Jaquelino Nascimento da Rosa

MODELING AND SIMULATION OF A LAB-SCALE POLYMERIZATION REACTOR

Universidade de Coimbra

Jaquelino Nascimento da Rosa

# MODELING AND SIMULATION OF A LAB-SCALE POLYMERIZATION REACTOR

September, 2011

UNIVERSIDADE DE COIMBRA

Departamento de Engenharia Química
Faculdade de Ciências e Tecnologia
Universidade de Coimbra

# Modeling and Simulation of a Lab-Scale Polymerization Reactor

Dissertação apresentada à Faculdade de Ciências e Tecnologia da Universidade de Coimbra, com vista à obtenção do grau de Mestre em Engenharia Química.

Jaquelino Nascimento da Rosa

Orientadores:
Jorge Fernando Coelho
Lino de Oliveira Santos

Coimbra, Setembro de 2011

"The simplest explanation
is most likely the correct one."
**William of Ockham**

To my beloved teacher,
**Maria Margarida Lopes Figueiredo**

# Abstract

The main goal of this work is to study the thermal dynamics of a lab-scale vessel that can be used for polymerization experiments. It involves a study of the thermal dynamics of a 250 mL jacketed lab-scale glass reactor intended to be used in the future for batch polymerization reaction experiments. A mathematical model is developed and is used to simulate the thermal dynamics of the system. The mathematical model available in the literature for methyl methacrylate (MMA) solution free-radical polymerization is also considered in this simulation study. The numerical implementation of the models has been made in the MATLAB programming language. The simulation results of the lab-scale reactor show the heating and cooling performance, and a feedback control scheme using a split-range control strategy is tested as well to simulate the temperature setpoint tracking of the MMA polymerization mixture in the lab-scale reactor.

# Resumo

Neste trabalho é efectuado um estudo térmico de um reactor laboratorial em vidro, dotado de uma camisa de arrefecimento/aquecimento e com uma capacidade de 250 mL. Prentende-se no futuro utilizar este reactor para realizar ensaios experimentais de polimerização. É formulado um modelo matemático do reactor e é simulado numericamente o seu comportamento dinâmico. O estudo desenvolvido inclui também o modelo matemático para a polimerização de metacrilato de metilo (MMA) em solução em modeo descontínuo, disponibilizado na literatura cientÃfica. A implementação numérica dos modelos matemáticos é efectuada com recurso à linguagem de programação MATLAB. Os resultados de simulação demonstram o desempenho térmico do reactor, nomeadamente a sua capacidade de resposta quando é aquecido ou arrefecido. É também apresentado um estudo do reactor sob controlo por realimentação da temperatura da mistura do sistema de polimerização de MMA, em que é adoptada uma estratégia de controlo do tipo *split-range*.

# Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisors, Dr Jorge Coelho and Dr Lino Santos, who have supported me throughout my dissertation with their guidance, patience, motivation, enthusiasm, knowledge and friendship whilst allowing me the room to work in my own way.

I am truly indebted and thankful to Dr Nuno Oliveira for having provided very useful numerical data from Silva (2005).

I would like to gratefully and sincerely thank my high school teachers, Elisângela Semedo, Teodorino Carvalho, João Pedro Semedo, Augusto Borges, and Ângela Varela, for their continuous support during my dissertation and my graduate studies.

I would like to thank my mother, Hermínia Veiga, and my family for their support, encouragement and unwavering love.

I offer my regards and blessings to the whole DEQ community and all of those who supported me in any respect during the completion of this work.

