_1 \quad \xrightarrow{k_{11}} \quad M_i(s) - M_1 - P_1 \tag{2.3}$$

$$M_i(s) - P_1 \quad + \quad M_2 \quad \xrightarrow{k_{12}} \quad M_i(s) - M_1 - P_2 \tag{2.4}$$

$$M_i(s) - P_2 \quad + \quad M_1 \quad \xrightarrow{k_{21}} \quad M_i(s) - M_2 - P_1 \tag{2.5}$$

$$M_i(s) - P_2 \quad + \quad M_2 \quad \xrightarrow{k_{22}} \quad M_i(s) - M_2 - P_2 \tag{2.6}$$

A more complete, yet more complex, model for the reaction kinetics is known as The Penultimate Model. This approach also recognizes the effect of the penultimate unit in the growing polymer chain (that is, the unit neighboring the unit at the end of the chain). An example is shown in Eqs. 2.7 - 2.8.

$$M_i(s) - M_1 - P_1 \quad + \quad M_1 \quad \xrightarrow{k_{111}} \quad M_i(s) - M_1 - M_1 - P_1 \tag{2.7}$$

$$M_i(s) - M_2 - P_1 \quad + \quad M_1 \quad \xrightarrow{k_{211}} \quad M_i(s) - M_2 - M_1 - P_1 \tag{2.8}$$

$$\text{In this case,} \quad k_{111} \neq k_{121}$$

in car tires, among other applications.

[7] This number is valid for a two-monomer system only, and will be higher in the event of introducing more monomer types, as this will allow for more possible combinations. Combinatorially speaking, the number of possible combinations is the amount of monomer types squared.

Notice that both Eq. 2.7 and Eq. 2.8 in the penultimate model are accounted for as the same reaction in the terminal model, namely Eq. 2.3, although they are not identical. In other words, the penultimate model approach represents reality in a better way, but also introduces more complexity. In many situations, the terminal model is sufficient, hence this model will be applied for the considerations in this project work. The reaction rate constants are usually modeled in a way similiar to the case for the initatior activation (Eq. 2.2), i.e. using an Arrhenius temperature dependency, as indicated in Eq. 2.9. For the propagation reactions, an adjustment factor ($k_{ij,adj}$) is included, like what was done for the initiation. Here, $E_{A,ij}$ is the activation energy for the propagation reaction where monomer type j adds to a growing chain with type 1 active endgroup.

$$k_{ij} = \frac{k_{ij,0}\, exp\left(-\frac{E_{A,ij}}{RT}\right)}{k_{ij,adj}} \quad , \quad i,j \in [1,2] \tag{2.9}$$

With the various possible combinations for chain propagation established, the composition of the reaction mixture aswell as the polymer chains is possible to model. This can provide valuable information about the copolymer system, depending on what is already known for the system. For instance, this information can be used to achieve correct values for cross-propagation rate constants (i.e. k_{12} and k_{21}) from experimental data in a case where these are unknown. The algebraic combination of the various propagation reactions can yield the so-called copolymer equation. The following derivation is based on three important assumptions.

1. The kinetics of each propagation step is first order with respect to the reactants, that is monomer and living/active chains, and the reactions are irreversible. These assumptions enables the establishment of simple kinetic expressions for the rate of change in each species.
2. The reaction kinetics assumes a "pseudo steady state"-situation where the generation and consumption of each radical species balance each other. The mathematical interpretation of this is: $k_{12}[P_1][M_2] = k_{21}[P_2][M_1]$.
3. The reactor in mind is a batch reactor. For reactors having external mass transfer, the resulting expression will look slightly different. Considerations must be taken, for instance, if the equation is to be deployed for a semi-batch reactor system.

Here, brackets indicate concentration. The change in the respective monomer concentrations can be expressed with respect to the proposed propagation reactions.

$$-\frac{d[M_1]}{dt} = k_{11}[P_1][M_1] + k_{21}[P_2][M_1] \tag{2.10}$$

$$-\frac{d[M_2]}{dt} = k_{12}[P_1][M_2] + k_{22}[P_2][M_2] \tag{2.11}$$

$$\Longrightarrow \quad \frac{d[M_1]}{d[M_2]} = \frac{k_{11}[P_1][M_1] + k_{21}[P_2][M_1]}{k_{12}[P_1][M_2] + k_{22}[P_2][M_2]} \tag{2.12}$$

Using the second assumption listed above, and introducing some extra variables, this expression can be rearranged in a comprehensible way for further applications. The introduced variables are the relative change in monomer concentration ($y = \frac{d[M_1]}{d[M_2]}$), the relative monomer concentration ($x = \frac{[M_1]}{[M_2]}$), the relative reactivity ratio between homo-propagation and cross-propagation for active endgroups of type 1 ($r_{12} = \frac{k_{11}}{k_{12}}$), and the relative reactivity ratio between homo-propagation and

cross-propagation for active endgroups of type 2 ($r_{21} = \frac{k_{22}}{k_{21}}$). The final expression, after introducing the new variables, is given in Eq. 2.13, which is known as The Copolymer Equation[8]. In a case where the reaction mixture composition is well monitored, this expression can be used to determine the relative reactivity ratios, for instance using a Fineman-Ross plot. On the other hand, if the rate constants are well known, this equation can be used to adequately express the polymer composition as a function of monomer blend composition.

$$y = \frac{1 + r_{12}x}{1 + \frac{r_{21}}{x}} \tag{2.13}$$

The values of the rate constant ratios (i.e. r_{12} and r_{21}) can also be used to qualitatively predict the polymer composition, seeing as they give the ratios between the rate constants for homo-propagation and cross-propagation. High values for the relative ratios indicate higher affinity for homo-propagation than cross-propagation, and *vice versa* for low values. Some special cases for the relative reactivity ratios are represented below.

- $r_{12} \gg 1$, $r_{21} \gg 1$: In this case, the affinity for homo-propagation totally outweighs the affinity for cross-propagation. For an active endgroup of type 1, it is virtually no desire to let monomer of type 2 add to the chain, and *vice versa*. The result is likely to be two different types of homopolymers. For many copolymer applications, this is not desirable, as it is indeed the combination of monomers that gives rise to wanted copolymer properties.

- $r_{12} > 1$, $r_{21} > 1$: In this case, the active endgroups affinity for homo-propagation is larger than the affinity for cross-propagation. This will promote copolymers with a block structure, with blocks of repeating units arising from the same type of monomer. The average length of the blocks will depend on the actual values for the relative ratios. This situation is illustrated in Fig. 2.4a.

- $r_{12} \approx 1$, $r_{21} \approx 1$: Here, the copolymer has the so-called "completely random" characteristic. This is intuitive, since this situation implies that an active endgroup has the same affinity for both monomer types in the mixture, regardless of whether the endgroup itself is type 1 or type 2. Hence, the general combination of monomers is hard to predict, and the specific monomer ordering becomes random. This situation is illustrated in Fig. 2.4b.

- $r_{12} \approx 0$, $r_{21} \approx 0$: In this case, the ratios are small. This indicates a low affinity to homo-propagation in comparison with cross-propagation. The result will be a so-called alternating copolymer, where every other unit is type 1 and every other unit is type 2.

Termination and chain transfer

Growing (living) polymer chains are terminated by several different mechanisms, which are summarized in Eqs. 2.14 - 2.25 below. For the reactions referred to as chain transfer to monomer, monomer units act to terminate the growing chain rather than to propagate. In this manner, the chain is terminated, and the monomer unit turns "living" in the sense that it now carries a radical. The monomer unit is now itself enable for further propagation, thus establishing a new chain.

[8]The Copolymer Equation is also known as the Mayo-Lewis equation, attributed to Frank R. Mayo and Frederick M. Lewis.

Figure 2.4: Illustration of typical block copolymers (a) and random copolymers (b).

Chain transfer to CTA proceeds in the same way, but instead of having a monomer unit terminate the growing chain, the chain transfer agent of the system is responsible. For termination by chain recombination, two growing chains meet and terminate each other by combining to a longer (dead) chain. The mechanistic idea of this phenomenon is drawn in Fig. 2.5. Chain termination by disproportiation is somewhat similar to termination by recombination, in that two living polymer chains meet and terminate each other. For chain disproportiation, however, the resulting products are two dead chains instead of one long chain. These two chains have different characterics at the endgroup, with one being unsaturated (having a double bond in the chain). The mechanism for this effect is drawn in Fig. 2.6. [2]

Chain transfer to monomer:

$$M_i(s) - P_1 \quad + \quad M_1 \quad \xrightarrow{k_{f11}} \underbrace{M_i(s) - D_1}_{\text{"Dead" chain}} + \underbrace{P_1}_{\text{"Living" monomer}} \tag{2.14}$$

$$M_i(s) - P_1 \quad + \quad M_2 \quad \xrightarrow{k_{f12}} M_i(s) - D_1 \quad + \quad P_2 \tag{2.15}$$

$$M_i(s) - P_2 \quad + \quad M_1 \quad \xrightarrow{k_{f21}} M_i(s) - D_2 \quad + \quad P_1 \tag{2.16}$$

$$M_i(s) - P_2 \quad + \quad M_2 \quad \xrightarrow{k_{f22}} M_i(s) - D_2 \quad + \quad P_2 \tag{2.17}$$

Chain transfer to CTA:

$$M_i(s) - P_1 \quad + \quad CTA \quad \xrightarrow{k_{f,CTA1}} M_i(s) - D_1 \quad + \quad P_1 \tag{2.18}$$

$$M_i(s) - P_2 \quad + \quad CTA \quad \xrightarrow{k_{f,CTA2}} M_i(s) - D_2 \quad + \quad P_1 \tag{2.19}$$

Termination by chain recombination:

$$M_i(s) - P_1 \quad + \quad M_i(r) - P_1 \quad \xrightarrow{k_{tc,11}} M_i(s) - M_1 - M_1 - M_i(r) \tag{2.20}$$

$$M_i(s) - P_2 \quad + \quad M_i(r) - P_1 \quad \xrightarrow{k_{tc,21}} M_i(s) - M_2 - M_1 - M_i(r) \tag{2.21}$$

$$M_i(s) - P_2 \quad + \quad M_i(r) - P_2 \quad \xrightarrow{k_{tc,22}} M_i(s) - M_2 - M_2 - M_i(r) \tag{2.22}$$

Termination by chain disproportionation:

$$M_i(s) - P_1 \quad + \quad M_i(r) - P_1 \quad \xrightarrow{k_{td,11}} M_i(s) - D_1 \quad + \quad M_i(r) - D_1 \tag{2.23}$$

$$M_i(s) - P_1 \quad + \quad M_i(r) - P_2 \quad \xrightarrow{k_{td,21}} M_i(s) - D_1 \quad + \quad M_i(r) - D_2 \tag{2.24}$$

$$M_i(s) - P_2 \quad + \quad M_i(r) - P_2 \quad \xrightarrow{k_{td,22}} M_i(s) - D_2 \quad + \quad M_i(r) - D_2 \tag{2.25}$$

For modeling purposes, the large amount of kinetic rate constants are usually reduced by combining

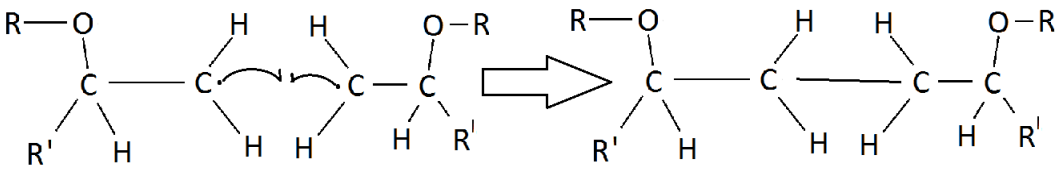


Figure 2.5: Mechanistic idea for polymer termination by chain recombination.

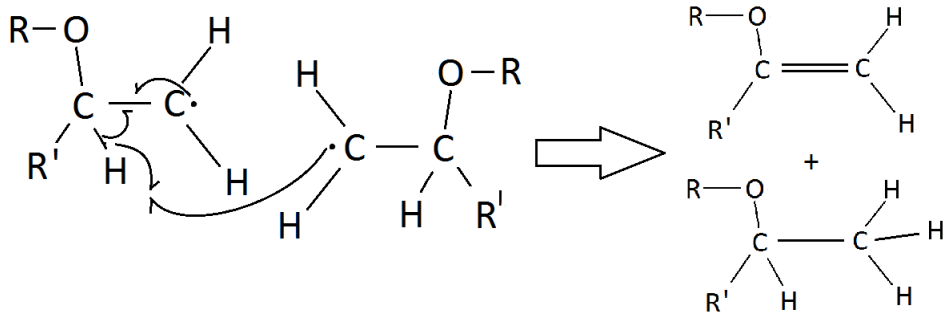


Figure 2.6: Mechanistic idea for polymer termination by chain disproportiation.

some of them to yield common rate constants. Cross-species chain transfer to monomer is modeled as the geometric average of the homo-species transfer, as indicated in Eq. 2.26. The homo-species chain transfer to monomer, which is indicated in Eq. 2.27, is modeled in the same spirit as the Arrhenius-approaches for the previously modeled reaction rate constants.

$$k_{fij} = \sqrt{k_{fii} k_{fjj}} \quad , \quad i, j \; = 1, \, 2 \tag{2.26}$$

$$k_{fii} = \frac{k_{fii,0} \, exp\left(-\frac{E_{fii}}{RT}\right)}{k_{fii,adj}} \tag{2.27}$$

The reaction rate constants for chain transfer to CTA can be modeled in an easy way, by simply combining the contributions from the two types of endgroups, as indicated in Eq. 2.28. Here, an adjustment factor is included aswell.

$$k_{f,CTA} = \frac{k_{f,CTA1} + k_{f,CTA2}}{k_{f,CTA,adj}} \tag{2.28}$$

For the termination reactions, the rate constants are combined into two rate constants, as indicated in Eqs. 2.29 & 2.30. These are again combined and adjusted as shown in Eq. 2.31.

$$k_{tc} = k_{tc,11} + k_{tc,21} + k_{tc,22} \tag{2.29}$$

$$k_{td} = k_{td,11} + k_{td,21} + k_{td,22} \tag{2.30}$$

$$k_T = \frac{k_{tc} + k_{td}}{k_{T,adj}} \tag{2.31}$$

This concludes the listing of the possible chemical reactions in the copolymerization system.

Phase distribution

A classic illustration of the nature of emulsion copolymerization is presented in Fig. 2.7, in order to give an overall representation of the behavior of these types of systems. This serves to sum up the preceding subsections on emulsion copolymerization, and motivate the discussion of phase distribution in such systems.

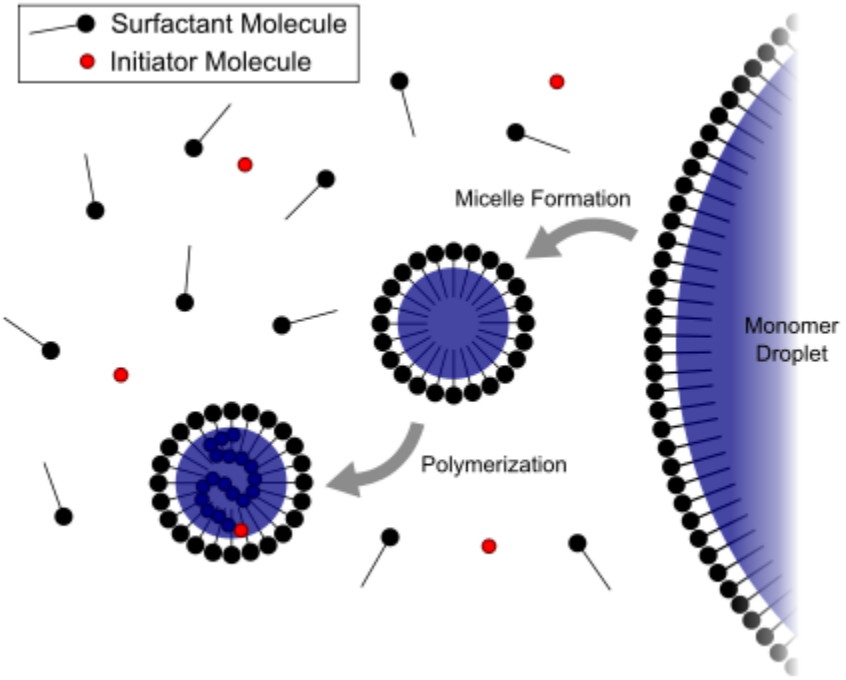


Figure 2.7: A classic illustration of emulsion polymerization.

There are three different phases typically considered in emulsion copolymerization processes, which are all included in Fig. 2.7. The aqueous (water) phase, represented by the white space in the figure, is largely made up of the water fed to the reactor, which acts as a bulk phase, dissolving the two other phases. Because of this, water is an important component within the system with respect to solubility issues, heat transfer properties both between the various phases and with external components, etc. The water phase is also the phase in which most of the initiator, CTA and emulgator/surfactant[9] is dissolved. The aqueous phase can hold both monomer and polymer, but the solubility of these species are usually very low, sometimes even negligible, compared to the solubility in the droplet (monomer) phase and the particle (micelle) phase. The micelles in which the polymer chains form and elongate are usually referred to as the particle phase of the emulsion system. The micelles hold both initiator and CTA, in relatively small concentrations, while the main part of the micelles are polymer chains or unreacted monomer units. The third phase in the

[9]Surfactant: **Surf**ace **act**ive **a**gen**t**. Surfactants are chemical species which align at the interphase boundaries in multiphase systems such as emulsions, dispersions, etc. and act to either promote or prevent phase separation. Since surfactants most often are added in very small quantities, and don't participate in chemical reactions, the surfactants are often omitted from the modeling for simplicity.

emulsion system is the monomer droplet phase, which is made up of unreacted monomer particles which have yet to travel to the particle (micelle) phase to participate in polymer chain propagation. The behavior of the droplet phase depends on the nature of the monomer particles, but also on the operation of the reactor. In several applications, reactors are run under so-called monomer starved conditions, in which the monomer droplet phase will be negligible. To achieve monomer starved conditions, the monomers are typically added carefully along the batch time in a typical semi-batch manner, such that a large excess of monomer is avoided at the beginning of the batch.

In certain copolymerization processes, the gas phase of the system must also be considered. Whether this is necessary or not depends on the nature of the monomers in the system in connection with the operating conditions of the reactor. In several cases for emulsion copolymerization, both in industrial and lab-scale experiments, the modeling of the gas phase is not crucial for the successful modeling of the liquid part of the reactor system.

When modeling the multiphase behavior of the emulsion copolymerization system, mass and energy transfer between the phases must be considered. Although there are several different approaches to model this, depending on the knowledge of the emulsion system, the solubility of the components in the respective phases is usually a proper starting point. In some cases, film models[10], or even more complex models, can be used to describe the internal interphase mass transfer. For agitated (semi-)batch reactors, the assumption of well-mixed reactor contents is widespread, and this helps to support the use of solubility to determine interphase mass transfer. In Eq. 2.32, the transfer of species i from phase w to phase p is formulated as the mass concentration difference between the equilibrium value (for instance given by the solubility of species i in phase p) and the actual value, multiplied with a mass transfer coefficient. The mass transfer coefficient ($k_{w\to p}$) and the equilibrium constant[11] ($H_{i,eq}$) are obvious candidates for off-line parameter estimation in an extensive investigation of the model, at least in the case where detailed considerations with respect to mass transfer has been done.

$$\dot{m}_{i,w\to p} = k_{w\to p}\left(H_{i,eq} w_{i,w} - w_{i,p}\right) \tag{2.32}$$

Free-radical species modeling

In a free-radical copolymerization system, the behavior of the radical species is of key importance. In the previous sections, the important chemical reactions of the system are listed, and they all include active radical species. Because of this, the successful modeling of the radicals is crucial in order to properly describe the chemical behavior of the system. Several approaches are found in the literature for modeling the radicals, and a publication by Li & Brooks is of particular interest in this sense, which will be elaborated below. [8]

For the notation used in modeling the radical species of the system, $\bar{n}$ denotes the average number of radicals per particle. Furthermore, σ, k and C denote generation (radicals entering the particles), desorption (radicals exiting the particles) and decay (termination) of radicals from the particles, respectively. Extra parameters, defined to simplify the expressions, are presented in

[10] Film models are models used to describe diffusion between different phases in a system. [15]

[11] This constant is denoted by H to emphasize the analogy to the Henrys Law-approach known for (dilute) vapor-liquid systems.

Eqs. 2.33 & 2.34.

$$\phi = \frac{2(2\sigma + k)}{2\sigma + k + C} \tag{2.33}$$

$$q = \sqrt{k^2 + 4\sigma\phi C} \tag{2.34}$$

The three approaches for radical species modeling which has been considered in the modeling work for the copolymerization case (see Sec. 3.3) are listed below.

1. Using a dynamic full population balance for particles carrying between 0 and i radicals, with i being a number chosen in advance. Choosing a high value for i gives more complexity, which in turn demands more computational power, but it also increases the accuracy of the radical model. Using a full population balance, the radical model becomes as indicated in Eqs. 2.36 - 2.38. For this purpose, it is necessary to formulate n as indicated in Eq. 2.35, i.e. a vector containing the amounts of polymer carrying the respective numbers of radicals.

$$n = \begin{bmatrix} n_0 \\ n_1 \\ n_2 \\ \vdots \\ n_i \end{bmatrix} \tag{2.35}$$

Here, n_0 is the amount of polymer having zero radicals, n_1 has one radical, etc. The change in the polymer species will then be given as indicated below.

$$\frac{dn}{dt} = An \tag{2.36}$$

$$A = \begin{bmatrix} -\sigma & k & 2C & 0 & 0 & 0 & 0 & \dots \\ \sigma & -\sigma - k & 2k & 6C & 0 & 0 & 0 & \dots \\ 0 & \sigma & -\sigma - 2k - 2C & 3k & 12C & 0 & 0 & \dots \\ 0 & 0 & \sigma & -\sigma - 3k - 6C & 4k & 20C & 0 & \dots \\ 0 & 0 & 0 & \sigma & -\sigma - 4k - 12C & 5k & 30C & \dots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix} \tag{2.37}$$

$$\frac{d\bar{n}}{dt} = \frac{1}{N_T} \begin{bmatrix} 0 & 1 & 2 & \cdots & i \end{bmatrix} \frac{dn}{dt} \tag{2.38}$$

Here, N_T denotes the total number of particles in the system, which is assumed to be constant[12] for these equations to hold. The complete strategy for constructing the A-matrix is given in App. A, where a MATLAB-script is provided to illustrate the general approach, depending on the chosen value for i.

[12] A constant number of particle in an emulsion copolymerization system is usually valid when seed particles for polymerization is used.

2. Using a steady state approximation for average number of radicals per particle. Li & Brooks have written a comprehensible text where the topic is approximations to the full population balance [8]. One of the outcomes is the steady-state approximate solution for the average number of radicals per particle, as indicated in Eq. 2.39. Using this approximation, the number of dynamic states for the system is reduced, but the model loses some of its accuracy.

$$\bar{n} = 2\sigma \frac{1 - exp(-qt)}{(k+q) - (k-q)\, exp(-qt)} \tag{2.39}$$

3. Using a dynamic Li-approximation for average number of radicals per particle. The strategy in this case is similar to the steady state approximation, but an important difference is that the average number of radicals per particle is maintained as a dynamic state of the system. The formulation of this approximation is as indicated in Eq. 2.40. [8]

$$\frac{d\bar{n}}{dt} = \sigma - k\bar{n} - \phi C \bar{n}^2 \tag{2.40}$$

Among these different approaches, the latter is the most used in this work, because it gives a satisfying trade-off between accuracy and complexity. This leads to satisfying computational time without compromising the detail of the model too much. A comparison between the three, illustrated for an entirely fictitious case[13], is illustrated in Fig. 2.8. This example shows that the dynamic Li-approximation agrees very well with the full population balance. The steady state-approximation also shows decent agreement, apart from the behavior at the very start of the simulation.

Copolymer product quality parameters

There are several quantities to calculate in order to investigate the quality/state of the copolymer product. The most common identifier for a copolymer blend is the molecular weight distribution of the copolymer. Usually, two different formulations for molecular weight is used, as presented in Eqs. 2.41 & 2.42, which in turn can be used to formulate a quantity known as Polydispersity Index (PDI), as presented in Eq. 2.43. A comprehensible text on molecular weight distribution for polymer systems is written by Crowley and Choi, made specifically for a free-radical polymerization system for control purposes [13]. The use of polymer moments for characterizing the copolymer system is also discussed by Gao and Penlidis [14] and Wyman [16], among others.

$$\text{Number average molecular weight:} \quad M_n = \sum_{i=1}^{m} (Y_i M_{Mi}) \quad \frac{\mu_1^{M1} + \mu_1^{M2} + \mu_1^D}{\mu_0^{M1} + \mu_0^{M2} + \mu_0^D} \tag{2.41}$$

$$\text{Weight average molecular weight:} \quad M_w = \sum_{i=1}^{m} (Y_i M_{Mi}) \quad \frac{\mu_2^{M1} + \mu_2^{M2} + \mu_2^D}{\mu_1^{M1} + \mu_1^{M2} + \mu_1^D} \tag{2.42}$$

$$\text{Polydispersity index:} \quad PDI = \frac{M_w}{M_n} \tag{2.43}$$

In these formulations, μ_i^j denotes the i'th order moment with respect to component j as active component. $P1$ and $P2$ denote active chains where the endgroup is monomer type 1 and 2, respectively, while D denotes dead/deactivated polymer chains. The various polymer moments are formulated in

[13] Here, dimensionless time is used along the time axis.

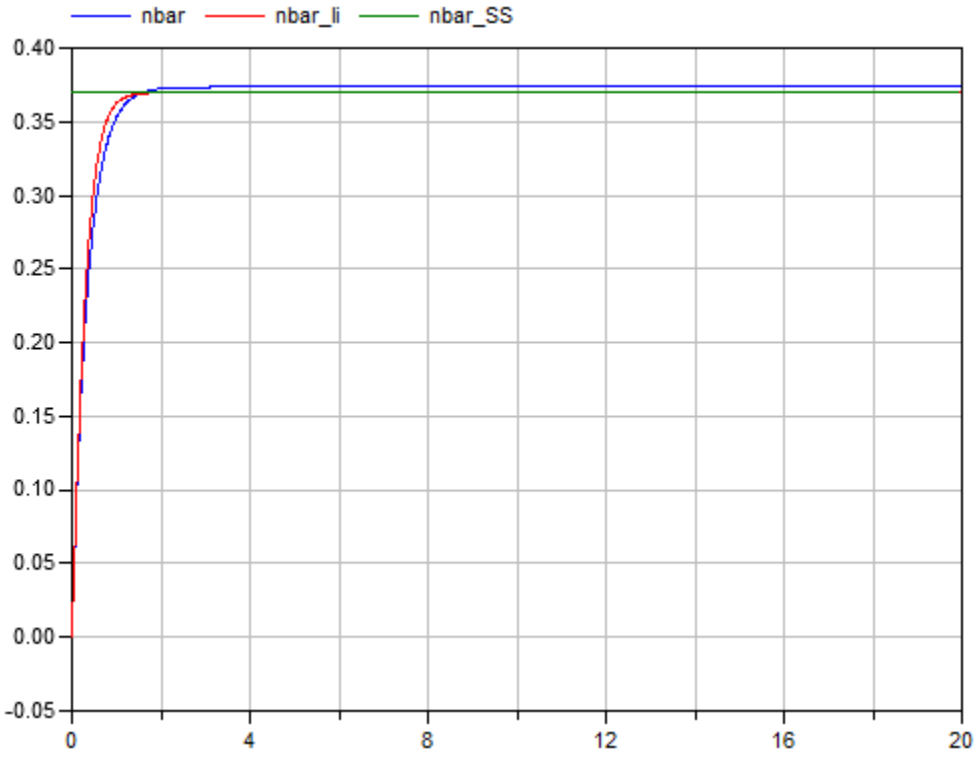


Figure 2.8: A plot showing the agreement between the three various approaches for radical species modeling, for a fictitious case. Curves show average number of radicals per particle versus dimensionless time.

a systematic manner in the appendix, App. B. Furthermore, Y_i denotes the abundance of monomer type i in the polymer, i.e. the relative consumption of monomer type i, while M_{Mi} is the molecular weight for each monomer unit of type i.

The degree of monomer conversion is another important quantity to calculate. This is discussed briefly in the section for semi-batch reactor modeling, Sec. 2.2, and two alternative formulations for monomer conversion is presented in Eqs. 2.53 & 2.54. In most copolymerization processes, the conversion of monomer is desired to be as high as possible. For reasons related to the kinetics of the system, this is not always achievable within the time limits of the batch.

2.2 Fundamentals of semi-batch reactor modeling

The semi-batch reactor is a popular reactor design for applications in polymer synthesis industry. One of the main reasons for this is the ability to better control the feeding of reactants, and hence indirectly control the temperature changes in the reactor in a desirable way. This is a huge advantage, because temperature control is of key importance in many polymer applications, both with respect to product quality and safety. This section provides a short introduction to modeling of semi-batch reactors from a general point of view. [6][7]

The semi-batch reactor has several similarites to the regular batch reactor which, by assump-

tion, is a perfectly mixed tank reactor with no gradients, neither with respect to concentration, temperature or any other intensive quantity, in spatial coordinates. Under this assumption, the modeling is simplified, but the reactor design accounts for time variation. It is important to note that a regular batch reactor does not allow for convective fluid transport across the reactor boundary during the course of the reaction. For a batch reactor, in its most simple form, the design equations (which are the component mass balances and energy balance, respectively) can hence be written as indicated in Eqs. 2.44 & 2.45.

$$\text{Mass balances:} \quad \frac{dN_i}{dt} = r_i V \qquad \text{i} = 1, \ldots, \text{n} \tag{2.44}$$

$$\text{Energy balance:} \quad \frac{dU}{dt} = Q + W_s \tag{2.45}$$

In this formulation, N_i is the amount of species i, U is the internal energy of the reactor, $V(t)$ is the (time dependent) volume of the reactor contents, r_i is the reaction rate of component i, W_s is the shaft work applied to the system, which is usually negligible, while Q is the energy change associated with external heat transfer (heating, cooling, heat loss, etc.). This model describes a system with n chemical species, each with its independent mass balance. The reaction rate of each component in the system (r_i) can be modeled in several ways, depending on the complexity and the nature of the chemical reactions. A common approach is to calculate the reaction rate using a rate law, in which the reaction rate is assumed to vary with the concentration of the respective component to some order, like indicated in Eq. 2.46. In this expression, k indicates the order of the rate law, and k_i is a reaction rate constant, for example calculated as indicated in Eq. 2.9 for the copolymerization case. In Eq. 2.9, the Arrhenius temperature dependency of the reaction rate constant is used, and this is a widespread approach for modeling kinetics of this kind.

$$r_i = k_i c_i^k \tag{2.46}$$

In some applications, assumptions can be made to simplify the model equations further. For instance, a so-called temperature explicit energy equation can be achieved under the assumption that the heat capacity (indicated by C_p in Eq. 2.47) of the reactor content is time independent. This assumption often holds for aqueous solutions, among other systems. In cases where this assumption does not hold, the heat capacity should be modeled dynamically as a temperature function in order to achive the temperature explicit form of the energy balance. The motivation for obtaining a temperature explicit energy balance is that the instensive property temperature appears much more tangible than energy. The fact that the measuring of temperature can be performed in a straight-forward manner for many applications also makes temperature a preferred quantity over energy. Another simplification for the model equations comes along when the total volume of the reactor content can be assumed to be constant as time passes in the reactor. Using $c_i = \frac{N_i}{V}$ (where c_i denotes the concentration of species i), this simplifies the molar mass balances. These common simplifying cases are given in Eqs. 2.47 & 2.48.

$$\text{Mass balances:} \quad \frac{dc_i}{dt} = r_i \qquad , \quad \text{i} = 1, \ldots, \text{n} \tag{2.47}$$

$$\text{Energy balance:} \quad mC_p \frac{dT}{dt} = W_s + Q \tag{2.48}$$

The model equations for the batch reactor are the basis for the approach towards a semi-batch reactor model which, in addition to "just" being a time dependent reaction vessel, also allow for

external mass transfer across the reactor boundary. In the COOPOL copolymerization reactor process, which is the basis for this work, for instance, the reactants are added gradually during the reaction time, instead of adding them all at the start of the reaction (which would be the regular batch reactor approach). Another strategy of interest could be to extract product gradually during the course of the batch, which is also allowed in the semi-batch reactor approach. The reactor design now approaches the nature of a continuous reactor, in which CSTR[14] probably is the closest relative. Effects from the flowing fluids, both with respect to species mass balances and the energy balance must now be added, and the volume of the fluid contents must be given attention as well.

$$\text{Mass balances:} \quad \frac{dN_i}{dt} = r_i V(t) + \dot{n}_i \quad \Longrightarrow \quad \frac{d}{dt}\left(c_i V(t)\right) = r_i V(t) + \dot{n}_i$$
$$\Longrightarrow \qquad V(t)\frac{dc_i}{dt} + c_i \frac{dV}{dt} = r_i V(t) + \dot{n}_i \tag{2.49}$$

In Eq. 2.49, $\dot{n}_i$ is the net flowrate of species i to the reactor, i.e. the difference between the inflow and the outflow. In this general approach, effort is made to emphasize that the species mass balances hold for formulations on both molar basis and mass basis, as long as the corresponding reaction rates etc. are treated in agreement with this, i.e. using the correct units. The mass basis is perhaps the most established and intuitive approach, and this is mainly due to the principle of conservation of mass. This enables the development of useful equations to describe the system, e.g. volume balances[15] or dynamic equations for the composition of the respective species in the reactor. In the expression below (Eq. 2.51), ρ is the density of the fluid mixture residing in the reactor.

$$\frac{dm_{tot}}{dt} = \dot{m}_{in} - \dot{m}_{out} \tag{2.50}$$
$$\Longrightarrow \qquad \frac{d}{dt}\left(\rho V\right) = \dot{m}_{in} - \dot{m}_{out}$$
$$\Longrightarrow \quad \rho\frac{dV}{dt} + V\frac{d\rho}{dt} = \dot{m}_{in} - \dot{m}_{out} \tag{2.51}$$

This equation can be combined with the species mass balances, Eq. 2.49, to eliminate the derivative of the volume from the mass balance if desired, but that also raises the need for a differential equation in time expressing the density variation. In many cases, simplifications can be introduced to make the model easier to treat, yet still represent reality in a good way. For instance, for a fluid with constant density and no fluid flow out of the reactor, this equation intuitively reduces to Eq. 2.52, which represents a special simplified case for the semi-batch reactor case. A conceptual drawing of a semi-batch tank reactor is given in Fig. 2.9.

$$\frac{dV}{dt} = \dot{V}_{in} \tag{2.52}$$

The conversion of reactant in a semi-batch reactor can be formulated in several ways. In Eqs. 2.53 & 2.54, two different formulations are shown for a batch simulated from $t = t_0$ to $t = t_1$

[14] Continuously stirred tank reactor: A time-invariant, perfectly mixed tank reactor.

[15] The expression "volume balance" should be used with care, since the volume itself is not a conserved quantity. In the same way an energy balance can *manifest* itself as an "enthalpy balance" in certain cases, the mass balance can be reformulated to a "volume balance" if this is desired by the modeler.

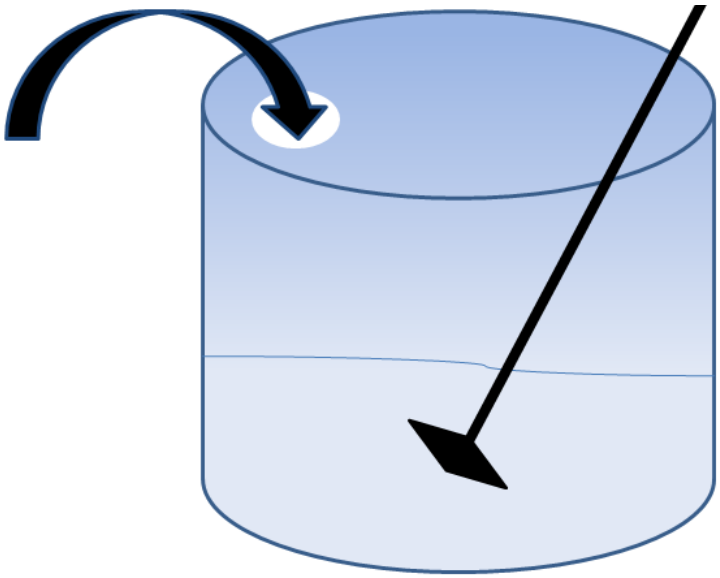


Figure 2.9: Conceptual illustration of a semi-batch tank reactor with stirrer and continuous feeding.

in time. Here, $R(t)$ denotes the molar amount of reactant in the reactor at time t and $\dot{n}_R(\tau)$ denotes the net flow of reactant at time τ[16]. The two expressions have striking similarities, but while the "continuous" (also referred to as instantaneous conversion) formulation considers the conversion with respect to accumulated monomer at time t, the "total" (also referred to as global conversion) formulation considers the conversion with respect to accumulated monomer at time t_1. That is, the total amount of monomer for the entire batch time. For the considerations done with respect to parameter fitting in Sec. 4.1, the "total" formulation is applied.

$$\text{"Total" formulation:} \quad X(t) = \frac{R(t_0) + \int_{t_0}^{t} \dot{n}_R(\tau)d\tau - R(t)}{R(t_0) + \int_{t_0}^{t_1} \dot{n}_R(\tau)d\tau} \tag{2.53}$$

$$\text{"Continuous" formulation:} \quad X(t) = \frac{R(t_0) + \int_{t_0}^{t} \dot{n}_R(\tau)d\tau - R(t)}{R(t_0) + \int_{t_0}^{t} \dot{n}_R(\tau)d\tau} \tag{2.54}$$

With this in mind, attention can be given to the energy balance of the reactor. The semi-batch reactor is an open system, and the energy balance will take effect from the mass flow dynamics of the semi-batch reactor. This represents a difference from the regular batch-reactor, which is a closed system without external convective flows. In the following expression, h and h_0 denote the specific enthalpy of the fluid in the reactor and the fluid feed, respectively. Note that due to the assumption of a perfectly mixed reactor vessel, the intensive properties will be the same in the entire reactor, and the specific enthalpy of the exiting fluid will hence be that of the uniform fluid residing in the reactor. The simplified energy balance, where external shaft work applied to the reactor is neglected[17], is proposed in Eq. 2.55. In addition, the enthalpy changes due to chemical

[16]The greek letter τ is here introduced to be used as an integration variable, with the main purpose being to avoid confusion when it comes to the variable t denoting time.

[17]This assumption does not generally hold. For many tank reactors, however, it does hold despite the fact that

reactions in the reactor system is important to consider in the modeling work.

$$\frac{dU}{dt} = Q + \dot{m}_{in} \cdot h_0 - \dot{m}_{out} \cdot h \tag{2.55}$$

For many reactor applications, the very purpose of modeling the energy balance is to achieve information about the temperature, which is a quantity of key importance in most cases, with respect to monitoring the behavior of the system. For a typical free-radical copolymerization reaction system for instance, as described in Sec. 2.1, the reactor temperature is of key importance for both the performance and safety considerations of the chemical reactor. Another reason for bothering with having the temperature as a key process variable is the fact that temperature, in contradiction to energy and enthalpy, is an intuitive and tangible quantity. For many reactor applications, the temperature is also easy to measure without having to deal with a significant time delay, thus providing a safe and reasonably accurate indirect measure of the state of the reactor. By modifying the energy balance (Eq. 2.55), the expression in Eq. 2.56 is achieved, thus allowing to specifically monitor the change in temperature over time for the reactor.

$$\begin{aligned} \frac{d}{dt}\left(\underbrace{m_R c_{p,R}(T - T_{ref})}_{\text{Reactor vessel}} + \underbrace{m c_p (T - T_{ref})}_{\text{Reactor contents}}\right) &= Q + \dot{m}_{in} \cdot h_0 - \dot{m}_{out} \cdot h \\ \implies \qquad \frac{dT}{dt} &= \frac{Q + \dot{m}_{in} \cdot h_0 - \dot{m}_{out} \cdot h}{m_R c_{p,R} + m c_p} \end{aligned} \tag{2.56}$$

In typical semi-batch reactor cases, the collection of balance equations add to yield a system of ordinary differential equations which must be solved simultaneously. This concludes the brief discussion on first principles modeling of semi-batch reactors. These concepts were applied when the model for the specific copolymerization case, which is described in Sec. 3.3, was developed.

2.3 Introduction to off-line estimation and constrained optimization

When designing models from first principles, most of the parameters used in the modeling work are more or less uncertain, as they are approximated, guessed from experience with similar systems, etc. Some parameters may even be entirely fictional due to "shortcuts" taken by the modeler. An example of such a case could be the interphase mass transfer for a system in which the diffusion is very complicated to describe. The specific phase(s) may even be assumed to be ideally mixed, leading to a case without intraphase concentration gradients and mass transfer. In such a case, a good strategy could be to soften the interphase[18] mass transfer process with a fictional time constant, which in turn will be modified for the model to fit reality in a best possible way. The scope of this section and the next is to motivate and explore the theory of both off-line and on-line state and parameter estimation for implementations on dynamic systems. This is a crucial part in a (nonlinear) model-based controller implementation.

For a process model to be valid for a specific system, the measurable outputs as predicted by the model need to match the reality as suggested by measurements from the corresponding real

the tank reactor is agitated, because the agitator contribution is small compared to the contributions from chemical reactions and cooling/heating.

[18] Emphasis is made to notify the distinction between interphase (transfer between two separate phases) and intraphase (transfer within one phase) mass transfer.

system. The constant (or weakly varying) parameters of the system must, in other words, be chosen such that the system outputs agree with measurements. With this motivation, methods for off-line parameter estimation are deployed to adjust, and thus improve, a model by modifying a set of parameters, such that the output predicted by the model agrees with reality (experimental data measurements) to a satisfying degree. As indicated in the previous paragraph, the total set of parameters may include parameters which are known to a sufficient degree, while some parameters are quite uncertain. From these, the most uncertain should be subject to modification while the certain parameters should generally remain the same. In this sense, the parameter estimation is a preliminary method of improving the process model prior to the on-line implementation. A general equation system for the model of a dynamic system is presented in Eqs. 2.57 - 2.59. Here, x is the vector of states for the system, u is the vector of inputs, while θ denotes the vector of parameters. That is, all the parameters for the system, which are assumed to be constant or weakly varying through the course of time (t) for the system. The respective values of the parameters may be more or less uncertain for this general case.

$$\frac{dx}{dt} = f(x, u, \theta, t) \tag{2.57}$$

$$0 = g(x, u, \theta, t) \tag{2.58}$$

$$y_p = h(x, u, \theta, t) \tag{2.59}$$

For this kind of system, given a vector of measurements in time (y_m), a selection of parameters (η) is chosen from the entire collection of parameters (θ), which will be subject to modification. For super-simplified models with few parameters, the modification can be achieved by manually adjusting the parameters in a trial-and-error manner in which the agreement between the model and the measurements is evaluated by the modeler. The systematic approach to parameter modification, however, is achieved by solving an optimization problem in which the sum of the deviations between the model predictions and the measurements is minimized. For most cases encountered in real-life applications, the process models are nonlinear and complex, containing a variety of parameters. The parameters are often interconnected with the states in an intricate manner, such that the immediate effect on the overall system behavior from changing the respective parameters may not necessarily be entirely intuitive. These characteristics all point toward using the systematic approach rather than the (tedious) trial-and-error approach. The general optimization problem to be solved is represented, mathematically, in Eq. 2.60. [12]

$$\begin{aligned} \min_{\eta} \quad & \sum_{k=1}^{N} \left(y_{p,k}(x, u, \theta, t) - y_{m,k}\right)^2 \\ & \eta \in \theta \\ \text{s.t.} \quad & \frac{dx}{dt} = f(x, u, \theta) \\ & 0 = g(x, u, \theta) \end{aligned} \tag{2.60}$$

The Cybernetica ModelFit software, for which a brief introduction is provided in Sec. 3.2, is able to establish and solve these kinds optimization problems for off-line cases in an elegant manner, but an advantage of using this software is the potential of extending the parameter estimation to on-line use. In addition, the software can include intial values for the states of the system into the

optimization problem. An alternative way to do off-line parameter estimation is to use the built-in MATLAB function `lsqcurvefit`, which is a least squares curve fitting tool, approximating parameter values for a function to fit data. The formulation is as presented in Eq. 2.61 which, generally speaking, is completely analogous to the problem introduced in Eq. 2.60. In this case, $F(x, \theta)$ is a general nonlinear function which is evaluated at discrete points in time (k), each corresponding to a measured value for the function output (y_k). Both x and θ are inputs to the function, but θ denotes the vector of parameters which is desired to be recalculated for the function to fit measurements better. By minimizing the sum of squares of the deviation between the model prediction and the measured value with respect to θ, optimal θ-values are found for the function.

$$\min_{\theta} \sum_{k=1}^{N} \left(F_k(x, \theta) - y_k\right)^2 \tag{2.61}$$

This approach is analogous to the strategy for off-line parameter estimation performed in the Cybernetica ModelFit software, which is evident from the comparison of Eq. 2.61 with Eq. 2.60. The Cybernetica ModelFit software is briefly described in Sec. 3.2.