# Nomenclature

| | | |
|---|---|---|
| $A$ | heat transfer area | $\mathrm{m}^2$ |
| $C_{\mathrm{p,j}}$ | jacket fluid heat capacity | $\mathrm{J\,kg^{-1}K^{-1}}$ |
| $C_{\mathrm{p,r}}$ | reaction medium heat capacity | $\mathrm{J\,kg^{-1}K^{-1}}$ |
| $d$ | impeller diameter | m |
| $D_{\mathrm{H}}$ | hydraulic diameter | m |
| $D_{\mathrm{i}}$ | reactor internal diameter | m |
| $D_{\mathrm{n}}$ | dead polymer concentration with $n$ repeating units | $\mathrm{mol\,m^{-3}}$ |
| $D_{\mathrm{o}}$ | reactor internal diameter | m |
| $f_{\mathrm{i}}$ | initiator efficiency factor | |
| $F_{\mathrm{j}}$ | jacket volumetric flow rate | $\mathrm{m^3 min^{-1}}$ |
| $f_{(\mathrm{m,n})}$ | weight fraction of polymer in chain length interval $[m, n]$ | |
| $g_{\mathrm{t}}$ | gel effect parameter | |
| $h_{\mathrm{i}}$ | internal heat transfer coefficient | $\mathrm{W\,m^{-2}K^{-1}}$ |
| $h_{\mathrm{o}}$ | external heat transfer coefficient | $\mathrm{W\,m^{-2}K^{-1}}$ |
| $I$ | initiator concentration | $\mathrm{mol\,m^{-3}}$ |
| $k$ | thermal conductivity | $\mathrm{W\,m^{-1}K^{-1}}$ |
| $K_{\mathrm{c}}$ | PID controller gain | |
| $k_{\mathrm{d}}$ | initiator decomposition rate constant | $\mathrm{min^{-1}}$ |
| $k_{\mathrm{fm}}$ | chain transfer to monomer rate constant | $\mathrm{m^3\,mol^{-1}min^{-1}}$ |
| $k_{\mathrm{fs}}$ | chain transfer to solvent rate constant | $\mathrm{m^3\,mol^{-1}min^{-1}}$ |
| $k_{\mathrm{i}}$ | primary radical formation rate constant | $\mathrm{m^3\,mol^{-1}min^{-1}}$ |
| $k_{\mathrm{p}}$ | propagation rate constant | $\mathrm{m^3\,mol^{-1}min^{-1}}$ |
| $k_{\mathrm{td}}$ | disproportionation termination rate constant | $\mathrm{m^3\,mol^{-1}min^{-1}}$ |
| $M$ | monomer concentration | $\mathrm{mol\,m^{-3}}$ |
| $mc$ | PID controller command signal | |
| $M_{\mathrm{t}}$ | total reaction mass | kg |
| $\overline{M}_{\mathrm{n}}$ | number average molecular weight | $\mathrm{kg\,mol^{-1}}$ |
| $\overline{M}_{\mathrm{w}}$ | weight average molecular weight | $\mathrm{kg\,mol^{-1}}$ |
| $N$ | impeller rotational speed | $\mathrm{rad\,min^{-1}}$ |
| $N_u$ | Nusselt number | |
| $P$ | total concentration of live polymer radicals | $\mathrm{mol\,m^{-3}}$ |
| $P_{\mathrm{n}}$ | live polymer radical concentration with $n$ repeating units | $\mathrm{mol\,m^{-3}}$ |
| $P_r$ | Prandtl number | |
| $R$ | primary radical concentration | $\mathrm{mol\,m^{-3}}$ |
| $R_e$ | Reynolds number | |
| $R_{\mathrm{H}}$ | hydraulic radius | m |
| $S$ | solvent concentration | $\mathrm{mol\,m^{-3}}$ |
| $T_{\mathrm{j}}$ | jacket fluid temperature | °C |

| | | |
|---|---|---|
| $T_{j,0}$ | jacket fluid initial temperature | °C |
| $T_r$ | reaction medium temperature | °C |
| $U$ | overall heat transfer coefficient calculation | $W\,m^{-2}K^{-1}$ |
| $U_{eff}$ | effective overall heat transfer coefficient calculation | $W\,m^{-2}K^{-1}$ |
| $V$ | reaction mixture volume | L |
| $v_f$ | free volume | |
| $v_{fcr}$ | critical free volume | |
| $V_j$ | jacket volume | $m^3$ |
| $w_i$ | initiator mass fraction | |
| $w_m$ | monomer mass fraction | |
| $w_{m,0}$ | monomer initial mass fraction | |
| $w_s$ | solvent mass fraction | |
| $w_p$ | polymer mass fraction | |
| $X$ | monomer conversion | |
| $y$ | measured value for the controlled variable | |
| $y_{sp}$ | controlled variable setpoint | |