Solving optimization problems of this type can be complicated, mainly because the function involved is nonlinear. In addition, the problems are often constrained. Unconstrained globally convex[19] problems with quadratic cost functions are relatively managable to solve, but this class of problems are not often encountered when treating real-life processes. Usually, the optimization problems are only locally convex or maybe not convex at all due to the nonlinearity and complexity of the system models. In dealing with real-life applications, the parameters are physical parameters in the sense that they cannot contradict common sense or the laws of nature. The optimization problems hence have to be provided with sensible constraints to maintain the physical validity of the solution. Examples of invalid situations are negative valued volumes and negative valued species concentrations. Another example could be heat transfer coefficients, in which the optimization problem may suggest values which greatly exceed previously recorded values for the specific materials involved. Common for these cases is that while the solution may be feasible and sensible in a mathematical sense, the physical interpretation is invalid. Because of this, the importance of sensible parameter constraints when performing parameter estimation is emphasized.

For a linear function, the solution to the optimization problem would be relatively straightforward to formulate and find, because this leaves a quadratic optimization problem. Methods for quadratic programming (QP) are well-established, and such problems are usually solved using line-search methods (LSM). Typical LSMs are the Steepest Descent Method, the Newton Method and various Quasi-Newton methods (in which the Hessian matrix from the Newton Method is approximated to decrease the computational effort). While details on these methods are omitted from this text, the strategy is to first decide the best direction of step in space[20]. Based on this, the step length is calculated, and this alternating procedure is carried out at each iteration of the solution method. Alternatively, so-called trust region methods (TRM) could be deployed, in which a region for the solution of the problem is expanded or contracted depending on whether the objective function is adequate in the region or not. [11]

When the system equations are nonlinear, the optimization problems turn more difficult to solve.

[19] Convexity is an important property in numerical optimization. For a convex problem, a local minimum will also be the global minimum and hence the optimal solution to the problem. [11]

[20] Space here referes to the entire space of the system, i.e. the dimensions of the system. For $x \in R^n$, this would mean n-dimensional space. A step in space simply denotes changes in the respective variables in n-space.

Several approaches and algorithms exist to treat problems like this. Among them are differentiation-free optimization (DFO) methods like the Nelder-Mead[21] algorithm, but the most common approach is probably the strategy of sequential quadratic programming (SQP). In this case, the optimization problem is reformulated as an approximate quadratic problem, which is solved as a series of QPs in an iterative manner. The Cybernetica ModelFit software (Sec. 3.2) uses SQP to solve optimization problems in which the model functions are nonlinear. The MATLAB alternative for optimizing nonlinear functions is the built-in `fmincon` function, in which a nonlinear optimization problem is solved with given constraints using an SQP algorithm. A thorough walkthrough on the solution strategy for SQP problems is not provided in this text. For more details on numerical optimization, the reader is referred to more extensive texts, e.g. Nocedal & Wright. [11]

A continuation to the theory of estimation for dynamic systems is provided in the next section, in which state and parameter estimation is considered for on-line applications.

2.4 On-line estimation and filtering

For on-line implementations, estimation of both states and parameters remain as important subjects. A block diagram illustration[22] of a typical MPC implementation is provided in Fig. 2.10. The

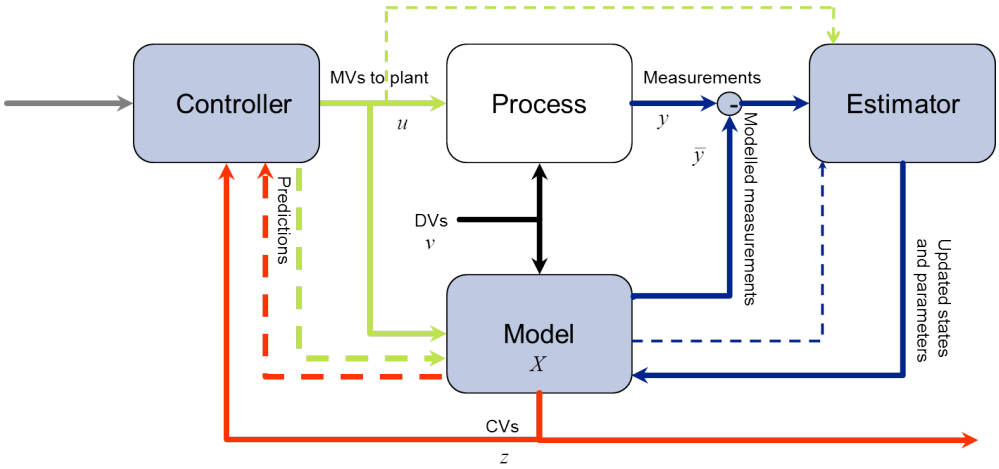


Figure 2.10: A conceptual block diagram showing the idea of an MPC controller scheme.

figure illustrates the roles and interconnection of the key components of such an implementation. Here, CVs denote controlled variables (i.e. system outputs), MVs denote manipulated variables (i.e. system inputs) and DVs denote disturbance variables to the system. In this setup, the estimator block is included, in close connection with the process model block. It is emphasized that the degree in which the process model represents the actual process depends on the quality of the process model. In most applications, there are deviations between the two, and to account for this is the quintessential purpose of the estimator. In other words, the estimator will continuously reinforce the process model depending on the outputs from the real process. The efficiency of the estimator is, conversely, largely dependent on the quality of the process model, as this provides the

[21] This algorithm is attributed to John Nelder & Roger Mead. The algorithm is also known as the Downhill Simplex Method or the Amoeba Method.

[22] This figure is printed with the permission of S. O. Hauger, Cybernetica AS.

basis for the estimator algorithm. The success of these mutual interactions will in turn enable the controller to choose the best possible action with respect to the controller objectives. The scope of this section is to perform a brief investigation of the behavior of the estimator block in Fig. 2.10. Extensive texts on on-line estimation are available, and the findings in this work is largely based on Nocedal & Wright [11], Rawlings & Mayne [17] aswell as Simon [18]. In addition, the study has been inspired by a comprehensible publication on dynamic estimation, including Kalman filtering, written by Schei [12].

When considering on-line estimation, there are several approaches to choose from when developing an estimator algorithm. The most common class of approaches is probably the estimator class known as the Kalman filter (KF). A systematic development of the Kalman[23] filter estimator equations for dynamic systems is provided in the appendix to this text, App. C. This additional section is included to establish a more complete theoretical background for the purpose of on-line estimation of states and parameters. A justification for locating this section in the appendix is the fact that these considerations remain largely theoretical, and without immediate implications for the results of this specific project work. The established theory for on-line estimation will, however, be crucial in the proposed extension to this work, where model-based predictive controller design, i.e. the controller block of Fig. 2.10, is considered. In such an event, all the components needed for an MPC implementation would be considered. Because of this, some of the results from App. C are presented and discussed here.

For the purposes of treating free-radical copolymerization reactors, systems with a high degree of nonlinearity are encountered, and the theory of estimation need to account for these effects. As discussed in App. C.3, there are several approaches to treat nonlinear systems. With increasing complexity comes increasing accuracy, and consequently also a larger demand for computational effort. One of the most straight-forward methods to deploy is the so-called Extended Kalman filter (EKF), which is elaborated in App. C.3. App. C is devoted to derive the various formulations for the Kalman filter depending on the characteristics of the system, and the most essential results are reproduced here.

For a system having both continuous dynamics and continuous measurements, the equations for the EKF become as indicated in Eqs. 2.62 - 2.64. In this case, both process noise (w) and measurement noise (v) are continuous variables with covariances Q and R, respectively.

$$\dot{\hat{x}} = f\left(\hat{x}, u, w_*, t\right) + K\left(y - h\left(\hat{x}, v_*, t\right)\right) \tag{2.62}$$

$$K = PC^T\tilde{R}^{-1} \tag{2.63}$$

$$\dot{P} = AP + PA^T + \tilde{Q} - PC^T\tilde{R}^{-1}CP \tag{2.64}$$

In this consideration, $\hat{x}$ is the state estimate, and $\dot{\hat{x}}$ denotes the state estimate derivative with respect to time. K_k is the gain of the Kalman filter, which uses the deviation between measured output (y) and predicted output ($h\left(\hat{x}, v_*, t\right)$) to update the state estimates. P is the covariance vector for the state estimates, while A is a system matrix originating from the formulation of the continuous system (Eq. C.1). These results are derived for a system in which both the model dynamics and the measurements are continuous, but this may not necessarily always be the case. In most real-life applications encountered, the dynamic system is modeled using a continuous formulation, while measurements are only available at specific points in time. To overcome this challenge, the so-called Hybrid EKF can be deployed, in which the formulations for continuous and discrete KF are

[23] Attributed to Rudolf E. Kàlmàn (1930 -), award-winning Hungarian-American mathematical systems theory scientist and electrical engineer.

combined to account for the behavior of the real-life system. In such an event, the equations for the estimator become as shown in Eqs. 2.65 - 2.69.

$$\text{Between } t_k^+ \text{ and } t_{k+1}^-: \qquad \dot{\hat{x}} = f\left(\hat{x}, u, w_*, t\right) \tag{2.65}$$

$$\dot{P} = AP + PA^T + \tilde{Q} \tag{2.66}$$

$$\text{Between } t_k^- \text{ and } t_k^+: \qquad K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + \tilde{R}_k\right)^{-1} \tag{2.67}$$

$$\hat{x}_k^+ = \hat{x}_k^- + K_k \left(y_k - h_k\left(\hat{x}_k^-, v_*, t_k\right)\right) \tag{2.68}$$

$$P_k^+ = \left(I - K_k H_k\right) P_k^- \left(I - K_k H_k\right)^T + K_k \tilde{R}_k K_k^T \tag{2.69}$$

Here, $\hat{x}_k$ is the state estimate at time t_k, where $\hat{x}_k^-$ and $\hat{x}_k^+$ denote the *a priori* and *a posteriori* state estimates, respectively. K_k is the Kalman filter gain, which uses the deviation between measured output (y_k) and predicted output ($h_k\left(\hat{x}_k^-, v_*, t_k\right)$) at time t_k to decide the correction between the *a priori* and *a posteriori* state estimate. Like for the EKF in Eqs. 2.62 - 2.64, P is the covariance for the state estimates, with P_k^- and P_k^+ denoting the *a priori* and *a posteriori* covariances at time t_k, respectively. A still represents the system matrix from the formulation of the continuous system (Eq. C.1).

Up until now, this entire section aswell as App. C have been largely aimed at state estimation. This is undoubtedly an important subject, but the development of the estimator algorithms for on-line state estimation also allows for on-line parameter estimation. In Sec. 2.3, the parameters of the system were referred to as constant or weakly varying. In the latter case, an interesting task is to use the estimator algorithm to also estimate parameters, and not only the states of the system, according to the measurements. In contrast to the strategy of Sec. 2.3, where preliminary off-line parameter estimation was introduced, this represents a method to address changes in weakly varying parameters for on-line use. From the total set of parameters for the system (θ), a sub-set (η) was chosen for optimization in the off-line parameter estimation study preliminary to the on-line case. Among these, some parameters (λ) may be expected to vary during the course of time for the system. These parameters are added to the vector of states as indicated in Eq. 2.70, and the resulting state vector (x') is referred to as the augmented state vector. It is emphasized that the new model function (f') is a reformulated edition of the original function (f), which accounts for the changes made in x to yield x'. The new model function also includes process noise affecting the parameters (w_λ), which is the source of changes in the parameter values.

$$x' = \begin{bmatrix} x \\ \lambda \end{bmatrix} \tag{2.70}$$

$$\dot{x}' = f'\left(x', u, w, w_\lambda\right) \tag{2.71}$$

Apart from the reformulation where the state vector has been extended to yield the so-called augmented state vector and the model function is adjusted, the estimator algorithms will be the same as before. The established estimator equations can hence be used directly, keeping in mind that the estimates now contain both state estimates and parameter estimates. The measurement function (h/h_k) must also be adjusted in agreement with this.

3 Model description and software features

This section introduces the software tools utilized in the work. Sec. 3.1 aims to introduce the programming language Modelica, which has been used to formulate the models used in the work. A brief description of the Dymola software, which is deployed to treat the Modelica code, is also provided. In Sec. 3.2, the Cybernetica ModelFit software is introduced, which is used for parameter fitting and simulations of the process model. A walkthrough of an example for a simplified string pendulum is provided in the appendix, App. F, in which the purpose is to illustrate the use of the various software tools that were deployed for the main case of the project work. While this represents a simple model (with only two states and two parameters), the example still illustrates the elegancy with which the process model can be formulated in Modelica and exported to the Cybernetica ModelFit software for simulation and parameter fitting. This example does not, however, illustrate the potential of assigning various subunits in a modular/hierarchical manner, but this is elaborated for the main reactor modeling case in Sec. 3.3.

Sec. 3.3 is written to include details and characteristics for the specific copolymerization case considered in this work, which is a semi-batch free-radical emulsion copolymerization process. Details on the specific reactor system are protected under confidentiality of the COOPOL project, and details on the respective monomers of the system has been omitted. The monomers are simply referred to as *monomer 1* and *monomer 2*. Details on the batch time are also omitted, and scaled dimensionless time has been used for the simulation. With the established theoretical background from Secs. 2.1 & 2.2, the specific reactor system is described. The treatment of this reactor system is considered in Sec. 4.

3.1 A brief introduction to Modelica & Dymola

Modelica is a programming language for object-oriented programming which was first released in 1997. Since then, the language has been developed to treat a range of applications within different fields, and although Modelica may not be the most abundant programming language with respect to world-wide use, the applications are numerous, for instance in the automotive industry. In being object-oriented, Modelica has similarities to classic programming languages like C++ or Java, but it also has differences, and Modelica is often referred to as a *modeling language* rather than a *programming language*. Among the strengths are easy declaration and treatment of variables, aswell as strong performance with respect to numerical efficiency and computational time. An example of a Modelica script is presented in Fig. F.2 in App. F, in which the dynamic model for a simplified pendulum is implemented as an example.

The Modelica language is excellent for solving DAE-systems[1]. The mathematical representation of such a problem is formulated in Eqs. 3.1 & 3.2. Eq. 3.3 shows how measurements from the process are *predicted* from the state of the system, but this measurement prediction is a concern of the application to a real system rather than a concern for the Modelica modeling in itself.

$$\dot{x} = f(x, u, \theta, t) \tag{3.1}$$

$$0 = g(x, u, \theta, t) \tag{3.2}$$

$$y_p = h(x, u, \theta, t) \tag{3.3}$$

[1]DAEs: A system of differential and algebraic equations, in contrast to ODE-system, which only contains ordinary differential equations.

In this formulation, x is the vector of states for the system, y_p is the vector of predicted system outputs corresponding to a vector of measurements from the real system (y_m), u is the vector of inputs to the system, and θ is the vector of parameters. The functions f, g and h indicate general functions for the system, describing the nature of the system. The dot-notation is used to indicate differentiation with respect to time. This is, generally speaking, the same system that was introduced for the general discussion of parameter fitting and numerical optimization in Sec. 2.3.

Dymola[2] is a proprietary software for implementation of Modelica code. In that sense, Dymola supports both editing, compilation and simulation (i.e. numerical integration etc.) of Modelica code. When solving problems like the one in Eqs. 3.1 & 3.2, Dymola deploys powerful algorithms which account for stiffness problems, etc. In addition to this, Dymola is able to produce simulation plots and data files for use in other applications. A valuable feature of the Modelica/Dymola combination, which is mentioned in Sec. 3.2, is the possibility of exporting the whole model, and not only the simulation results, to other applications using the so-called model exchange.

A special treat of Dymola is the opportunity to use a graphical user interface (GUI), which enables the user to perform graphical drag-and-drop modeling of processes, given the desired components/subunits of the process[3]. In Fig. 3.5 in Sec. 3.3, the graphical interface is displayed for a reactor test case. This promotes the use and development of modeling libraries for various types of processes, and represents a large potential with respect to reusability and modificability of models, which can prove to be a huge advantage to the modeler. With these possibilities, Dymola can be used as a dynamic process simulator, like Aspen Plus or Aspen HYSYS, with the option of manipulating each unit[4] as desired using the Modelica language. This fascilitates a large degree of customization in the modeling work. Using a modular and hierarchical approach to modeling also enables the modeler to interchange and/or modify certain parts of the model without compromising the respective other components or the overall structure of the model. This strategy is illustrated in Fig. 3.1. Fig. 3.1a and Fig. 3.1b show two different approaches for modeling a specific system, whereas Fig. 3.1b is the modular approach. The two approaches yield the same overall model, and the simulation results will be similar, but the advantage of the modular approach is, as already mentioned, the possibility of *easily* exchanging certain parts/components without compromising the total structure of the model. The benefits of this strategy are elaborated in Sec. 3.3, when the specific test case for the copolymer reactor is introduced and described.

3.2 The Cybernetica ModelFit software

Cybernetica ModelFit is a software developed by Cybernetica AS for state and parameter estimation. The ModelFit software is designed to work alongside software for nonlinear model-based predictive control (NMPC), but ModelFit is in itself a model simulator tool which can be used to run ballistic model simulations and perform off-line parameter fitting, which is the purpose of this work.

To utilize the ModelFit software, the traditional approach is to formulate the specific process model in the programming language C, as this provides a suitable basis for importing the model

[2]Dymola: **Dy**namic **Mo**deling **La**boratory, software developed by Dassault Systemes AB, Sweden.

[3]The approach of designing subunits which are combined to yield the complete process model, rather than creating one single large model script, is referred to as the modular approach.

[4]One of the main drawbacks of Modelica/Dymola in comparison with the mentioned commercial process simulator tools is the fact that apart from the fields of electrical circuit modeling, mechanical modeling and fluid flow modeling, the standard libraries of Dymola are scarce. Because of this, the modeler is sometimes required to make components "from scratch" or adopt components made by a third-party contributor.

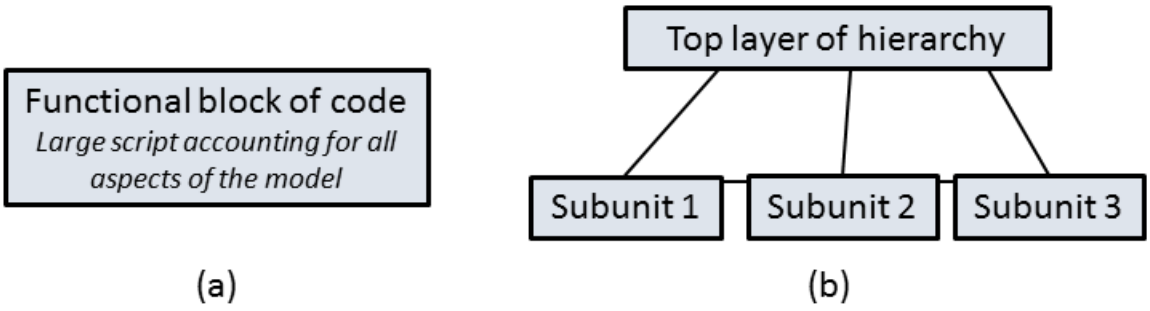


Figure 3.1: A figure showing the general idea of interconnecting Modelica units in a hierarchy.

into the ModelFit software. ModelFit is, in other words, programmed to recognize the various features of the model from the formulation of the model script. For models formulated in other environments than C, a technique[5] for so-called model exchange exists, which is referred to as the Functional Mock-up Interface (FMI). The FMI is a tool to establish a platform which enables the communication between models and applications initially formulated in different languages. One of the supported languages in the FMI is Modelica, and this motivates the use of Modelica for these kinds of applications, in agreement with the introduction given in Sec. 3.1. Modelica models can, in other words, easily be exported to the ModelFit software, and the ModelFit software will even recognize which variables are inputs and which variables are outputs, given that the Modelica model is formulated correctly. The FMI translation tool, which is relatively easy to administer, is implemented in the Dymola software.

Given an imported model, e.g. using the model exchange strategy as introduced above, ModelFit can be used to test models, and perform off-line parameter fitting using given experimental measurements. ModelFit has built-in routines for treating DAEs, where the user can choose among several alternatives. Simple Euler-integration is available, where the user is inclined to set step length of the method, and a mid-point Euler method is also implemented. The most sophisticated tool available to the user for solving the system equations is the CVODEs tool[6]. The choice of solution method depends on the complexity and stiffness of the system. The trade-off between computational time and accuracy/convergence of the solution may also play a part in deciding the solution method.

In the procedure for calculating the optimal parameter values for a model to fit experimental measurements, the software solves a minimalization problem with respect to a chosen set of parameters, in which the sum of squares for the deviations between the model predictions and the experimental measurements is desired to be as low as possible. This is formulated mathematically in Eq. 3.4.

[5]The FMI is developed as a part of the MODELISAR European project, which had the german automotive company Daimler AG among its main initiators and contributors.

[6]CVODEs is a tool developed under the SUNDIALS (**Su**ite of **N**onlinear and **Di**fferential/**Al**gebraic equation **S**olvers) project at the Center for Applied Scientific Computing of the Lawrence Livermore National Laboratory, designed for numerical solution of stiff DAEs. [9]

$$\min_{\eta} \sum_{k=1}^{n_{ky}} (y_{p,k} - y_{m,k})^2 \qquad (3.4)$$
$$\eta \in \theta$$

In this formulation, $y_{p,k}$ and $y_{m,k}$ are model predictions and measurements, respectively, for the sample point k. The number of valid measurements is represented by n_{ky}, while η is the selection of parameters among all the parameters (θ) which are chosen for optimization.

For setting up and solving these kinds of optimization problems, the ModelFit software has several additional features. The software is constructed to provide a user-friendly way to decide what parameters to estimate, i.e. to decide η from θ. If the user also wishes to estimate initial values for the states of the system, these variables are accessed and activated for estimation in the same simple manner as for the parameters. In agreement with what was mentioned regarding the physical interpretation of the parameters in Sec. 2.3, the maximum and minimum values for the respective variables are easily accessible for the user. In addition to this, ModelFit provides valuable supporting calculations during the course of the optimization procedure, making it easier for the user to evaluate the quality of the solution. One of these calculated quantities is the scaled Hessian condition number, for which an example is illustrated in Fig. 3.2. Generally speaking, the condition number indicates how the output value of a function will respond to a change in the input value(s). For the case of parameter fitting, the function in mind is the cost function as introduced in Eq. 3.4. A high value for the condition number for the *scaled* Hessian matrix is usually an indication that some of the parameters in the set has low individual sensitivities towards the cost function. Another possible cause of a high value for the condition number is linear dependence between some of the optimization variables. A way of resolving this could be to re-evaluate the combination of optimization variables (η). Another quantity calculated for the respective optimization variables is the identifiability ranking, for which an example is shown in Fig. 3.3. The identifiability ranking can be interpreted in somewhat of the same way as the condition number for the Hessian matrix, in that it indicates the sensitivity of the respective optimization variables on the cost function. In this consideration, however, the variables are ranked, and the user is able to better decide which variables to dismiss when re-choosing variables for optimization. A low identifiability ranking usually implies a low sensitivity to the cost function for that specific variable or a strong linear dependence to one of the other variables. In other words, the variables with the lowest identifiability rankings are the most sensible candidates for being dismissed from the optimization procedure. Both Figs. 3.2 & 3.3 originate from the parameter fitting procedure described in Sec. 4.3, but for the purpose of this section, they are included to portray the features of the ModelFit software. [21]

In App. F, the ModelFit software is demonstrated for a simplified model of a pendulum. In this case, the two parameters of the model are modified to yield virtually perfect agreement between the output predicted by the model and the reality as suggested by corresponding measurements. Because of the simplicity of the model, Eulers method for solving the equation system was chosen, using a short step-length, and the solution and simulation of the system proceeded without problems. The strategy for treating the main case of this project work (Sec. 3.3) is completely analogous to the example in App. F, apart from the fact that the CVODEs solver is chosen for the main case because of stiffness related issues for the system. In addition to this, the main case also presents with a significantly higher amount of both states and parameters than the simplified example. Because of this, it is expected that deciding variables for optimization, i.e. choosing η

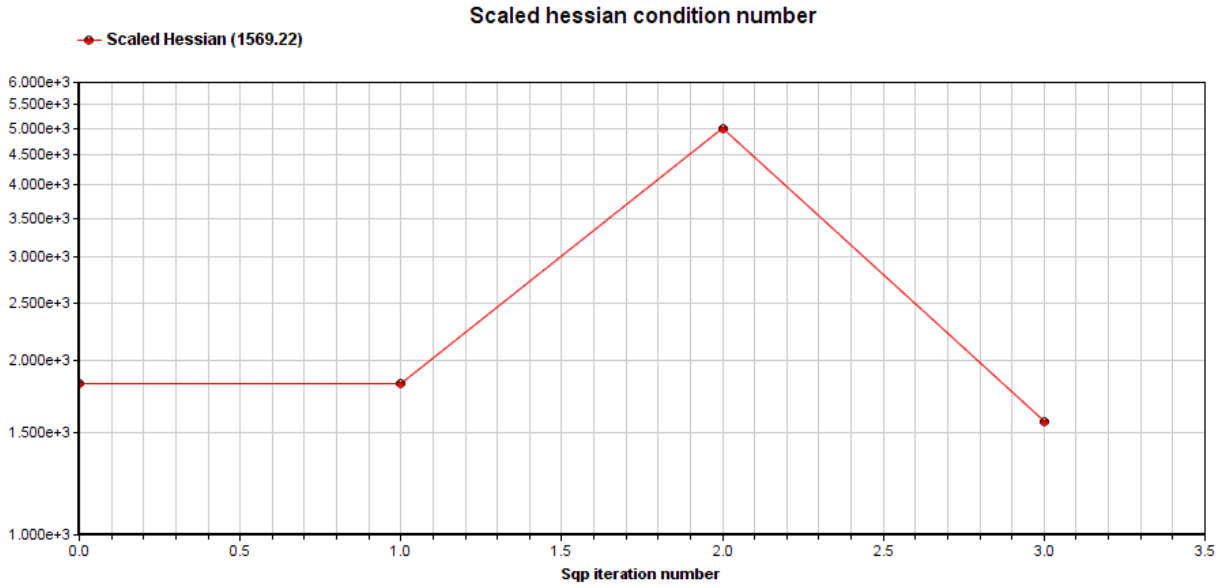


Figure 3.2: Changes in the condition number for the scaled Hessian matrix in the model fitting calculation.

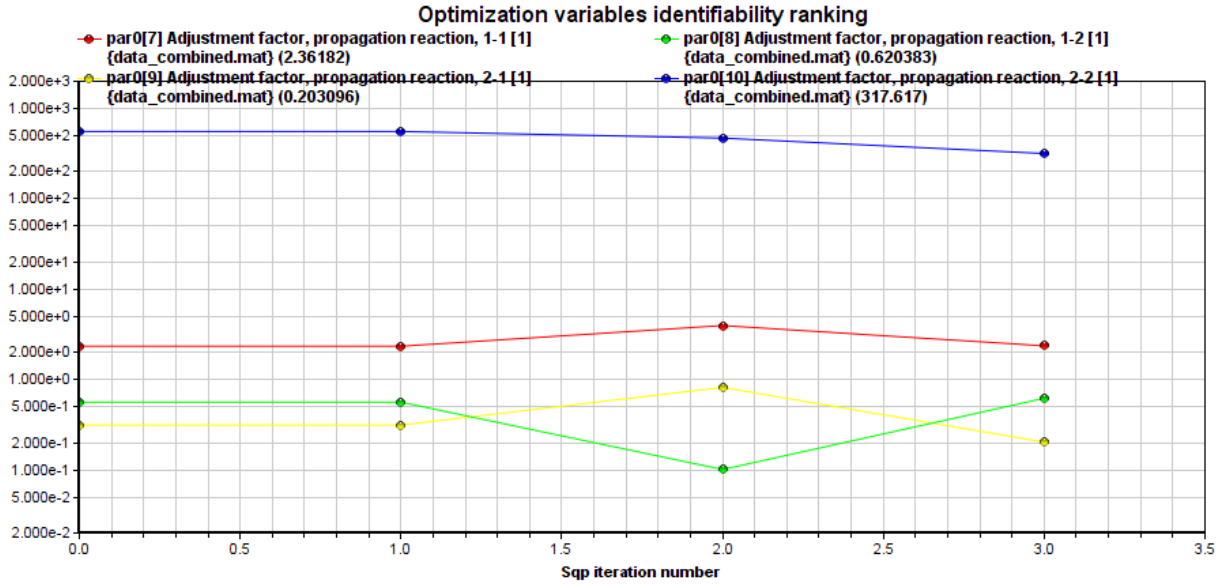


Figure 3.3: Changes in the identifiability ranking for the variables during model fitting calculation.

from θ, may be harder than for the pendulum example, for which the choice of optimization variables was obvious. Nevertheless, the general solution strategy remains the same, and the benefits of using the ModelFit tool are evident in both cases.

This concludes the short introduction to the Cybernetica ModelFit software, which will be utilized when performing parameter fitting for the emulsion copolymerization process described in Sec. 3.3. The results from the off-line parameter fitting are presented in Sec. 4.

3.3 Model description for the established model on emulsion copolymerization

The purpose of this section is to establish the remaining details regarding the reactor system treated in Sec. 4, referred to as the main case of the project work. The theoretical background for the established model is presented in Secs. 2.1 & 2.2, and this provides the basis for the modeling work. All balance equations, i.e. energy balances and species mass balances are constructed as indicated in Secs. 2.1 & 2.2, with reaction kinetics etc. as suggested. The model is formulated in the Modelica language using the Dymola tool, as introduced in Sec. 3.1.

The studied process is a semi-batch free-radical emulsion copolymerization process in which two different monomer types are combined to yield a copolymer product. The process is initiated using polymer seed particles and a chemical initiator compound. The tank reactor is assumed to be perfectly mixed due to agitation of the content. In this spirit, the system has been modeled without spatial gradients. This implies good internal mixing in each of the phases present in the reactor, aswell as rapid transfer of both heat and mass between the respective phases. From a modeling point of view, this will avoid the need for partial differential equations (PDEs) to describe the system, since ordinary differential equations (ODEs) with respect to time will suffice to describe the dynamic system. In the modeling work, the monomer droplet phase of the system is assumed to be negligible, and the gas phase of the system has not been considered in the modeling work as this is not believed to contribute significantly to the overall behavior of the reactor system. In this sense, the system is an entirely liquid system (emulsion), in which a polymer particle phase ("oil phase") and an aqueous phase ("water phase") are considered. The neglecting of the monomer droplet phase is in agreement with the semi-batch reactor strategy where reactants (monomer) are added slowly over the course of time for the batch, thus achieving so-called monomer starved conditions. The feeding to the reactor is illustrated in Fig. 3.4, where the feeding is split into three separate streams which are individually controllable. Feed stream 1 contains water and emulgator, feed stream 2 contains monomer (approximately 80/20 wt% M1/M2) and some CTA, while feed stream 3 contains initiator dissolved in water. The continuous feeding of reactants is aborted halfway through the course of the batch, for the remaining reactants to react completely.

The cooling mechanism for the reactor is a jacket covering most of the reactor wall. Direct heat loss from the reactor to the surrounding environment occurs at the spots which are not covered by the jacket, and these areas are located at the top and bottom of the reactor. The cooling jacket, which for this installation usually contains water with temperatures in the range 70-90 ^oC, will experience heat loss to the surroundings depending on the exposed surface area.

The main characteristics of the established model with respect to inputs and outputs are given in Tab. 3.1. The responsibility of a controller for such a reactor is to control both the feeding of reactants aswell as the amount of cooling fluid fed to the system in accordance with the controller objectives. Notice that the temperature of the cooling fluid may be either a disturbance to the system or an input, which is dependent on the specific plant for which the reactor model is applied. In some cases, the cooling fluid is given by an uncontrollable part of the process like an upstream

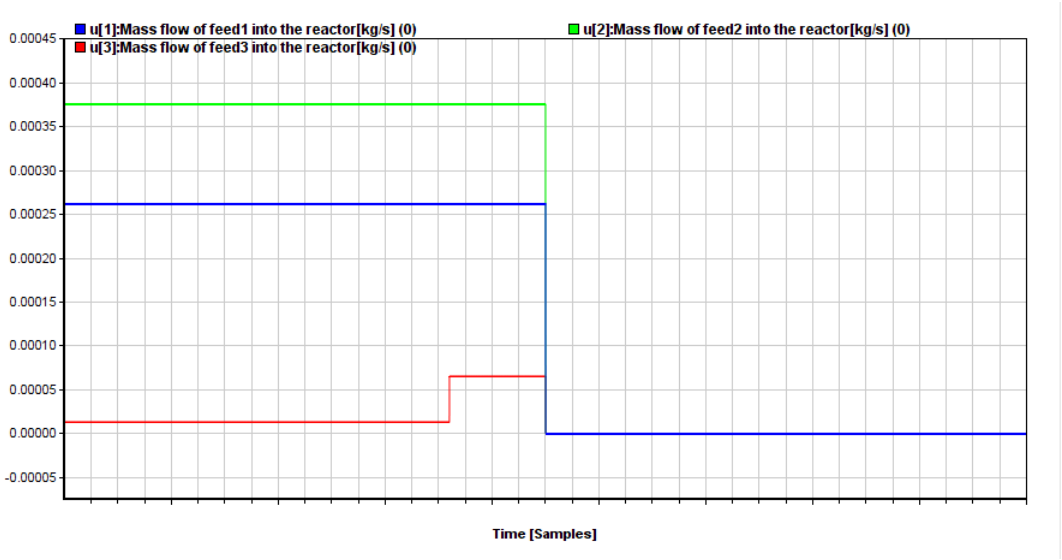


Figure 3.4: Illustration of the feeding to the reactor during the time of the batch.

unit or a cooling water reservoir. In other cases, the cooling fluid is the product of a supporting process built specifically for providing cooling fluid. In the latter case, where the temperature of the cooling fluid will be controllable, the controller will also add this quantity to the list of inputs in order to achieve the controller objectives. Notice also that the reactor temperature is represented by two separate temperatures, i.e. the temperatures of the respective phases in the system. This separation is a consequence of a choice made in the modeling work, i.e. to model the energy contents of the two phases separately. Because of good mixing in the reactor, the two temperatures will for all intents and purposes be equal, and will both pose a valid measure of the reactor temperature in overall. It is emphasized that although the main disturbances to the process are listed (Tab. 3.1), changes in the DVs are not considered in the simulations.

Table 3.1: Model characteristics for the established semi-batch case.

Important MVs	**Important CVs**	**Important DVs**
Mass flow, Feed stream 1	Temperature, reactor	Temperature, surroundings
Mass flow, Feed stream 2	Composition, particle phase	*Temperature, cooling fluid*
Mass flow, Feed stream 3	Composition, aqueous phase	Temperature, feed streams
Mass flow, cooling fluid	Degree of monomer conversion	Composition, feed streams
Temperature, cooling fluid	Polymer molecular weight	
	Polydispersity Index	

Throughout the text, the fact that the the model is constructed from first principles is emphasized, along with the fact that certain parameters are initially fictional. While this is largely true, some empirical correlations are deployed in the modeling, and most of the physical data for the system are acquired from the litterature. A brief collection of fluid properties for the system is presented in App. D. An important empirical part of the model is the properties for heat transfer between the contents of the reactor vessel, the cooling fluid in the cooling jacket and the surround-

ing environment. These models are developed for a small lab-scale tank reactor, where the thermal conductivities are modeled as a linear function of the total volume of the contents of the vessel, i.e. the sum of the volumes for the respective phases in the reactor. These correlations are shown in Eqs. 3.5 & 3.6, in which kA_{SR} and kA_{SJ} denote the thermal conductivity between the surrounding environment and the reactor vessel and the cooling jacket, respectively. Here, the volume of the vessel content (V) is given in litre (dm^3). It is emphasized that these correlations may not hold for reactors with a different geometric shape and different size, and this may raise the need for parameter fitting if the model is deployed for similar reactors with different cooling mechanisms. The experimental data used in the work (App. E) was acquired using a lab-scale reactor, hence justifying the choice of heat transfer properties for this case.

$$kA_{SR} = 0.018264V + 0.55210 \tag{3.5}$$

$$kA_{SJ} = 0.8591641V + 1.502800 \tag{3.6}$$

During the modeling work, effort was made to make the models receptive to parameter fitting/estimation. In this spirit, most internal processes such as chemical reactions, interphase mass transfer, etc. are formulated using parameters with adjustment/correction factors. Having established this, parameter estimation for an internal process can be performed by approaching the corresponding adjustment factor for the process rather than directly approaching the parameters themselves. In doing this, the original parameter values, which may be acquired from the literature, are kept intact. The parameters themselves may in turn be changed, but this should preferably be done after careful evaluation of the results from the parameter fitting procedure.

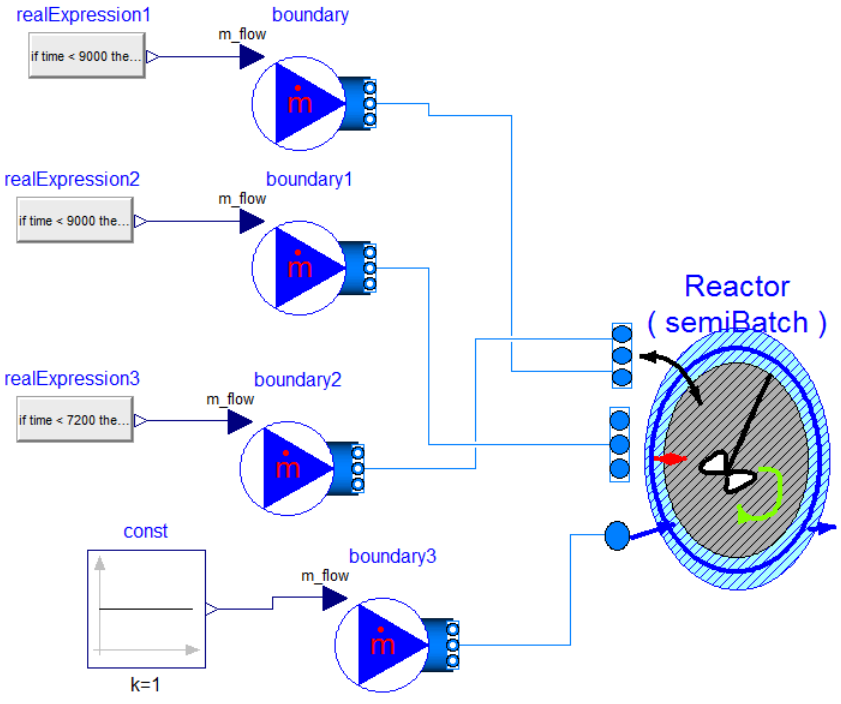


Figure 3.5: Graphical Dymola representation for a reactor test case.

The established case for the reactor system as it appears in the GUI of Dymola is shown in

Fig. 3.5. At this layer of the model, the reactant feeding and cooling to the system are shown together with the top layer of the semi-batch reactor model. The entire code for the model is not included in this report, mainly because the code is extensive despite the fact that the model is declared as a combination of various subunits. This strategy is inspired by the idea proposed in Fig. 3.1b in Sec. 3.1, and for the specific reactor case, the hierarchical structure becomes as illustrated in Fig. 3.6. The Reactor block in Fig 3.5 is, in other words, not in itself a complex unit with an extensive code attached to it, but rather a top layer governing the interconnection and communication between the various subunits of the model. These subunits can range from the cooling jacket model, the model for the reactor vessel itself, or the thermodynamics of the fluids involved, as indicated in Fig. 3.6.

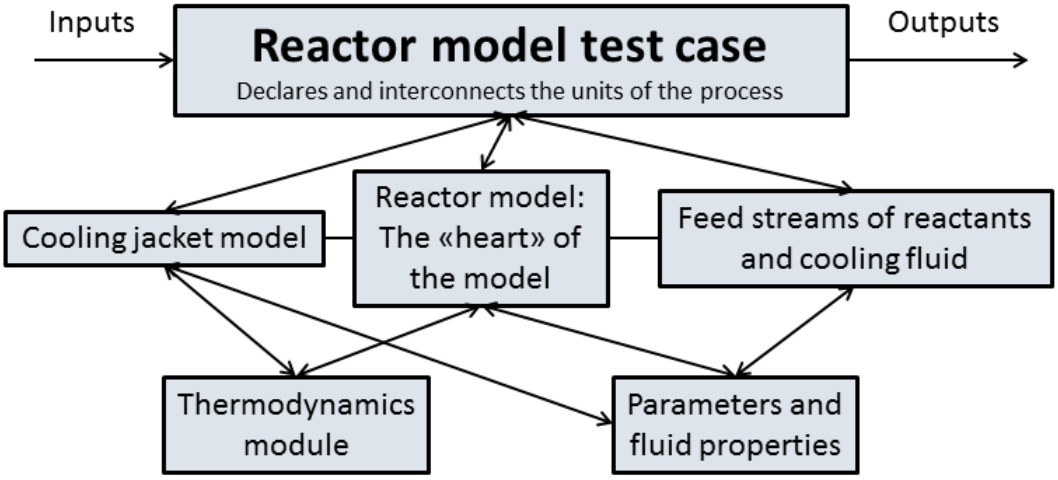


Figure 3.6: A figure showing the interconnection of Modelica units for the semi-batch copolymerization reactor system in a hierarchical manner.

An important aspect of the model is the reusability and the opportunity to modify the model in an easy way. The modular approach in designing subunits of the model (Fig. 3.6) is of key importance with respect to this. An additional feature of Modelica, which strengthens this strategy, is known as fluid package modeling. Instead of modeling each chemical species of the reactor system as an individual fluid, the respective phases of the system are assigned to a fluid package each. In this sense, the physical state, the thermodynamic state, the phase composition, etc. of the phases in the reactor are stored and accounted for by a fluid package, while the changes in the fluid are governed by the balance equations provided in the reactor model subunit, etc. The benefit of this strategy is the ability to change the components of the system, e.g. using different monomer species, by introducing a new fluid package. This would require only minor changes to the other subunits of the overall system. In this work, only one case has been considered, but to achieve adaptable models for long-term use, these features are valuable, and has hence been included in the modeling work.

This concludes the description of the semi-batch reactor system considered in this work. The treatment of this system with respect to parameter estimation is considered in Sec. 4.

4 Results from off-line parameter estimation

This section presents the results from the main task of the project work, which is to validate the established process model for semi-batch emulsion copolymerzation. This section starts off by investigating the behavior of the original model (as introduced in Sec. 3.3) in Sec. 4.1. Here, the purpose is to explore to what extent the model predictions deviate from the provided measurements, and identify the main reasons for the deviations. The experimental basis is a lab-scale test reactor, for which the experimental data is summarized in App. E. In Sec 4.2, these findings are used to explore how the model changes with respect to manual changes in certain parameter values. Based on the indications from the trial-and-error approach in Sec. 4.2, the Cybernetica ModelFit software (Sec. 3.2) has been used to perform optimal parameter fitting, as described in Sec. 4.3. Finally, a short summary evaluating the characteristics and performance of the model after the optimal parameter fitting is presented in Sec. 4.4.
Note: The simulations in this section are all given with masked time axes. The reason for this is that details regarding the batch time is confidential information under the COOPOL project. This modification to the simulations is not believed to pose a deterioration of the results and the conclusions of the work.

4.1 Model behavior before parameter fitting

As a starting point for discussing the validity of the developed models, the models are compared (in their original first principles form) to experimental data. In Fig. 4.1 the conversion of reactant is plotted for both the process model (red curve) and experimental lab-scale experiments[1] (blue curve). The conversion is here defined as indicated in Eq. 2.53 in Sec. 2.2. In Fig. 4.2, the temperature of the particle phase[2] in the reactor is shown for both model predictions (red curve) and measurements (blue curve). In addition, the development of the molecular weight of the copolymer product is shown in Fig. 4.3. For the molecular weight distribution, the experiments only give the terminal values, i.e. the values at the very end of the batch. From these preliminary results, it is evident that the established process model deviates somewhat from the truth as portayed by the experiments. The characteristics of the two curves for the conversion are similar, yet the actual values deviate to some extent. A summary of the most important differences between the model predictions and the measurements, as indicated by this initial consideration, are listed below.

1. The model prediction has a lower conversion than the experiments have. This effect is evident for most of the batch simulation time, especially at the end.