***Greek Letters***

| | | |
|---|---|---|
| $\alpha$ | probability of propagation | |
| $\Delta H$ | heat of reaction | $J\,mol^{-1}$ |
| $\Delta t$ | sampling time | min |
| $\epsilon_k$ | controller error signal | |
| $\lambda_k$ | kth moment of dead polymer molecular weight distribution | |
| $\mu$ | reaction medium viscosity | Pa s |
| $\mu_w$ | reaction medium viscosity at wall temperature | Pa s |
| $\rho_m$ | monomer density | $kg\,m^{-3}$ |
| $\rho_p$ | polymer density | $kg\,m^{-3}$ |
| $\rho_s$ | solvent density | $kg\,m^{-3}$ |
| $\tau_D$ | derivative time | min |
| $\tau_I$ | integral time | min |
| $\phi_m$ | monomer volume fraction | |
| $\phi_p$ | polymer volume fraction | |
| $\phi_s$ | solvent volume fraction | |

**Acronyms**

| | |
|---|---|
| CLD | chain length distribution |
| MMA | methyl methacrylate |
| MWD | molecular weight distribution |
| ODE | ordinary differential equation |
| PID | proportional/integral/derivative |
| PMMA | Poly(methyl methacrylate) |
| PTFE | polytetrafluoroethylene |
| SISO | single input, single output |
| WCLD | weight chain length distribution |

# Contents

# List of Tables

# List of Figures

# 1. Introduction

Polymerization reactions pose some specific challenges in contrast to ordinary reaction systems through the necessity of, not only monitoring the conversion, but also the polymer molecular weight distribution (MWD) has to be maintained in order to produce polymers of acceptable quality. The strong nonlinearities of polymerization reactors make it even more difficult to control a polymerization reactor (Adebekun and Schork, 1989).

The solution polymerization of methyl methacrylate (MMA) in a lab-scale batch reactor, proceeded by the free-radical mechanism is the system studied in this work, with ethyl acetate as the solvent and 2,2'-azobis(2-methylbutanenitrile) as the radical initiator. This system was chosen due to great versatility of poly(methyl methacrylate) (PMMA), providing it a wide range of applications, and also to the fact that a large amount of information can be found in the literature.

The number average molecular weight ($\overline{M}_{\mathrm{n}}$) and weight average molecular weight ($\overline{M}_{\mathrm{w}}$) are convenient polymer characterization parameters as they represent compact and convenient information about a polymerization system. $\overline{M}_{\mathrm{n}}$ is more sensitive to molecules of low molecular weight, while $\overline{M}_{\mathrm{w}}$ has the opposite behavior. The ratio between $\overline{M}_{\mathrm{w}}$ and $\overline{M}_{\mathrm{n}}$ is called polydispersity and it measures the breadth of the polymer MWD. Nevertheless, polymers with similar polydispersity value may exhibit significantly different MWD and, consequently, different end-use properties. Therefore, to fully characterize a polymer MWD a new method is needed and the method of finite molecular weight moments, presented in Crowley and Choi (1997a) can, conveniently, sort out this issue. This method approximates the infinite molecular weight domain by a number of finite chain length intervals, each one containing a determined weight fraction of polymer. The number of intervals is established so that a good resolution of MWD is achieved, and the maximum chain length is chosen so that the chain length domain encompasses at least 99.9% of polymer weight. With a polymerization kinetic model, molecular weight averages and molecular weight distribution can be determined by integrating the mass balance equations simultaneously with the method of molecular weight moments.