2. The experiments approach steady state (100% conversion) faster than the model prediction does. In fact, the model prediction does not reach 100% conversion at all during the simulation time.

3. The molecular weights show agreement with experimental data with respect to their relative ratio, i.e. the polydispersity index, but the respective values are significantly lower than the experiments suggest.

[1] The experimental data is found in the appendix, App. E

[2] In agreement with what was discussed about fluid package modeling in Sec. 3.3, the respective phases of the system will have individual temperatures. Since the reactor is assumed to be well agitated, the heat transfer between the phases is rapid, and hence the temperature of the phases will be virtually equal. The particle phase temperature is, in other words, a representative measure of the overall reactor temperature.

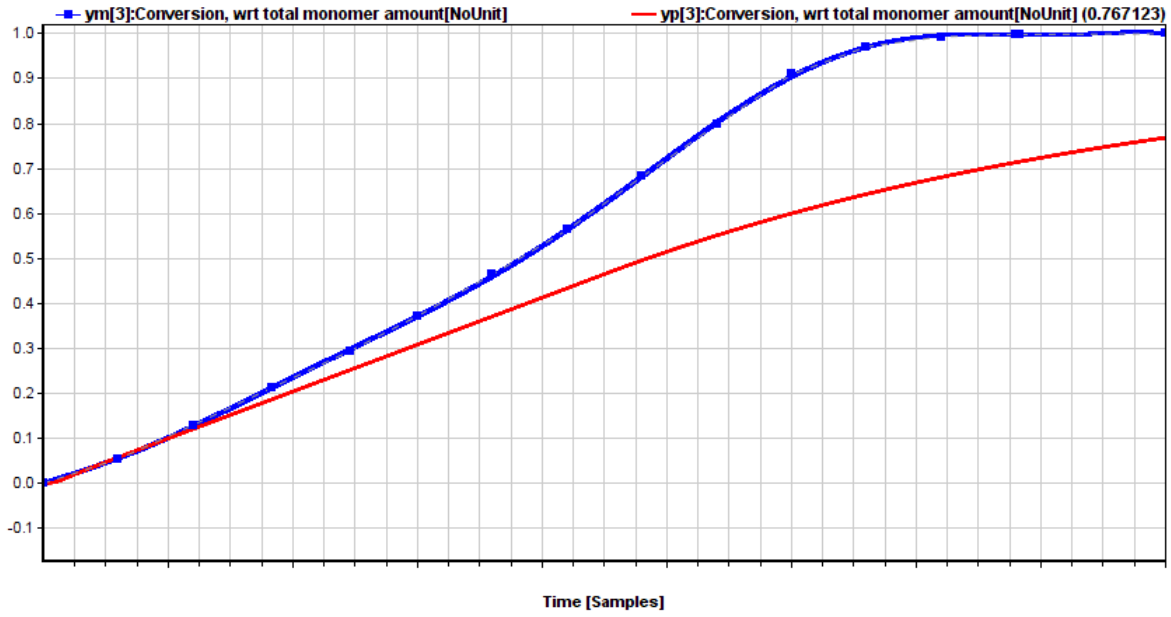


Figure 4.1: A plot showing the conversion of fed monomer to the reactor, before parameter fitting has been performed.

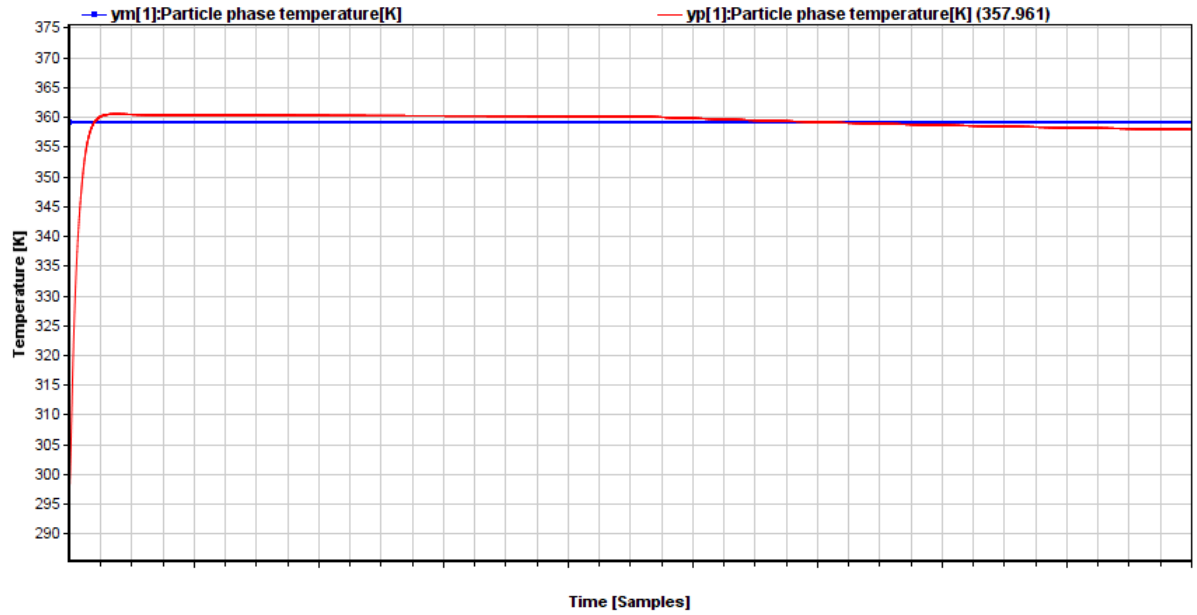


Figure 4.2: A plot showing the temperature of the particle phase in the reactor, before parameter fitting has been performed.

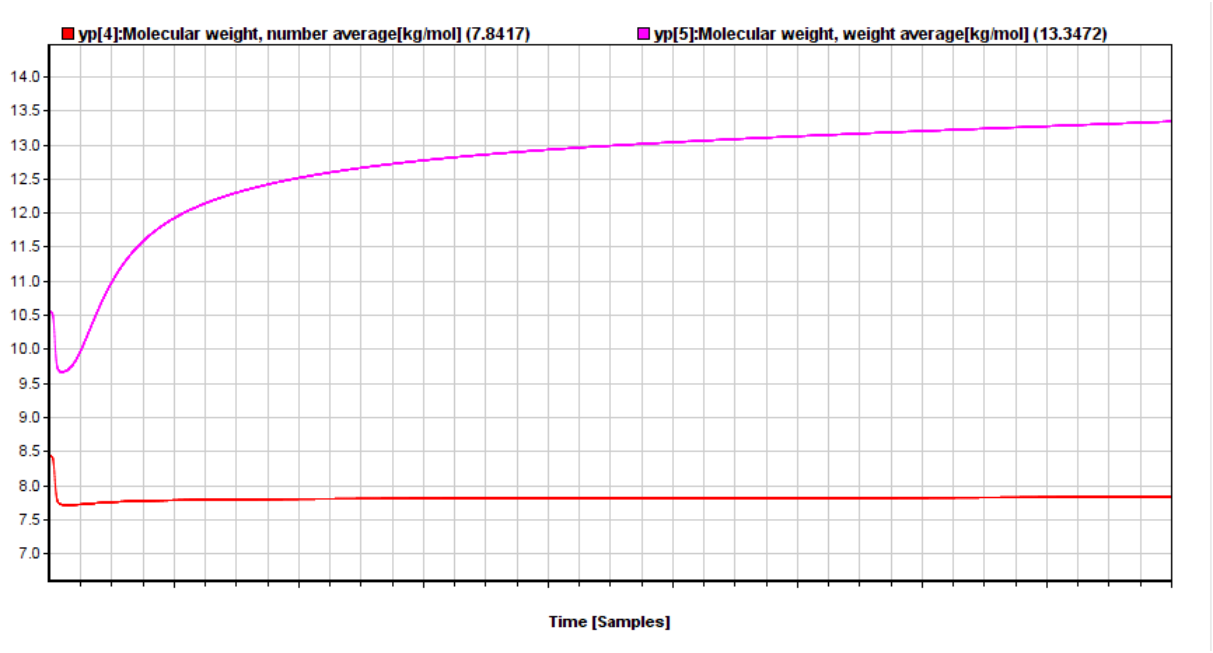


Figure 4.3: A plot showing the molecular weights (weight and number average) of the copolymer product, before parameter fitting has been performed.

The differences are, generally speaking, not tremendous, yet they are significant, and hence they serve as an indication that parameter estimation is needed for the model. As illustrated in the temperature plot (Fig. 4.2), the temperature agreement between the model prediction and the experiments is very good. From these initial considerations, the actions suggested for further treatment of the model parameters are listed below.

1. Investigate and adjust the various factors governing the propagation, for the model to show better agreement with the consumption of monomer during the course of the batch time. It is expected that the kinetic factors should be higher.

2. Investigate (and possibly adjust) factor(s) governing termination, in order to achieve close to 100% conversion at the end of the batch time, and thus have better agreement with experiments. Recalling how molecular weights are defined in Sec. 2.1 using polymer moments from App. B, changing the parameters governing termination will also affect the molecular weights, which are important quantities for the product quality.

3. Adjust the heat conduction constant between the reactor vessel and the cooling jacket in order to get even better agreement for the temperatures. Other factors governing the temperature profile of the reactor could also be investigated, but this is not first priority in validating the model, since the model gives good agreement with the data when it comes to the temperatures.

These suggestions for change are considered in the following sections. It is emphasized that because of the complexity of the system, the various parameters may be interconnected in such a manner that the overall effect of changing each of them is non-trivial. The overall effect of changing a combination of parameters is usually even harder to predict using intuition.

4.2 Introductory case: Manual parameter fitting

The purpose of this section is to identify and explore how the model changes when some of the key parameters are changed. For the purpose of this section, the changing of the parameters has been performed manually in a trial-and-error manner, simply to identify the qualitative behavior of the model relative to the parameters. The optimal approach to parameter fitting is left for the section following this one (Sec. 4.3).

In Fig. 4.4, the conversion of reactant is plotted for the model after the kinetic reaction rate constants have been adjusted in a trial-and-error manner. As before, the blue curve represents the measurements while the red curve shows model output. The motivation for this change is to get higher conversion of monomer throughout the batch. The conversion does indeed increase when this change is applied, and close to 100% conversion is achieved at the end of the batch time. The conversion is, however, slightly too high to agree with the experimental data during the first half of the simulation, and simply adjusting the kinetic factors does not appear to be a sufficient solution in itself. In Fig. 4.5, the molecular weight distributions for the copolymer product is shown again, for termination parameters fitted in a trial-and-error manner. The characteristic shape of the curves as well as the ratio between the values, i.e. the polydispersity index as defined in Eq. 2.43, remains approximately the same, which is desirable[3]. In this sense, the model output for molecular weight distribution has not changed much, but when it comes to the respective values for the molecular weights, the model predictions now agree with the experiments in a much more satisfying way.

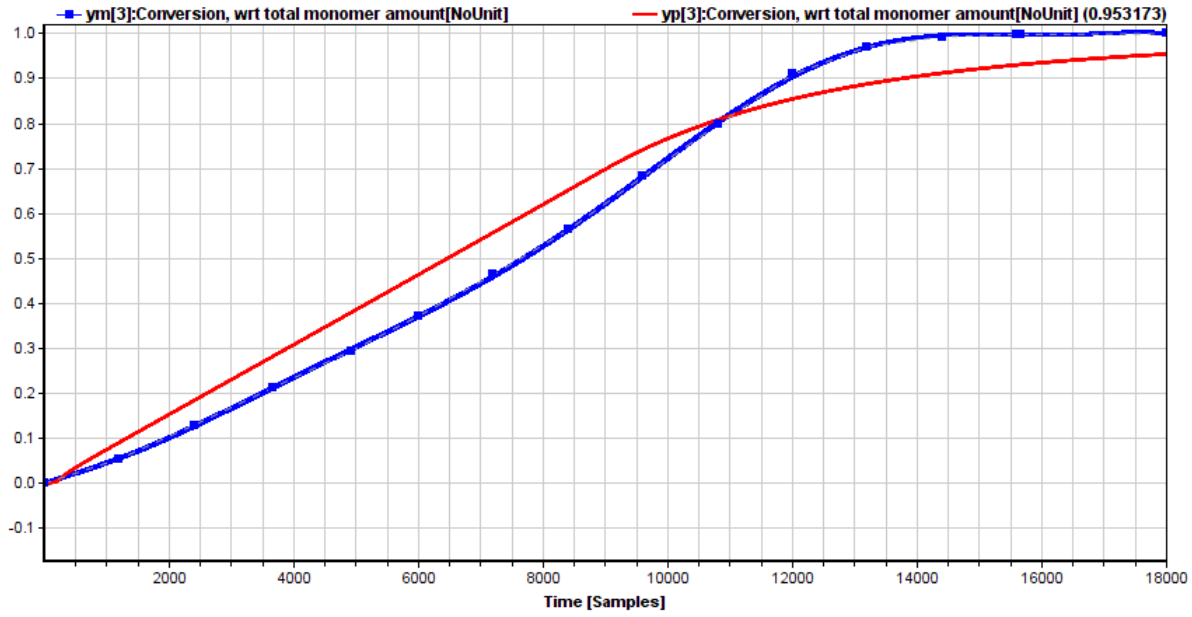


Figure 4.4: Illustration of monomer conversion with kinetic reaction rate constants adjusted in a trial-and-error manner.

[3]A polydispersity index (Eq. 2.43) of just over 2 is expected for this reactor case, because a low degree of chain branching is expected.

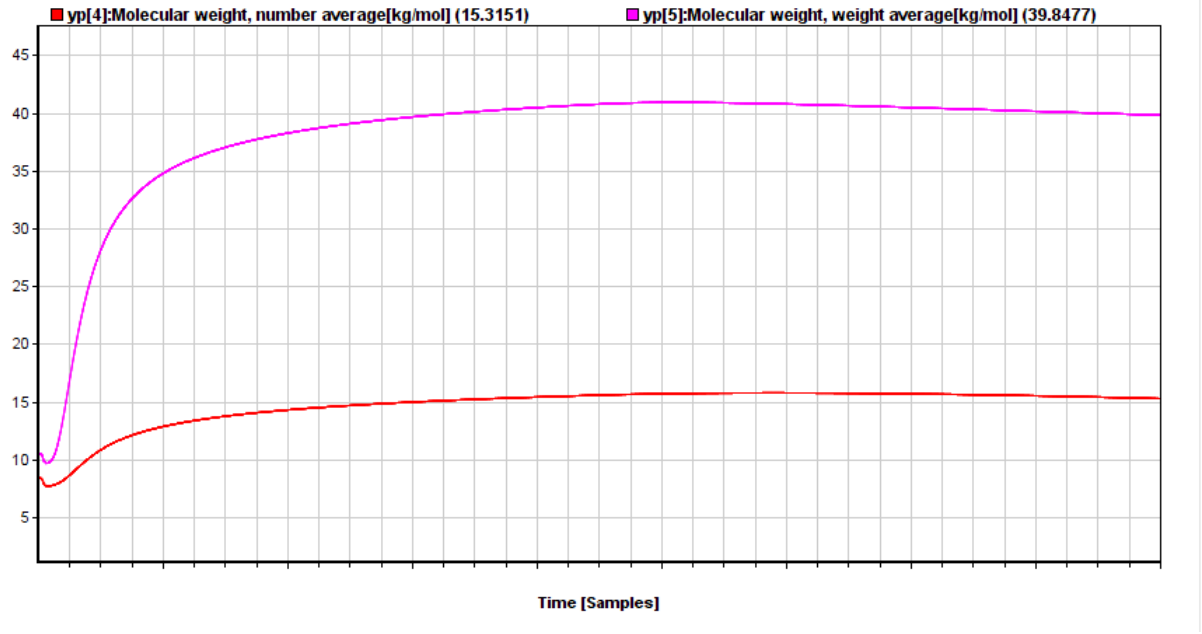


Figure 4.5: A plot showing development of copolymer molecular weight, after parameters governing termination has been adjusted in a trial-and-error manner.

Generally speaking, there are still deviations between the model predictions and the measurements, and this fact motivates the strive for optimally fitted parameters. It is important to point out that the optimal solution is not necessarily expected to yield perfect agreement between the model predictions and the experimental measurements, but rather give the best possible result given the chosen set of parameters for optimization. The actual changes made for the parameters in the trial-and-error approach are presented in Tab. 4.1. This serves as a starting point for the optimal analysis of the parameters, which will be described in Sec. 4.3.

Table 4.1: Parameter changes for manual trial-and-error model fitting.

Parameter	**Original value**		**New value**
Adjustment factor, 1-1 propagation rate constant	50	→	20
Adjustment factor, 1-2 propagation rate constant	22	→	8
Adjustment factor, 2-1 propagation rate constant	50	→	20
Adjustment factor, 2-2 propagation rate constant	22	→	8
Adjustment factor, monomer 1 transfer to CTA	20	→	55
Adjustment factor, monomer 2 transfer to CTA	20	→	55

4.3 Optimal parameter fitting

In this section, the purpose is to further explore the adjustment factors for the system that was investigated in Sec. 4.2. In addition, several other parameters are chosen for optimization. The

vector of parameters chosen for optimization (η) contains 11 parameters related to propagation, termination or interphase mass transfer. The respective parameters, with optimal changes, are shown in Tab. 4.2. For solving the optimization problems involved, the Cybernetica ModelFit software (as introduced in Sec. 3.2) has been utilized, in agreement with the theoretical background established in Sec. 2.3.

The resulting curve for the conversion of reactant after the optimization of the parameters is shown in Fig. 4.6. As before, the red curve indicates the model output. When being compared to the result from the trial-and-error approach, as presented in Fig. 4.4, the differences are small. The results are similar in character, although the respective values of the parameters differ between the two cases. The optimal case represents the solution with the lowest sum of deviations, mathematically speaking, and this is the solution of highest interest. It is emphasized that the trial-and-error solution, although tedious, did not prove to be terrible compared to the optimal solution. An interesting observation is that the trial-and-error approach (Fig. 4.4) actually yields

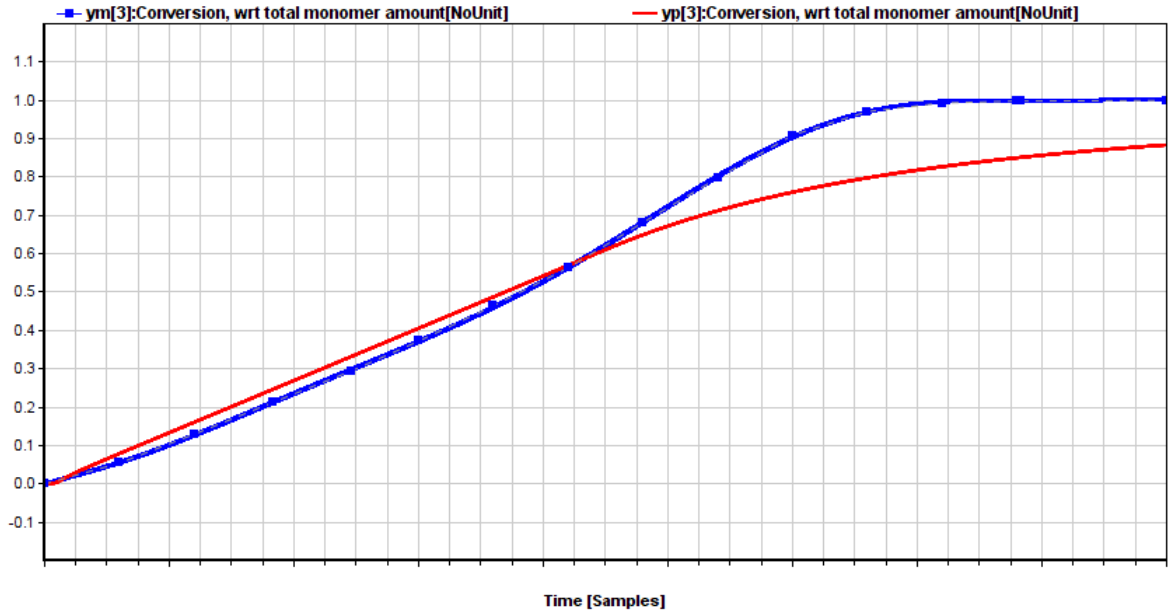


Figure 4.6: Figure showing monomer conversion, using optimally fitted parameters.

a higher conversion of monomer at the end of the batch than the optimal approach (Fig. 4.6) does. From this it can be interpreted that the initial difference (in the first half of the batch time) between the conversion plots as suggested by the trial-and-error solution is larger than the difference at the end of the batch time suggested by the optimal solution. It may, however, appear counter-intuitive and undesirable that the optimal solution has achieved lower conversion of monomer than the trial-and-error solution at the end of the batch. A way to work around this, if desired, would be to weigh the deviations at the end of the batch time heavier, with respect to the cost function, than the deviations at the start of the batch time. In such an event, the deviations at the end of the batch would be penalized more than the deviations earlier in the batch, and the solution to the optimization problem would change accordingly, "trying harder" to minimalize the deviations

at the end than the deviations in the start. This strategy has not been deployed in this work, however, and all measurement throughout the batch are considered equally important.

The curves for the molecular weight distributions of the system are shown in Fig. 4.7. By comparison with the trial-and-error solution (Fig. 4.5), the two results are virtually the same, which is yet another good indication for the trial-and-error approach. The optimal solution does, however, represent the mathematically favorable and thus most interesting solution. It is emphasized that although the two approaches yield a similar output with respect to the molecular weight distributions, they have different combinations of parameter values. In Fig. 4.7, the experimental values

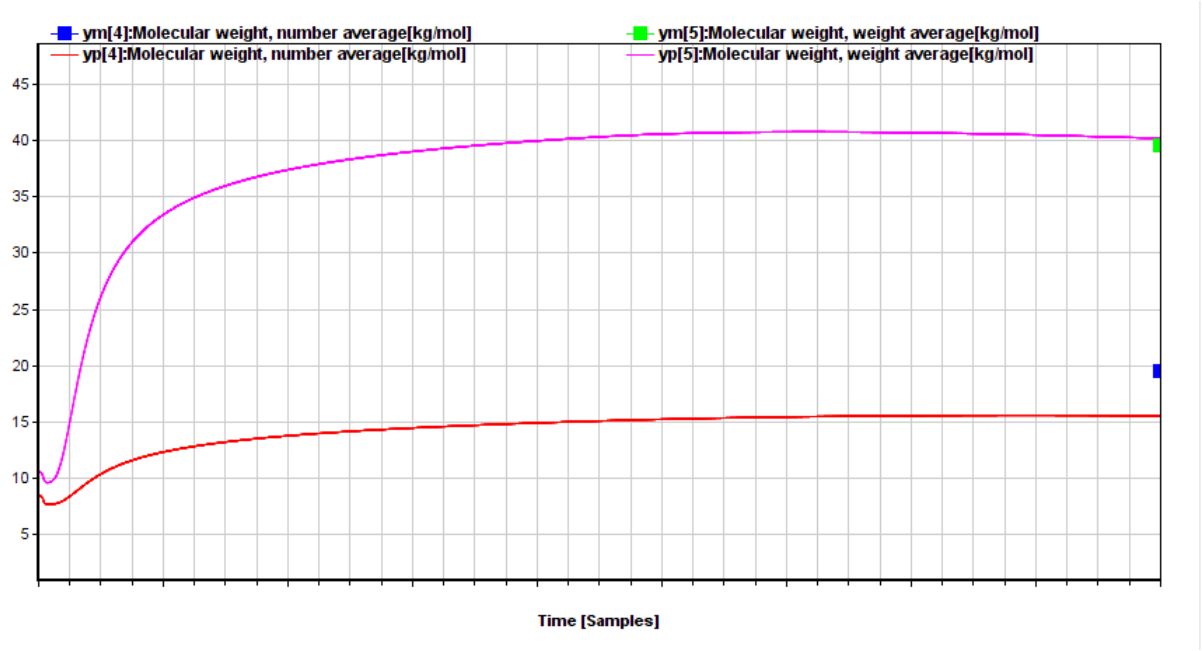


Figure 4.7: A plot showing copolymer molecular weights, using optimally fitted parameters.

for the molecular weights at the end of the batch are indicated as blue and green squares for the number and weight average molecular weight (Eqs. 2.41 & 2.42), respectively, to better display the agreement between the model output and the experimental measurements. The fact that the molecular weight distribution is only measured at the very end of the batch makes it difficult to say anything about the characteristics of the development of the molecular weights throughout the batch. In a case where the molecular weight distribution was measured several times throughout the batch, this would allow for a more thorough examination of the results, and perhaps even better values for the fitted parameters.

The changes made to the respective parameters from the model chosen for optimization are shown in Tab. 4.2.

4.4 Summary: Model validity after optimal parameter fitting

This section is added to provide a brief summary of the achievements of the parameter fitting procedure. The most important changes are listed below, and the specific changes made in the selected parameters are presented in Tab. 4.2.

Table 4.2: Parameter changes for optimal model fitting.

Parameter	**Original value**		**New value**
Adjustment factor, 1-1 propagation rate constant	50	→	150.0
Adjustment factor, 1-2 propagation rate constant	22	→	18.5673
Adjustment factor, 2-1 propagation rate constant	50	→	68.3477
Adjustment factor, 2-2 propagation rate constant	22	→	46.6841
Common adj. factor for propagation	1	→	2.6914
Common adj. factor for monomer transfer to CTA	1	→	2.975
Adjustment factor, termination by recombination	800	→	1166.80
Adjustment factor, termination by disproportiation	800	→	400.0
Interphase mass transfer coefficient, monomer 1	100 000	→	102 264.93
Interphase mass transfer coefficient, monomer 2	100 000	→	72 383.51
Frac. between term. by rec. and total term.	1	→	0.8996

- The factors governing the rate of the propagation reactions have been modified for all four possible propagation reactions. In addition, a common adjustment factor has been introduced for all four reactions. The results contribute to give better agreement between the model output and the measurements for reactant conversion.

- The factors governing termination have been adjusted. Instead of adjusting the individual factors for monomer transfer to CTA, the two reactions have been combined in a common adjustment factor. In addition, the respective adjustment factors for termination by recombination and disproportiation has been optimized. The fraction between termination by recombination has also been subject to optimization, that is the percentage of the total chain termination accounted for by recombination, in agreement with the formulations in Sec. 2.1. This has led to better agreement between the model and the measurements with respect to the molecular weight distributions.

- The softening coefficient ("fictional time constant") for species mass transfer between the phases of the system has been optimized for both monomer types. In this case, increasing the coefficient will yield a slower mass transfer.

In total, the improved model presents with good agreement between the measurements and the model outputs. The model presents with significant demands for computational power, and this may pose a threat to the on-line use of the model in a complete controller implementation, but the parameter fitting itself is considered successful with respect to deploying a model-based predictive controller.

5 Conclusions

The theoretical concepts for free-radical emulsion copolymerization have been established. This has been utilized to formulate a model in the programming language Modelica for a semi-batch case on copolymerization of two different monomer types. The model was made largely from first principles, and has been subject to off-line parameter estimation to fit the model to experimental data from a lab-scale reactor. The experimental reactor operates the same chemical system under the same conditions, thus representing a corresponding real-life case to the established model.

For the purpose of off-line parameter estimation for the model, necessary theoretical background was explored, and software tools such as the Cybernetica ModelFit software was introduced. The model was initially treated by modifying certain parameters in a trial-and-error manner to investige the agreement between model output and experimental measurements. The ModelFit software was then successfully utilized to perform off-line parameter fitting for the established model, adjusting 11 of the parameters of the model to yield an optimal solution.

Trial-and-error adjustment of parameters is inaccurate and tedious in comparison to optimal adjusting, which is a systematic way to achieve decent parameter values. The results of the work show, however, that the improvement from the trial-and-error solution to the optimal solution is small when just a small selection of parameters is subject to modification. It is emphasized that the optimal solutions are the desirable ones, although the trial-and-error solutions show good agreement with the optimal solutions.

The computational time involved in solving optimization problems like the ones encountered in the off-line parameter estimation of this work is definitely a concern to be taken into consideration. The computational effort is very important, in particular, when dealing with on-line systems and control applications. The calculations in this work was performed off-line, yet they still serve to indicate the demand for computational effort for the model. The complexity of the process model was evident when performing SQP iterations to solve the parameter estimation problem, in which scenarios with up to several minutes per iteration was experienced. The model used in this work may, in other words, need improvements with respect to the numerical efficiency if the model is to be used for an on-line implementation. For the suggested extension to this work, which is the design of a nonlinear model-based predictive controller, this is exactly the case.

The theory of on-line estimation methods was explored, and this provides a basis for the development of estimator algorithms in the proposed extension to this work, where a complete controller implementation is considered.

References

[1] PlasticsEurope, Plastics - the Facts 2012, An analysis of European Platics production, demand and waste data for 2011, Belgium, 2012

[2] P. C. Painter & M. M. Coleman, Fundamentals of Polymer Science, 2^{nd} Ed., CRC Press, 1997

[3] I. K. Khairullin, Adhesive-Melts - The Most Dynamically Developing Area in World Production and Consumption of Adhesives, Pol. Sc. Series D, Glues and Sealing Materials, 2013, Vol. 6, No. 1

[4] H. Ghodke, S. Raman & B. E. Ydstie, Modeling and Control of Free Radical Co-Polymerization, Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, USA

[5] A. H. Helgesen, Toolbox for generation of nonlinear control models for semi-batch emulsion polymerization reactors, Master thesis, Department of Chemical Engineering, Norwegian University of Science and Technology, 2011

[6] A. Nyström, Modeling and Simulation of a Multi Phase Semi-batch Reactor, Master thesis, Department of Mathematical Sciences, Chalmers University of Technology & Göteborg University, Gothenburg, Sweden, 2007

[7] H. S. Fogler & N. Gurmen, University of Michigan, Note on modeling of semi-batch reactors, 2007, URL: `http://www.umich.edu/~essen/html/06chap/html/prs_cstr.htm`

[8] B. Li & B. W. Brooks, Prediction of the Average Number of Radicals per Particle for Emulsion Polymerization, J. Pol. Sc., Vol. 31, Iss. 9, Aug. 1993

[9] A. C. Hindmarsh & R. Serban, User Documentation for cvodes v2.7.0, March 2012, Center for Applied Scientific Computing, Lawrence Livermore National Laboratory
URL: `http://computation.llnl.gov/casc/sundials/documentation/cvs_guide.pdf`

[10] H. Olsson, H. Elmqvist & M. Otter, Modelica - A Unified Object-Oriented Language for Systems Modeling, Language Specification, Version 3.3, May 9, 2012
URL: `https://www.modelica.org/documents/ModelicaSpec33.pdf`

[11] J. Nocedal & S. J. Wright, Numerical Optimization, 2^{nd} Ed., Springer, 2006

[12] T. S. Schei, On-line estimation for process control and optimization applications, 8^{th} International IFAC Symposium on Dynamics and Control of Process Systems, Preprints Vol. 2, June 6-8, 2007

[13] T. J. Crowley & K. Y. Choi, Calculation of Molecular Weight Distribution from Molecular Weight Moments in Free Radical Polymerization, Ind. Eng. Chem. Res. 1997, 36, 1419-1423

[14] J. Gao & A. Penlidis, Mathematical modeling and computer simulator/database for emulsion polymerizations, Prog. Polym. Sci. 27 (2002) 403-535, Elsevier

[15] H. A. Jakobsen, Chemical Reactor Modeling: Multiphase Reactive Flows, Springer, 2008

[16] C. E. Wyman, Polymer Moment Equations for Distributed Parameter Systems, AlChE J., Vol. 21, No. 2, 1975

[17] J.B. Rawlings & D. Q. Mayne, Model Predictive Control: Theory and Design, 2013, Nob Hill Publishing, LLC

[18] D. Simon, Optimal State Estimation: Kalman, H_∞ and Nonlinear approaches, 1^{st} Ed., John Wiley & Sons, 2006

[19] E. O. Kreyszig, Advanced Engineering Mathematics, 9^{th} Ed., Wiley, 2005

[20] R. E. Walpole, R. H. Myers, S. L. Myers & K. Ye, Probability and Statistics for Engineers and Scientists, 8^{th} Ed., Pearson Education, 2007

[21] Cybernetica ModelFit User Manual, v. 1.20, Cybernetica AS

[22] Deliverable 4.6: Control model for industrial scale semi-batch polymerisation reactor, internal report of the COOPOL project

A Additional information for radical species modeling

When applying a full population balance to describe the radicals of the copolymerization system, as introduced in Sec. 2.1, the calculations use a system of equations involving a so-called A-matrix. The purpose of this matrix is to govern the respective changes in particles carrying between 0 and N radicals. The A-matrix is constructed as indicated in the MATLAB script[1] in Fig. A.1.

In this script, sigma (σ), k and C represent the corresponding quantities from Sec. 2.1, while N denotes the maximum number of radicals per particle considered in the approach.

```
function [ dn ] = ndot_full( t, n )
% ndot_full
%   function to calculate the time derivative of relative
%   radical frequency vector for emulsion polymerization

Li_parameters;
N = max(size(n))-1;       % Max. number of radicals per particle

A = [ -sigma k 2*C 0 zeros(1,N-3 );
    sigma -(sigma + k ) 2*k 6*C zeros(1,N-3 );
    zeros(  N-1, N+1 ) ];

for i = 2:N,
    A(i+1,i)   =  sigma;
    A(i+1,i+1) = - sigma - i * k - i * (i-1) * C;
    if i < N,
        A(i+1,i+2) = (i+1) * k;
    end
    if i < N-1,
        A(i+1,i+3) = (i+2)*(i+1) * C;
    end
end

dn = A * n ;
end
```

Figure A.1: MATLAB code for generating the A-matrix used in the full population balance, used for radical species modeling in Sec. 2.1.

[1] This script is printed with the permission of P. Singstad, Cybernetica AS.

B Polymer moments for copolymer product calculations

This section presents the polymer moments used in the calculations for product quality. The polymer moments were introduced in Sec. 2.1 for calculating the molecular weight distribution for the copolymer product, as described by Crowley and Choi [13]. The use of moments for describing the molecular weight distribution is also described by Gao and Penlidis. [14]

The definition of the polymer moments is given in Eqs. B.1 - B.3. Here, i denotes the order of the moment, while $[P_j(n)]$ denotes the concentration of living polymer med endgroup j, having a length of n units, etc.

$$\mu_i^{P1} = \sum_{n=1}^{\infty} n^i \left[P_1(n)\right] \tag{B.1}$$

$$\mu_i^{P2} = \sum_{n=1}^{\infty} n^i \left[P_2(n)\right] \tag{B.2}$$

$$\mu_i^{D} = \sum_{n=1}^{\infty} n^i \left[D(n)\right] \tag{B.3}$$

Zeroth, first and second order moments, with respect to living chains with type 1 endgroup:

$$\begin{aligned}\frac{d\mu_0^{P1}}{dt} &= \frac{[M_1]}{[M_1]+[M_2]}\frac{2f_I k_I n_I}{V_p} - k_{p12}[M_2]\mu_0^{P1} + k_{p21}[M_1]\mu_0^{P2} \\ &\quad - k_{f12}[M_2]\mu_0^{P1} + k_{f21}[M_1]\mu_0^{P2} - \phi\left(k_{tc}+k_{td}\right)\left(\left(\mu_0^{P1}\right)^2 + \mu_0^{P1}\mu_0^{P2}\right)\end{aligned} \tag{B.4}$$

$$\begin{aligned}\frac{d\mu_1^{P1}}{dt} &= \frac{[M_1]}{[M_1]+[M_2]}\frac{2f_I k_I n_I}{V_p} + k_{p11}[M_1]\mu_0^{P1} + k_{p21}[M_1]\left(\mu_0^{P2}+\mu_1^{P2}\right) \\ &\quad - k_{p12}[M_2]\mu_1^{P1} - k_{f,CTA,1}[CTA]\left(\mu_1^{P1}-\mu_0^{P1}\right) - k_{f11}[M_1]\left(\mu_1^{P1}-\mu_0^{P1}\right) \\ &\quad + k_{f21}[M_1]\mu_0^{P2} - k_{f12}[M_2]\mu_1^{P1} - \phi\left(k_{tc}+k_{td}\right)\left(\mu_0^{P1}\mu_1^{P1}+\mu_1^{P1}\mu_0^{P2}\right)\end{aligned} \tag{B.5}$$

$$\begin{aligned}\frac{d\mu_2^{P1}}{dt} &= \frac{[M_1]}{[M_1]+[M_2]}\frac{2f_I k_I n_I}{V_p} + k_{p11}[M_1]\left(\mu_0^{P1}+2\mu_1^{P1}\right) \\ &\quad + k_{p21}[M_1]\left(\mu_0^{P2}+2\mu_1^{P2}+\mu_2^{P2}\right) - k_{p12}[M_2]\mu_2^{P1} \\ &\quad - k_{f,CTA,1}[CTA]\left(\mu_2^{P1}-\mu_0^{P1}\right) - k_{f11}[M_1]\left(\mu_2^{P1}-\mu_0^{P1}\right) + k_{f21}[M_1]\mu_0^{P2} \\ &\quad - k_{f12}[M_2]\mu_2^{P1} - \phi\left(k_{tc}+k_{td}\right)\left(\mu_0^{P1}\mu_2^{P1}+\mu_2^{P1}\mu_0^{P2}\right)\end{aligned} \tag{B.6}$$

Zeroth, first and second order moments, with respect to living chains with type 2 endgroup:

$$\begin{aligned}\frac{d\mu_0^{P2}}{dt} &= \frac{[M_2]}{[M_1]+[M_2]}\frac{2f_I k_I n_I}{V_p} + k_{p12}[M_2]\mu_0^{P1} - k_{p21}[M_1]\mu_0^{P2} \\ &\quad + k_{f12}[M_2]\mu_0^{P1} - k_{f21}[M_1]\mu_0^{P2} - \phi\left(k_{tc}+k_{td}\right)\left(\left(\mu_0^{P2}\right)^2 + \mu_0^{P1}\mu_0^{P2}\right) \qquad \text{(B.7)} \\ \frac{d\mu_1^{P2}}{dt} &= \frac{[M_2]}{[M_1]+[M_2]}\frac{2f_I k_I n_I}{V_p} + k_{p22}[M_2]\mu_0^{P2} - k_{p21}[M_1]\mu_1^{P2} + k_{p12}[M_2]\left(\mu_0^{P1}+\mu_1^{P1}\right) \\ &\quad - k_{f,CTA,2}[CTA]\left(\mu_1^{P2}-\mu_0^{P2}\right) - k_{f22}[M_2]\left(\mu_1^{P2}-\mu_0^{P2}\right) \\ &\quad - k_{f21}[M_1]\mu_1^{P2} + k_{f12}[M_2]\mu_0^{P1} - \phi\left(k_{tc}+k_{td}\right)\left(\mu_0^{P1}\mu_1^{P2} + \mu_1^{P2}\mu_0^{P2}\right) \qquad \text{(B.8)} \\ \frac{d\mu_2^{P2}}{dt} &= \frac{[M_2]}{[M_1]+[M_2]}\frac{2f_I k_I n_I}{V_p} + k_{p22}[M_2]\left(\mu_0^{P2}+2\mu_1^{P2}\right) - k_{p21}[M_1]\mu_2^{P2} \\ &\quad + k_{p12}[M_2]\left(\mu_0^{P1}+2\mu_1^{P1}+\mu_2^{P1}\right) - k_{f,CTA,2}[CTA]\left(\mu_2^{P2}-\mu_0^{P2}\right) \\ &\quad - k_{f22}[M_2]\left(\mu_2^{P2}-\mu_0^{P2}\right) - k_{f21}[M_1]\mu_2^{P2} + k_{f12}[M_2]\mu_0^{P1} \\ &\quad - \phi\left(k_{tc}+k_{td}\right)\left(\mu_0^{P1}\mu_2^{P2} + \mu_2^{P2}\mu_0^{P2}\right) \qquad \text{(B.9)}\end{aligned}$$

Zeroth, first and second order moments, with respect to dead chains:

$$\begin{aligned}\frac{d\mu_0^{D}}{dt} &= k_{f,CTA,1}[CTA]\mu_0^{P1} + k_{f,CTA,2}[CTA]\mu_0^{P2} + k_{f11}[M_1]\mu_0^{P1} \\ &\quad + k_{f22}[M_2]\mu_0^{P2} + k_{f12}[M_2]\mu_0^{P1} + k_{f21}[M_1]\mu_0^{P2} \\ &\quad + \phi k_{tc}\left(\frac{1}{2}\mu_0^{P1}\mu_0^{P1} + \mu_0^{P2}\mu_0^{P1} + \frac{1}{2}\mu_0^{P2}\mu_0^{P2}\right) \\ &\quad + \phi k_{td}\left(\mu_0^{P1}\mu_0^{P1} + 2\mu_0^{P2}\mu_0^{P1} + \mu_0^{P2}\mu_0^{P2}\right) \qquad \text{(B.10)} \\ \frac{d\mu_1^{D}}{dt} &= k_{f,CTA,1}[CTA]\mu_1^{P1} + k_{f,CTA,2}[CTA]\mu_1^{P2} + k_{f11}[M_1]\mu_1^{P1} \\ &\quad + k_{f22}[M_2]\mu_1^{P2} + k_{f12}[M_2]\mu_1^{P1} + k_{f21}[M_1]\mu_1^{P2} \\ &\quad + \phi\left(k_{tc}+k_{td}\right)\left(\mu_0^{P1}\mu_1^{P1} + \mu_0^{P2}\mu_1^{P1} + \mu_1^{P2}\mu_0^{P1} + \mu_0^{P2}\mu_1^{P2}\right) \qquad \text{(B.11)} \\ \frac{d\mu_2^{D}}{dt} &= k_{f,CTA,1}[CTA]\mu_2^{P1} + k_{f,CTA,2}[CTA]\mu_2^{P2} + k_{f11}[M_1]\mu_2^{P1} \\ &\quad + k_{f22}[M_2]\mu_2^{P2} + k_{f12}[M_2]\mu_2^{P1} + k_{f21}[M_1]\mu_2^{P2} \\ &\quad + \phi k_{tc}\left(\mu_0^{P1}\mu_2^{P1} + \mu_1^{P1}\mu_1^{P1} + \mu_0^{P2}\mu_2^{P1} + 2\mu_1^{P2}\mu_1^{P1} + \mu_2^{P2}\mu_0^{P1} + \mu_0^{P2}\mu_2^{P2} + \mu_1^{P2}\mu_1^{P2}\right) \\ &\quad + \phi k_{td}\left(\mu_2^{P1}\mu_0^{P1} + \mu_0^{P2}\mu_2^{P1} + \mu_2^{P2}\mu_0^{P1} + \mu_2^{P2}\mu_0^{P2}\right) \qquad \text{(B.12)}\end{aligned}$$

This system of differential equations provides a significant contribution to the total number of states for the model. This acts to increase the demand for computational power. The initialization of these states could also be subject to estimation, in addition to the estimation of the parameters of the system.

C Estimator derivation

This additional section is written to include details on the derivation of the Kalman filter (KF) equations used for on-line estimation, as discussed in Sec. 2.4. The definition of an estimator is of key importance when designing an on-line controller system, and may sometimes prove to be harder than designing the controller itself. In Sec. C.1, the Kalman filter for a linear time-discrete system is established as a starting point, and this is extended to continuous linear systems in Sec. C.2. In Sec. C.3, the (Extended) Kalman filter (EKF) for nonlinear systems is covered. The following derivation adopts the approach, and hence the notation, of Simon. [18]

C.1 Kalman filter estimator for linear time-discrete systems

The starting point for the derivation of the Kalman filter is a linear time-discrete system. This system is represented, in a general formulation, in Eqs. C.3 & C.4. In the case where the linear system is not time-discrete but continuous, which indeed is the case for most dynamic systems, the linear continuous system is converted to a discrete system. The formulation of the continuous system is shown in Eq. C.1, where the dotted notation indicates differentiation with respect to time.

$$\dot{x} = Ax + Bu + \omega \tag{C.1}$$

The discretized solution[2] to such a system is indicated in Eq. C.2, in which x_k is the vector of states and u_k is the vector of inputs at time t_k. The discrete time interval $t_k - t_{k-1}$ is denoted Δt_k.

$$x_k = e^{A\Delta t_k} x_{k-1} + \int_{t_{k-1}}^{t_k} e^{A(t_k-\tau)} \left[B(\tau)u(\tau) + \omega(\tau)\right] d\tau \tag{C.2}$$

The general formulation for the linear time-discrete system is presented in Eqs. C.3 & C.4.

$$x_k = F_{k-1}x_{k-1} + G_{k-1}u_{k-1} + w_{k-1} \tag{C.3}$$

$$y_k = H_k x_k + v_k \tag{C.4}$$

In this system, x_k is the vector of states at the discrete point t_k in time. The corresponding vector of measurements, y_k, is represented by the sum of the state measurements ($H_k x_k$) and the measurement noise (v_k). Here, H_k is a matrix deciding which of the states that go into the measurement vector. F_k and G_k are the system matrices, governing how the states and inputs, respectively, at time t_k affect the state at time t_{k+1}. In special situations where F and G are time independent (constant) matrices, the system is said to be linear time-independent (LTI). The process noise at time t_k is represented by w_k. Both noise terms (w_k and v_k) are assumed to be zero-mean white noise with covariances[3] Q_k and R_k, respectively. The zero-mean assumption is justified by the requirement that systematic mean-contributions from the noise are accounted for in the model, i.e. the system matrices. By comparison between the solution for the discretized

[2] These expressions contain mathematical terms such as matrix exponentials, convolution integrals, and other complex concepts. These concepts are not elaborated in this text, and the curious reader is referred to more extensive texts for a walkthrough, e.g. Kreyszig. [19]

[3] Details on theoretical concepts from the field of statistics are omitted from this text. For theoretical background on statistics, Walpole *et. al.* was consulted. [20]

continuous system (Eq. C.2) and the general discrete system (Eq. C.3), the system matrices must be as indicated in Eqs. C.5 - C.7.

$$F_{k-1} = e^{A\Delta t_k} \tag{C.5}$$

$$G_{k-1} = \int_{t_{k-1}}^{t_k} e^{A(t_k-\tau)} B d\tau \tag{C.6}$$

$$w_{k-1} = \int_{t_{k-1}}^{t_k} e^{A(t_k-\tau)} \omega(\tau) d\tau \tag{C.7}$$

The purpose of the estimator is to propose an optimal estimate of the state at time t_k (x_k), and this estimate is denoted $\hat{x}_k$. A distinction is made between *a priori* estimates and *a posteriori*[4] estimates, which are denoted $\hat{x}_k^-$ and $\hat{x}_k^+$, respectively. In this sense, whether an estimate at time t_k is *a priori* or *a posteriori* is a question of whether the measurement information at time t_k is used or not. Mathematically, this can be formulated as in Eqs. C.8 & C.9 for *a priori* and *a posteriori*, respectively. It is emphasized that $\hat{x}_k^-$ and $\hat{x}_k^+$ are estimates of the same quantity, i.e. the state vector at time t_k (x_k).

$$\hat{x}_k^- = E\left[x_k \,\middle|\, y_1, y_2, \ldots, y_{k-1} \right] \tag{C.8}$$

$$\hat{x}_k^+ = E\left[x_k \,\middle|\, y_1, y_2, \ldots, y_k \right] \tag{C.9}$$

Here, $E[\cdots]$ denotes the expected value. As is evident from Eqs. C.8 & C.9, the two estimates use the same previous measurements ($y_1, y_2, \ldots, y_{k-1}$). The only difference is that the *a posteriori* estimate also utilizes the measurement at time t_k (y_k).

From this point on, P_k is introduced to denote the covariance of the state estimate error at time t_k. Notice that the state covariance also has a distinction between *a priori* and *a posteriori* information, as indicated in Eqs. C.10 & C.11.

$$P_k^- = E\left[\left(x_k - \hat{x}_k^- \right) \left(x_k - \hat{x}_k^- \right)^T \right] \tag{C.10}$$

$$P_k^+ = E\left[\left(x_k - \hat{x}_k^+ \right) \left(x_k - \hat{x}_k^+ \right)^T \right] \tag{C.11}$$

The system is initiated by an initial state, x_0^+, having a covariance P_0^+. In the case where x_0^+ is well known, the covariance (i.e. uncertainty) is zero, but in the case where x_0^+ is (very) uncertain, P_0^+ approaches ∞I. The next step is to analyze how the state estimates and covariances propagate through time. Using the established definition in Eq. C.8 in combination with the proposed model in Eq. C.3, the general propagation of state estimates becomes as indicated in Eq. C.12. A justification for this is that the process noise (w_k) has a zero-mean propability distribution.

$$\begin{aligned} & \hat{x}_1^- = F_0 \hat{x}_0^+ + G_0 u_0 \\ \text{General expression:} \quad & \hat{x}_k^- = F_{k-1} x_{k-1}^+ + G_{k-1} u_{k-1} \end{aligned} \tag{C.12}$$

[4]These terms are orginially known from philosophy. *A priori* and *a posteriori* mean "from the earlier" and "from the later", respectively, and these terms are used to describe the "quality" of the knowledge.

The propagation of the covariances is slightly more complicated. The approach starts from the definition in Eq. C.10, the model in Eq. C.3 and the result in Eq. C.12. The result, showing how the covariances propagate in time, is provided in Eq. C.13.

$$\begin{aligned}
P_k^- &= E\left[\left(x_k - \hat{x}_k^-\right)\left(x_k - \hat{x}_k^-\right)^T\right] \\
\left(x_k - \hat{x}_k^-\right) &= F_{k-1}x_{k-1} + G_{k-1}u_{k-1} + w_{k-1} - \hat{x}_k^- \\
\hat{x}_k^- &= F_{k-1}\hat{x}_{k-1}^+ + G_{k-1}u_{k-1} \\
\implies \quad \left(x_k - \hat{x}_k^-\right) &= F_{k-1}\left(x_{k-1} - \hat{x}_{k-1}^+\right) + w_{k-1} \\
\left(x_k - \hat{x}_k^-\right)\left(x_k - \hat{x}_k^-\right)^T &= F_{k-1}\left(x_{k-1} - \hat{x}_{k-1}^+\right)\left(x_{k-1} - \hat{x}_{k-1}^+\right)^T F_{k-1}^T + w_{k-1}w_{k-1}^T \\
&\quad + F_{k-1}\left(x_{k-1} - \hat{x}_{k-1}^+\right)w_{k-1}^T + w_{k-1}\left(x_{k-1} - \hat{x}_{k-1}^+\right)^T F_{k-1}^T \\
\implies \quad P_k^- &= F_{k-1}P_{k-1}^+F_{k-1}^T + Q_{k-1}
\end{aligned} \tag{C.13}$$

This result was found by recognizing the expected value of each of the terms in the expression for $\left(x_k - \hat{x}_k^-\right)\left(x_k - \hat{x}_k^-\right)^T$, once it was written out. This is illustrated in Eq. C.14. Eq. C.15 is in agreement with the proposed covariance of the process noise, while Eqs. C.16 & C.17 agrees with the fact that the estimation error $\left(x_k - \hat{x}_k^-\right)$ is uncorrelated with the process noise (w_k).

$$\begin{aligned}
E\left[F_{k-1}\left(x_{k-1} - \hat{x}_{k-1}^+\right)\left(x_{k-1} - \hat{x}_{k-1}^+\right)^T\right] &= F_{k-1}E\left[\left(x_{k-1} - \hat{x}_{k-1}^+\right)\left(x_{k-1} - \hat{x}_{k-1}^+\right)^T\right]F_{k-1}^T \\
&= F_{k-1}P_{k-1}^+F_{k-1}^T
\end{aligned} \tag{C.14}$$

$$E\left[w_{k-1}w_{k-1}^T\right] = Q_{k-1} \tag{C.15}$$

$$E\left[w_{k-1}\left(x_{k-1} - \hat{x}_{k-1}^+\right)^T F_{k-1}^T\right] = 0 \tag{C.16}$$

$$E\left[F_{k-1}\left(x_{k-1} - \hat{x}_{k-1}^+\right)w_{k-1}^T\right] = 0 \tag{C.17}$$

It has now been investigated how both state estimates and covariances propagate through time for a linear time-discrete system, as indicated by Eqs. C.12 & C.13, respectively. These equations give the relationships for the transitions $\hat{x}_k^+ \to \hat{x}_{k+1}^-$ and $P_k^+ \to P_{k+1}^-$, i.e. the propagation in time. The next step is to formulate expressions for the transitions $\hat{x}_k^- \to \hat{x}_k^+$ and $P_k^- \to P_k^+$, i.e. the transition from *a priori* estimates to *a posteriori* estimates at time t_k. This is performed using the approach of recursive least squares estimation. Having a measurement (y_k) as indicated in Eq. C.4, the measurement correction is formulated as indicated in Eq. C.18. Here, K_k is the estimator gain matrix, which becomes the Kalman filter gain when the errors between the states and the state estimates are minimized.

$$\hat{x}_k^+ = \hat{x}_k^- + K_k\left(y_k - H_k\hat{x}_k^-\right) \tag{C.18}$$

To minimize the estimation errors, a proper cost function[5] is defined as indicated in Eq. C.19, in

[5] In this formulation, `Tr` denotes the Trace of the matrix. This is a matrix operation which consists of calculating the sum of the elements on the diagonal of the matrix.

which the error ϵ is defined as shown in Eq. C.20.

$$\begin{aligned} J_k &= E\left[\left(x_1 - \hat{x}_1^+\right)^2\right] + E\left[\left(x_2 - \hat{x}_2^+\right)^2\right] + \cdots + E\left[\left(x_n - \hat{x}_n^+\right)^2\right] \\ &= E\left[\epsilon_{x1,k}^2 + \epsilon_{x2,k}^2 + \cdots + \epsilon_{xn,k}^2\right] \\ &= E\left[\epsilon_{x,k}^T \epsilon_{x,k}\right] \\ &= Tr\left(E\left[\epsilon_{x,k}\epsilon_{x,k}^T\right]\right) \end{aligned} \tag{C.19}$$

$$\epsilon_{xi} = x_i - \hat{x}_i^+ \tag{C.20}$$

The expected value for the vector of errors is needed, and this expression can be manipulated by combining the error definition (Eq. C.20), the measurement definition (Eq. C.4) and the estimator measurement correction (Eq. C.18). The result is portrayed in Eq. C.21.

$$\begin{aligned} E\left[\epsilon_{x,k}\right] &= E\left[x - \hat{x}_k^+\right] \\ &= E\left[x - \hat{x}_k^- - K_k\left(y_k - H_k\hat{x}_k^-\right)\right] \\ &= E\left[x - \hat{x}_k^- - K_k\left(H_k x + v_k - H_k\hat{x}_k^-\right)\right] \\ &= E\left[x - \hat{x}_k^- - K_k H_k\left(x - \hat{x}_k^-\right) - K_k v_k\right] \\ &= \left(I - K_k H_k\right) E\left[x - \hat{x}_k^-\right] - K_k E\left[v_k\right] \end{aligned} \tag{C.21}$$

This result is possible to expand to give the expected value for the square of the error, as shown in Eq. C.22.

$$\begin{aligned} E\left[\epsilon_{x,k}\epsilon_{x,k}^T\right] &= \left(I - K_k H_k\right) E\left[\left(x - \hat{x}_k^-\right)\left(x - \hat{x}_k^-\right)^T\right]\left(I - K_k H_k\right)^T \\ &\quad - K_k E\left[v_k\left(x - \hat{x}_k^-\right)^T\right] + K_k E\left[v_k v_k^T\right] K_k^T \\ &\quad - \left(I - K_k H_k\right) E\left[\left(x - \hat{x}_k^-\right) v_k^T\right] K_k^T \\ \implies \quad E\left[\epsilon_{x,k}\epsilon_{x,k}^T\right] &= \left(I - K_k H_k\right) P_k^- \left(I - K_k H_k\right)^T + K_k R_k K_k^T \end{aligned} \tag{C.22}$$

This is achieved by recognizing the definition of P_k^-, aswell as remembering that the noises are uncorrelated with the error. R_k is, as postulated earlier, the covariance of the measurement noise at time t_k. These requirements are summarized in Eqs. C.23 - C.25.

$$P_k^- = E\left[\left(x_k - \hat{x}_k^-\right)\left(x_k - \hat{x}_k^-\right)^T\right] \tag{C.23}$$

$$E\left[v_k\left(x - \hat{x}_k^-\right)\right] = 0 \tag{C.24}$$

$$E\left[v_k v_k^T\right] = R_k \tag{C.25}$$

Remembering the definition of P_k^+ from Eq. C.11, these previous results also enable the formulation of P_k^+, as represented in Eq. C.26.

$$\begin{aligned} & P_k^+ = E\left[\epsilon_{x,k}\epsilon_{x,k}^T\right] \\ \implies \quad & P_k^+ = \left(I - K_k H_k\right) P_k^- \left(I - K_k H_k\right)^T + K_k R_k K_k^T \end{aligned} \tag{C.26}$$

The purpose in the next steps is to solve an optimization problem to uncover the optimal estimator gain, K_k, which is the Kalman filter gain. To do this, the cost function (accounting for the estimator error), is differentiated with respect to K_k and set equal to zero. This procedure is shown in Eqs. C.27 - C.31. The step between Eq. C.30 and Eq. C.31 is performed by postulating that $\frac{\partial\left(Tr\left(ABA^T\right)\right)}{\partial A} = 2AB$, given that B is symmetric. In this case, this is assumed to be fulfilled.

$$\frac{\partial J}{\partial K_k} = 0 \tag{C.27}$$

$$\frac{\partial\left(Tr\left(E\left[\epsilon_{x,k}\epsilon_{x,k}^T\right]\right)\right)}{\partial K_k} = 0 \tag{C.28}$$

$$\frac{\partial}{\partial K_k}\left(Tr\left(\left(I - K_k H_k\right) P_k^- \left(I - K_k H_k\right)^T + K_k R_k K_k^T\right)\right) = 0 \tag{C.29}$$

$$\frac{\partial\left(Tr\left(\left(I - K_k H_k\right) P_k^- \left(I - K_k H_k\right)^T\right)\right)}{\partial K_k} + \frac{\partial\left(Tr\left(K_k R_k K_k^T\right)\right)}{\partial K_k} = 0 \tag{C.30}$$

$$2\left(I - K_k H_k\right) P_k^- \left(-H_k^T\right) + 2K_k R_k = 0 \tag{C.31}$$

Eq. C.31 is reorganized to yield the final expression for the Kalman filter gain, which is shown in Eq. C.32.

$$\implies \quad K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + R_k\right)^{-1} \tag{C.32}$$

Finally, the resulting equations for the Kalman filter algorithm are listed in Eqs. C.33 - C.37. This system of equations is usually referred to as Riccati-equations.

$$P_k^- = F_{k-1} P_{k-1}^+ F_{k-1}^T + Q_{k-1} \tag{C.33}$$

$$K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + R_k\right)^{-1} \tag{C.34}$$

$$\hat{x}_k^- = F_{k-1}\hat{x}_{k-1}^+ + G_{k-1}u_{k-1} \tag{C.35}$$

$$\hat{x}_k^+ = \hat{x}_k^- + K_k\left(y_k - H_k\hat{x}_k^-\right) \tag{C.36}$$

$$P_k^+ = \left(I - K_k H_k\right) P_k^- \left(I - K_k H_k\right)^T + K_k R_k K_k^T \tag{C.37}$$

In some cases, it is desirable to combine the measurement updates and the estimation propagation in time to yield one expression including both. In other words, the *a priori* state estimate update from time t_{k-1} to time t_k is combined with the *a posteriori* update at time t_k, and the same strategy

is applied to the covariances (P_k). This is illustrated in Eqs. C.38 & C.39.

$$\left.\begin{aligned}\hat{x}_{k-1}^{+} &\rightarrow \hat{x}_{k}^{-}\\ \hat{x}_{k}^{-} &\rightarrow \hat{x}_{k}^{+}\end{aligned}\right\} \implies \hat{x}_{k-1}^{+} \rightarrow \hat{x}_{k}^{+}$$

$$\implies \quad \hat{x}_{k}^{+} = (I - K_k H_k)\left(F_{k-1}\hat{x}_{k-1}^{+} + G_{k-1}u_{k-1}\right) + K_k y_k \tag{C.38}$$

$$\left.\begin{aligned}P_{k-1}^{+} &\rightarrow P_{k}^{-}\\ P_{k}^{-} &\rightarrow P_{k}^{+}\end{aligned}\right\} \implies P_{k-1}^{+} \rightarrow P_{k}^{+}$$

$$\implies \quad P_{k}^{+} = (I - K_k H_k)\left(F_{k-1}P_{k-1}^{+}F_{k-1}^{T} + Q_{k-1}\right) \tag{C.39}$$

This concludes the establishment of the Kalman filter equations for a linear time-discrete system.

C.2 Kalman filter estimator for linear continuous systems

The purpose of this section is to extend the established state estimation theory for linear time-discrete system (Sec. C.1) to also account for linear continuous systems. Continuous systems are often encountered in dynamic modeling of systems from real applications, and this is an important class of systems, considering both the linear and the nonlinear cases. In this section, the linear approach is discussed. The linear continuous system is equal to the one introduced in Sec. C.1, and the measurement dynamics are also included, as indicated in Eqs. C.40 & C.41. As for the linear case, the system is initiated by the initial state estimates ($\hat{x}_0^+$) which is known to some accuracy, reflected in the covariances of the initial states (P_0^+).

$$\dot{x} = Ax + Bu + w \tag{C.40}$$

$$y = Cx + v \tag{C.41}$$

The strategy for extending the Kalman filter equations to the continuous system is to examine and compare the system matrices for the two system formulations. The system matrices for the discrete system, as introduced in Eqs. C.5 & C.6 in Sec. C.1, are reproduced in Eqs. C.42 & C.43. Notice that the G-matrix has been rewritten[6] for the purpose of this section, and that the time indices have been omitted to emphasize the transfer from the discrete formulation to the continuous formulation. Here, T is included to denote a time interval (Δt), which is assumed to be small. Under the assumption that T is small (and later to be infinitesimally small), the matrix exponential[7] expressions can be approximated to yield far simpler expressions.

$$F = e^{AT} \qquad \approx (I + AT) \tag{C.42}$$

$$G = e^{AT}\left[I - e^{-AT}\right]A^{-1}B \quad \approx BT \tag{C.43}$$

The covariances for the noises of the system must also be considered. For the discrete case, the process noise (w_k) and measurement noise (v_k) at time t_k have covariances Q_k and R_k, respectively, while the continuous case has instantaneous covariances of Q_c and R_c, respectively. For the continuous noise functions, this is formulated mathematically in Eqs. C.44 & C.45.

$$E\left[w(t)w(\tau)\right] = Q_c\delta(t-\tau) \tag{C.44}$$

$$E\left[v(t)v(\tau)\right] = R_c\delta(t-\tau) \tag{C.45}$$

[6] Careful examination between Eq. C.6 and Eq. C.43 confirms that the reformulation is valid.

[7] The matrix exponential is defined as an infinite series: $e^X = \sum_{k=0}^{\infty}\frac{1}{k!}X^k = I + X + \frac{1}{2!}X^2 + \frac{1}{3!}X^3 + \cdots$. It has many properties in common with the traditional exponential function (i.e. the base of the natural logarithm).

For this to hold for a small time interval T, it can be shown that the relationships between Q and Q_c and R and R_c must be as Eqs. C.46 & C.47, respectively. This is related to the total covariance associated with the entire time interval, i.e. sum of time intervals.

$$Q = Q_c \Delta t \tag{C.46}$$

$$R = \frac{R_c}{\Delta t} \tag{C.47}$$

Starting with the Kalman filter gain equation in Eq. C.34 from Sec. C.1, the corresponding terms for the continous system are inserted to yield the continuous Kalman filter gain, as represented in Eq. C.48.

$$\begin{aligned} K_k &= P_k^- H_k^T \left(H_k P_k^- H_k^T + R_k \right)^{-1} \\ K_k &= P_k^- C^T \left(C P_k^- C^T + \frac{R_c}{T} \right)^{-1} \\ \frac{K_k}{T} &= P_k^- C^T \left(C P_k^- C^T T + R_c \right)^{-1} \\ \lim_{T \to 0} \frac{K_k}{T} &= P_k^- C^T R_c^{-1} \end{aligned} \tag{C.48}$$

The covariances for the states of the system also need attention. The covariance formulations from Eqs. C.33 & C.37 are reproduced in Eqs. C.49 & C.50, respectively. For the purpose of the continuous formulation, the indices of the system matrices have been omitted, and Eq. C.33 has been reorganized[8].

$$P_{k+1}^- = F P_k^+ F^T + Q \tag{C.49}$$

$$P_k^+ = \left(I - K_k C \right) P_k^- \tag{C.50}$$

For small values of T, the approximation in Eq. C.42 is valid, and the equation for the covariances are developed as indicated below. The final result, known as the differential Riccati equation is presented in Eq. C.51.

$$\begin{aligned} P_{k+1}^- &= F P_k^+ F^T + Q \\ &= \left(I + AT \right) P_k^+ \left(I + AT \right)^T + Q_c T \\ &= P_k^+ + \left(A P_k^+ + P_k^+ A^T + Q_c \right) T + A P_k^+ A^T T^2 \\ &= \left(I - K_k C \right) P_k^- + A \left(I - K_k C \right) P_k^- A^T T^2 \\ &\quad + \left[A \left(I - K_k C \right) P_k^- p \left(I - K_k C \right) P_k^- A^T + Q_c \right] T \\ \implies \quad \frac{P_{k+1}^- - P_k^-}{T} &= -\frac{K_k C P_k^-}{T} + A P_k^+ A^T T \\ &\quad + \left(A P_k^- A K_k C P_k^- + P_k^- A^T - K_k C P_k^- A^T + Q_c \right) \\ \implies \quad \dot{P} = \lim_{T \to 0} \frac{P_{k+1}^- - P_k^-}{T} &= -P C^T R_c^{-1} C P + A P + P A^T + Q_c \end{aligned} \tag{C.51}$$

[8]The algebraic manipulation of the expression has been omitted from this text, but careful analysis shows that the two expression are equal. [18]

In this operation, the already established expression for the continuous Kalman filter gain, which is presented in Eq. C.48, was applied.

The next step is to establish a differential formulation for the state estimate propagation in time. The strategy is completely analogous to what was done for the covariance in Eq. C.51. The state estimate equations, which are indicated in Eqs. C.35 & C.36 for the discrete case, are rewritten for the continuous case in Eqs. C.52 & C.53.

$$\hat{x}_{k+1}^{-} = F\hat{x}_k^{+} + Gu_k \tag{C.52}$$

$$\hat{x}_k^{+} = \hat{x}_k^{-} + K_k\left(y_k - H\hat{x}_k^{-}\right) \tag{C.53}$$

For small values of T, the approximations in Eqs. C.42 & C.43 are valid, and Eqs. C.52 & C.53 are combined to yield the final result presented in Eq. C.54 as shown below.

$$\begin{aligned}
\hat{x}_{k+1}^{+} &= F\hat{x}_k^{+} + Gu_k + K_k\left(y_k - HF\hat{x}_k^{+} - HGu_k\right) \\
&= (I + AT)\,\hat{x}_k^{+} + BTu_k \\
&\quad + K_k\left(y_k - C\,(I + AT)\,\hat{x}_k^{+} - CBTu_k\right) \\
&= \hat{x}_k^{+} + AT\hat{x}_k^{-} + BTu_k \\
&\quad + PC^T R_c^{-1} T\left(y_k - C\hat{x}_k^{+} - CAT\hat{x}_k^{+} - CBTu_k\right) \\
\Longrightarrow \qquad \frac{\hat{x}_{k+1}^{+} - \hat{x}_k^{+}}{T} &= A\hat{x}_k + Bu_k \\
&\quad + PC^T R_c^{-1}\left(y_k - C\hat{x}_k^{+} - CAT\hat{x}_k^{+} - CBTu_k\right) \\
\Longrightarrow \quad \dot{\hat{x}} = \lim_{T\to 0}\frac{\hat{x}_{k+1}^{+} - \hat{x}_k^{+}}{T} &= A\hat{x} + Bu + K\,(y - C\hat{x})
\end{aligned} \tag{C.54}$$

In this operation, the already established expression for the continuous Kalman filter gain, which is presented in Eq. C.48, was applied.

Finally, the resulting equations for the linear continous system are achieved, as presented in Eqs. C.55 - C.57.

$$\dot{\hat{x}} = A\hat{x} + Bu + K\,(y - C\hat{x}) \tag{C.55}$$

$$K = PC^T R_c^{-1} \tag{C.56}$$

$$\dot{P} = -PC^T R_c^{-1} CP + AP + PA^T + Q_c \tag{C.57}$$

This system of equations is often referred to as the differential Riccati equations for the Kalman filter estimator.

C.3 Extended Kalman filter estimator for nonlinear continuous systems

The purpose of this section is to briefly describe how Kalman filter estimators can be applied for nonlinear continuous systems. In previous sections, the filtering equations for linear systems have been established. These results provide a useful basis for further considerations, but emphasis is put on the fact that real-life systems which are entirely linear are non-existent.

In some cases, it may be sufficient to approximate the nonlinear model as a linearized model. The procedure of linearization is illustrated in Eqs. C.58 - C.60. Here, Φ is an arbitrary continuous multivariable function, chosen to illustrate the procedure. The linearization procedure has its basis in the theory of Taylor series, in which a linearized function can be considered to be the first order Taylor series polynomial of the orginal function, when all higher order terms are neglected. In Eqs. C.58 - C.60, the star ($*$) indicates nominal trajectory values, from which the deviations in the linearized function are considered. The nominal trajectory is defined as indicated in Eqs. C.61 & C.62.

$$\Phi = f(x_1, x_2, \ldots, x_n) \tag{C.58}$$

$$\implies \quad \Phi \approx \Phi_* + \left.\frac{\partial f}{\partial x_1}\right|_* (x_1 - x_1^*) + \left.\frac{\partial f}{\partial x_2}\right|_* (x_2 - x_2^*) + \cdots + \left.\frac{\partial f}{\partial x_n}\right|_* (x_n - x_n^*) \tag{C.59}$$

$$\implies \quad \Delta\Phi \approx k_1\Delta x_1 + k_2\Delta x_2 + \cdots + k_n\Delta x_n \tag{C.60}$$

$$\dot{x}_* = f(x_*, u_*, w_*, t) \tag{C.61}$$

$$y_* = h(x_*, v_*, t) \tag{C.62}$$

Using this strategy, which (necessarily) also applies for differential equations, a nonlinear dynamic model can be linearized as indicated in Eqs. C.63 - C.66.

$$\dot{x} = f\left(x, u, w, t\right) \tag{C.63}$$

$$y = h\left(x, v, t\right) \tag{C.64}$$

$$\Delta\dot{x} = \underbrace{\left.\frac{\partial f}{\partial x}\right|_*}_{A} \Delta x + \underbrace{\left.\frac{\partial f}{\partial u}\right|_*}_{B} \Delta u + \underbrace{\left.\frac{\partial f}{\partial w}\right|_*}_{D} \Delta w$$

$$\implies \quad \Delta\dot{x} = A\Delta x + B\Delta u + D\Delta w \tag{C.65}$$

$$\Delta y = \underbrace{\left.\frac{\partial h}{\partial x}\right|_*}_{C} \Delta x + \underbrace{\left.\frac{\partial h}{\partial v}\right|_*}_{E} \Delta v$$

$$\implies \quad \Delta y = C\Delta x + E\Delta v \tag{C.66}$$

At this point, a Linearized Kalman filter (LKF) estimator can be formulated for the linearized model in agreement with the results from the discussion on linear continuous estimators (Sec. C.2). In that case, the results would be as indicated in Eqs. C.67 - C.69, in which $\tilde{Q} = DQD^T$ and $\tilde{R} = ERE^T$, i.e. the covariance of the process noise and measurement noise, respectively.

$$\Delta\dot{\hat{x}} = A\Delta\hat{x} + K\left(\Delta y - C\Delta\hat{x}\right) \tag{C.67}$$

$$K = PC^T\tilde{R}^{-1} \tag{C.68}$$

$$\dot{P} = AP + PA^T + \tilde{Q}^T - PC^T\tilde{R}^{-1}CP \tag{C.69}$$

Another approach for nonlinear state estimation, which is slightly more sophisticated, is the so-called Extended Kalman filter (EKF). This is believed to be the most abundant approach utilized in the world when it comes to nonlinear estimation implementations [18]. In the EKF, one of the most apparent problems associated with the Kalman filter for the linearized process is addressed directly, namely the fact that the optimal trajectory (from which the deviations are calculated) may not be

straight-forward to decide. In this sense, the definition of the nominal trajectory (Eqs. C.61 & C.62) is combined with the result from the linearized state estimation procedure (Eq. C.67) to yield the expression in Eq. C.70.

$$\dot{x}_* + \Delta\dot{\hat{x}} = f\left(x_*, u_*, w_*, t\right) + A\Delta\hat{x} + K\left(y - y_* - C\left(\hat{x} - x_*\right)\right) \tag{C.70}$$

The next step in the strategy, which is the key maneuver of the EKF, is to set x_* equal to $\hat{x}$, i.e. to use the state estimates as the nominal trajectory. In this operation, it is kept in mind that $\Delta\hat{x} = \hat{x} - x_*$. This will change the differential equation for the state estimates, while the expressions for the estimator gain aswell as covariances are the same. Finally, the system of equations for the EKF is presented in Eqs. C.71 - C.73.

$$\dot{\hat{x}} = f\left(\hat{x}, u, w_*, t\right) + K\left(y - h\left(\hat{x}, v_*, t\right)\right) \tag{C.71}$$

$$K = PC^T\tilde{R}^{-1} \tag{C.72}$$

$$\dot{P} = AP + PA^T + \tilde{Q} - PC^T\tilde{R}^{-1}CP \tag{C.73}$$

In most real-life engineering cases, continuous models are used for the dynamic systems, while the measurements are discrete, i.e. sampled at specific points in time, as indicated in Eqs. C.74 & C.75.

$$\dot{x} = f\left(x, u, w, t\right) \tag{C.74}$$

$$y_k = h_k\left(x_k, v_k\right) \tag{C.75}$$

In this case, a so-called Hybrid EKF (HEKF) is deployed. Here, the purpose is to use the nonlinear continuous dynamics to propagate the state estimates in time without the use of measurements between the points of measurement, yet include a discrete step from *a priori* to *a posteriori* at each time t_k where a measurement is available. The set of equations for this case is presented in Eqs. C.76 - C.80, in agreement with previous results. Here, the process noise (Q) is a continuous quantity, while the measurement noise (R_k) is discrete at each time instant of valid measurement.

Between t_k^+ and t_{k+1}^-:

$$\dot{\hat{x}} = f\left(\hat{x}, u, w_*, t\right) \tag{C.76}$$

$$\dot{P} = AP + PA^T + \tilde{Q} \tag{C.77}$$

Between t_k^- and t_k^+:

$$K_k = P_k^- H_k^T\left(H_k P_k^- H_k^T + \tilde{R}_k\right)^{-1} \tag{C.78}$$

$$\hat{x}_k^+ = \hat{x}_k^- + K_k\left(y_k - h_k\left(\hat{x}_k^-, v_*, t_k\right)\right) \tag{C.79}$$

$$P_k^+ = \left(I - K_k H_k\right)P_k^-\left(I - K_k H_k\right)^T + K_k\tilde{R}_k K_k^T \tag{C.80}$$

In addition to what these sections have been considering, there are more sophisticated approaches available to be deployed for on-line estimation. The Iterated Extended Kalman filter (IEKF) is one example, in which the Taylor series expansion, i.e. the linearization, of the model functions are re-iterated at each point using the *a posteriori* state estimates. There also exist higher order methods[9] for implementing the EKF, none of which are considered in this project work. The Unscented Kalman Filter (UKF) is another example of a more complicated estimator, different

[9]The strategy for the higher order methods is to use higher order Taylor series expansion polynomials than the first order method which was deployed for the LKF and the EKF. This will act to reduce the error originating from the linearization, but it will lead to more complex calculations and hence a larger demand for computational power. [18]

from the EKF in that it reduces the error originating from the linearization in the EKF. Possible extensions to the EKF, like the UKF, is left for future work, and detailed considerations has been omitted from this text.

When considering which estimator to implement, it must be recognized that while some filters are more accurate than others, they demand more computational power. This can prove to be a major concern for on-line applications, and the performance of the estimator must be evaluated in close comparison with the actual system on which the estimator will be implemented. Generally speaking, the performance of the estimator with respect to computational time is closely correlated with the performance of the process model itself (i.e. the complexity of the process model).

This concludes the discussion on Kalman filter estimators for dynamic systems, for the purposes of this project work. For off-line parameter estimation and model fitting, it is not crucial to establish the on-line filtering algorithms. This will, however, be a key feature in the complete model-based predictive controller, which development and design is suggested as an extension to this work.

D Fluid properties for the emulsion copolymerization system

This additional section includes physical properties for the components involved in the copolymer system. Here, ρ denotes fluid density, given in kg/m^3 and c_p denotes specific heat capacity, given in kJ/kg. These quantities are collected for different temperatures, and fitted to give polynominal functions in temperature (T). This data is acquired from an internal report from the COOPOL project regarding the modeling of the specific reactor system. [22]

Water:

$$\rho_W = 1.597e^{-5}(T-273.15)^3 - 5.926e^{-3}(T-273.15)^2 + 0.01741(T-273.15) + 1000; \quad \text{(D.1)}$$

$$c_{p,W} = 5.745e^{-13}(T-273.15)^6 - 2.181e^{-10}(T-273.15)^5 + 3.421e^{-8}(T-273.15)^4 - 2.831e^{-6}(T-273.15)^3 + 1.381e^{-4}(T-273.15)^2 - 3.656e^{-3}(T-273.15) + 4.218 \quad \text{(D.2)}$$

Monomer type 1:

$$\rho_{M1} = 1055 - 0.001T^2 - 0.2268T \quad \text{(D.3)}$$

$$c_{p,M1} = 0.7994 + 0.003222T \quad \text{(D.4)}$$

$$\rho_{Pol,1} = 1055 - 0.2(T-273.15) \quad \text{(D.5)}$$

$$c_{p,Pol,1} = 1.08665 + 8e^{-8}T^3 - 5.727e^{-5}T^2 + 0.0176516T \quad \text{(D.6)}$$

Monomer type 2:

$$\rho_{M2} = 1191.9583 - 1.0006T \quad \text{(D.7)}$$

$$c_{p,M2} = 0.57385 + 0.00418T \quad \text{(D.8)}$$

$$\rho_{Pol,2} = 1050 - 0.2(T-273.15) \quad \text{(D.9)}$$

$$c_{p,Pol,2} = 1.7509942 + 0.002573124(T-273.15) \quad \text{(D.10)}$$

E Experimental data

This additional section is added to include explicit details regarding the experimental data used in the parameter fitting of the emulsion copolymerization process. The experimental data is acquired from an internal database in the COOPOL project. The experiment was conducted at the Technical University of Dortmund (Technische Universität Dortmund), and submitted to the COOPOL project as a starting point for discussing the behavior of the specific semi-batch reactor system. The data in Tab. E.1 is presented in dimensionless time.

Table E.1: Experimental data used for model validation.

Time [-]	**Conversion [%]**
0.0000	0.00
0.0667	5.41
0.1333	12.97
0.2033	21.24
0.2733	29.44
0.3333	37.22
0.4000	46.43
0.4667	56.35
0.5333	68.19
0.6000	79.79
0.6667	90.93
0.7333	97.08
0.8000	99.33
0.8667	99.84
0.8700	99.82
1.0000	99.99

Molecular weight at end of batch time, number average [kg/mol]: 19.496
Molecular weight at end of batch time, weight average [kg/mol]: 39.495

F Pendulum example model for software demonstration

The purpose of this additional section is to provide details on the simplified pendulum model used for illustrating the features of Modelica[1], Dymola and Cybernetica ModelFit. These software tools are described in the text, and this section presents how a simple problem is implemented and treated using these tools. The reason for choosing a simplied example is that the Modelica code of the semi-batch emulsion copolymerization reactor is quite extensive. The specific problem is a fictitious, super-simplified pendulum without any loss. The pendulum is represented in Fig. F.1. In this figure, L denotes the length of the string (which is assumed to have zero mass), Φ is the

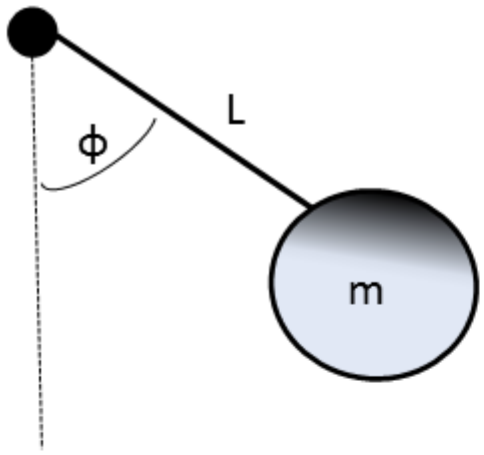


Figure F.1: Conceptual drawing of a simplified pendulum, used for software demonstration.

angle describing the pendulum deviation from the vertical rest position, and m is the mass of the pendulum. Using ω to denote the angular velocity of the pendulum, J to denote the moment of inertia and g to denote the gravitational acceleration, the dynamic description of the pendulum can be formulated as indicated in Eqs. F.1 - F.3.

$$\frac{d\Phi}{dt} = \omega \tag{F.1}$$

$$J\,\frac{d\omega}{dt} = -mgL\,sin(\Phi) \tag{F.2}$$

$$J = mL^2 \tag{F.3}$$

In other words, the pendulum description is a system of two ordinary differential equations. The straight-forward Modelica implementation of this is portayed in Fig. F.2. From this simple example, several of Modelicas strengths are evident. The easy declaration of variables in agreement with whether they are states, parameters, constants, etc. works alongside the intuitive use of units for the variables. In the `equation`-part, the dynamic equations are entered in a straight-forward manner, with the possibility of using built-in functions, e.g. the declaration of differentiation with respect to time (`der()`) or the sine function (`sin()`), as shown in Fig. F.2.

Fig. F.3 shows a ModelFit simulation for the angle of the pendulum (Φ for the Modelica-code from Fig. F.2 (blue line). The other line (pink color) shows the measured angles for another

[1]The curious reader is referred to more extensive publications on Modelica for more details. [10]

```
within ;
model pendulum

parameter Modelica.SIunits.Mass m = 1 "Mass";
parameter Modelica.SIunits.Length L = 1 "Length";
parameter Modelica.SIunits.Acceleration g = 9.81 "Acceleration";
parameter Modelica.SIunits.MomentOfInertia J = m*L^2 "MOI";
Modelica.SIunits.Angle phi(start=0.1);
Modelica.SIunits.AngularVelocity w(start=0);

equation
der(phi) = w;
J*der(w) = -m*g*L*sin(phi);

  annotation (uses(Modelica(version="3.2")));
end pendulum;
```

Figure F.2: Modelica representation of a simplified pendulum model, for illustration purposes.

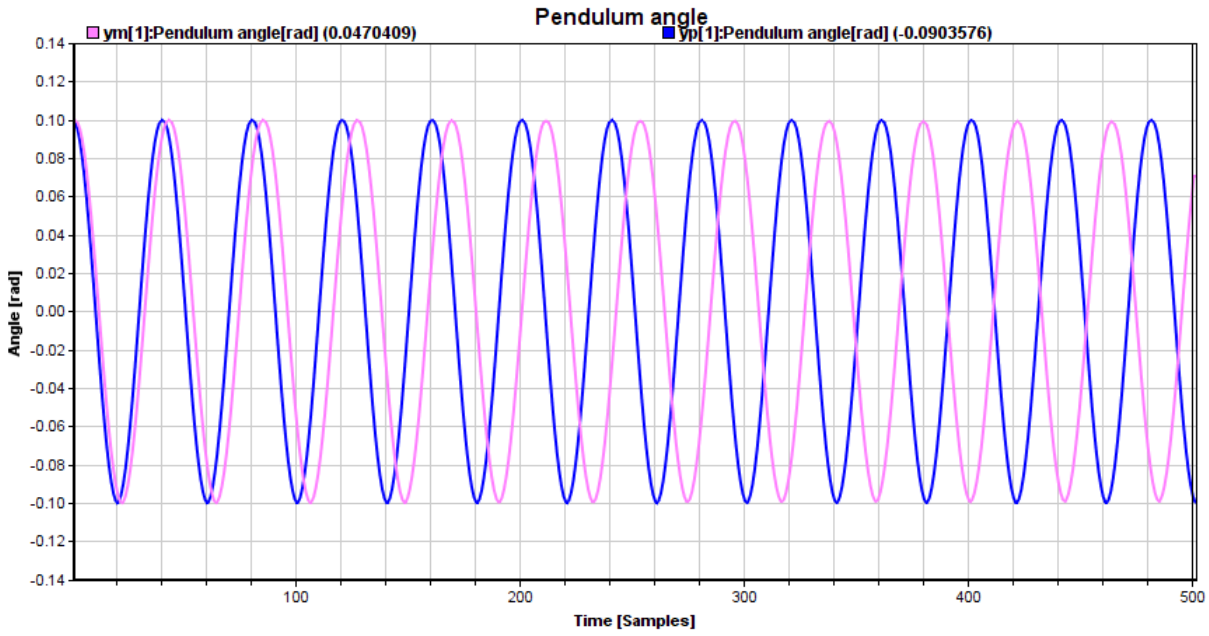


Figure F.3: ModelFit simulation of pendulum example, showing model predictions and measurements, using ballistic simulation of model.

pendulum, which is expected to follow the description of the model predictions, but with different parameter values. From this plot it is clear that the model predictions and measurements don't agree too well. The purpose of the parameter fitting, which will be demonstrated next, is that certain parameters in the model can be adjusted such that the model predictions agree with the

measure data. For this pendulum model, optimization will be carried out with respect to the pendulum mass and the length of the string. These two parameters are activated for optimization in the software, and the results are as given in Tab. F.1.

Table F.1: Optimally fitted parameters for the pendulum model.

Variable	**Original value**		**Optimal value**
Pendulum mass [kg]	1.0	→	0.979426
String length [m]	1.0	→	0.922690

Applying these changes gives a different behavior of the model in comparison with the experimental data. The angle for the pendulum is plotted again with the newly optimized parameter values, and this is shown in Fig. F.4. In this case, the model predictions and experimental measurements are virtually inseparable, and this illustrates the potential of this tool. It is important to emphasize that the actual (in this case physical) parameters of the problem may actually be different from what the ModelFit software suggests, but the result of the optimization is the best combination of parameter values to give agreement between the model predictions and measurements, according to the ModelFit software.

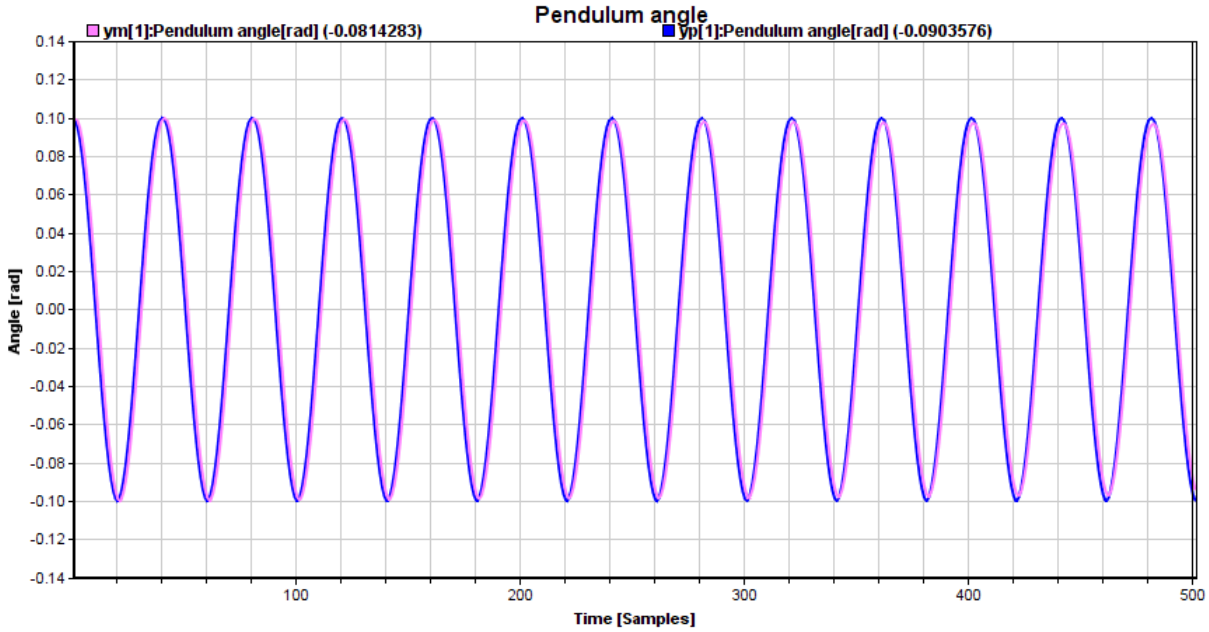


Figure F.4: ModelFit simulation of pendulum example, showing model predictions and measurements, with fitted parameters for the model.

This concludes the example on implementing a simple physical problem in the Modelica programming language and performing parameter fitting using the Cybernetica ModelFit software. The motivation, strategy and procedure for performing parameter fitting on the semi-batch model for emulsion copolymerization is analogous.