A reliable control system is mandatory in polymerization reactors since changes in MWD may reveal irreversible, but also to prevent thermal runaway. More challenges are added by the fact of the reaction taking place in a batch reactor, which is characterized by having a nonlinear behavior and time-varying characteristics unlike continuous reactors, as pointed out by Silva and Oliveira (2002). There is a need to remove the reaction heat due to the exothermic nature of MMA polymerization reaction (Adebekun and Schork, 1989), but heating is also required, especially at the beginning of reaction, to raise the temperature to its desired value. In order to ac-

complish both heating and cooling tasks, a split range control strategy is adopted, where the output of the controller is split to the hot and cold streams (Seborg et al., 2004). Split range controllers can have different arrangements, but for this work the cold stream is turned on when the controller signal is between 0 and 50% while the hot stream remains closed and when the controller signal is between 50 and 100% the hot stream is turned on while the cold stream is turned off (Section 3.2.1).

In batch polymerization reactors, the temperature is often maintained by manipulating the coolant temperature or its flow rate. Sometimes, to overcome the nonlinear nature of the process and the poor dynamics of heat removal, especially caused by gel effect, the control performance can be recovered by employing a cascade control strategy: a master controller which sets the reactor temperature setpoint for coolant temperature, and a slave controller to control the coolant temperature by manipulating the coolant flow rate into the jacket (Schork et al., 1993).

## 1.1. Thesis outline

Chapter 2 describes the batch solution free-radical polymerization of methyl methacrylate (MMA) system. The dynamic model is stated and its numerical implementation is developed. A comparison of the results obtained by simulation is made with experimental data available in (Crowley and Choi, 1997b), and simulated profiles obtained in (Silva, 2005). In Chapter 3 the description of the lab-scale vessel reactor is introduced together with a study of the thermal dynamics of the heating and cooling processes. These studies required the development of the lab-scale reactor model and of simulation programs written in MATLAB. Section 3.1 assesses the thermal behavior of the vessel without any reaction through several heating and cooling simulations. Here, several modeling approaches for the lab-scale vessel jacket are evaluated. In Section 3.2 the mathematical model of the lab-scale reactor is formulated to include the MMA polymerization reaction. Simulation results are presented with temperature control performed by a PID controller. A split-range control strategy that manipulates a hot and cold water stream entering the reactor jacket is considered in this study as well. Finally, the main conclusions and future work directions are given in Chapter 4.

# 2. Batch Solution Free-radical Polymerization of Methyl Methacrylate (MMA)

Poly(methyl methacrylate) (PMMA) is a transparent thermoplastic used in a wide range of fields including medicine, art and aesthetics. It is also often used as an alternative to glass due to its shatter-resistance property (Arora et al., 2010). This chapter focus on the simulation of a solution free-radical polymerization of methyl methacrylate. The reaction takes place in a batch reactor operating mode which is widely used in commercial production of polymer solutions of high-quality resins (Rantow and Soroush, 2005). The batch reactor model for this system is taken from Crowley and Choi (1997a). The results obtained by simulation are compared with simulation and experimental data from literature (Crowley and Choi, 1997b; Silva, 2005) in order to validate the numerical implementation. This simulation study addresses as well a single-input single-output (SISO) PID control system to control the reactor temperature to its desired value during the course of the polymerization reaction.

The resulting polymer in a polymerization process is characterized not only by its average molecular weight but also by its molecular weight distribution (MWD). The control of the polymer chain length distribution (CLD) and of the corresponding MWD is of paramount importance because of end-use properties such as tensile strength and impact strength that are strongly dependent on the MWD (Crowley and Choi, 1997a). For instance, polymers with distinct MWD have different melting points as well as different melted polymer flow properties (Ellis et al., 1988). The molecular weight distribution and the average molecular weight, which are the number average molecular weight ($\overline{M}_{\mathrm{n}}$) and the weight average molecular weight ($\overline{M}_{\mathrm{w}}$), can be calculated based on the polymerization kinetic model (e.g., see Table 2.1), and using the method of molecular weight moments described in Section 2.2.1.

## 2.1. MMA free-radical polymerization kinetics

The kinetic scheme of the free-radical polymerization of MMA is presented in Table 2.1. It comprises the initiation step, where the initiator (I) decomposes to form reactive radicals (R), the addition of monomer molecules (M) to the reactive polymer chain formed from the reactive radicals, the chain transfer to monomer and solvent, and the termination step where deactivation of polymer radicals occur and

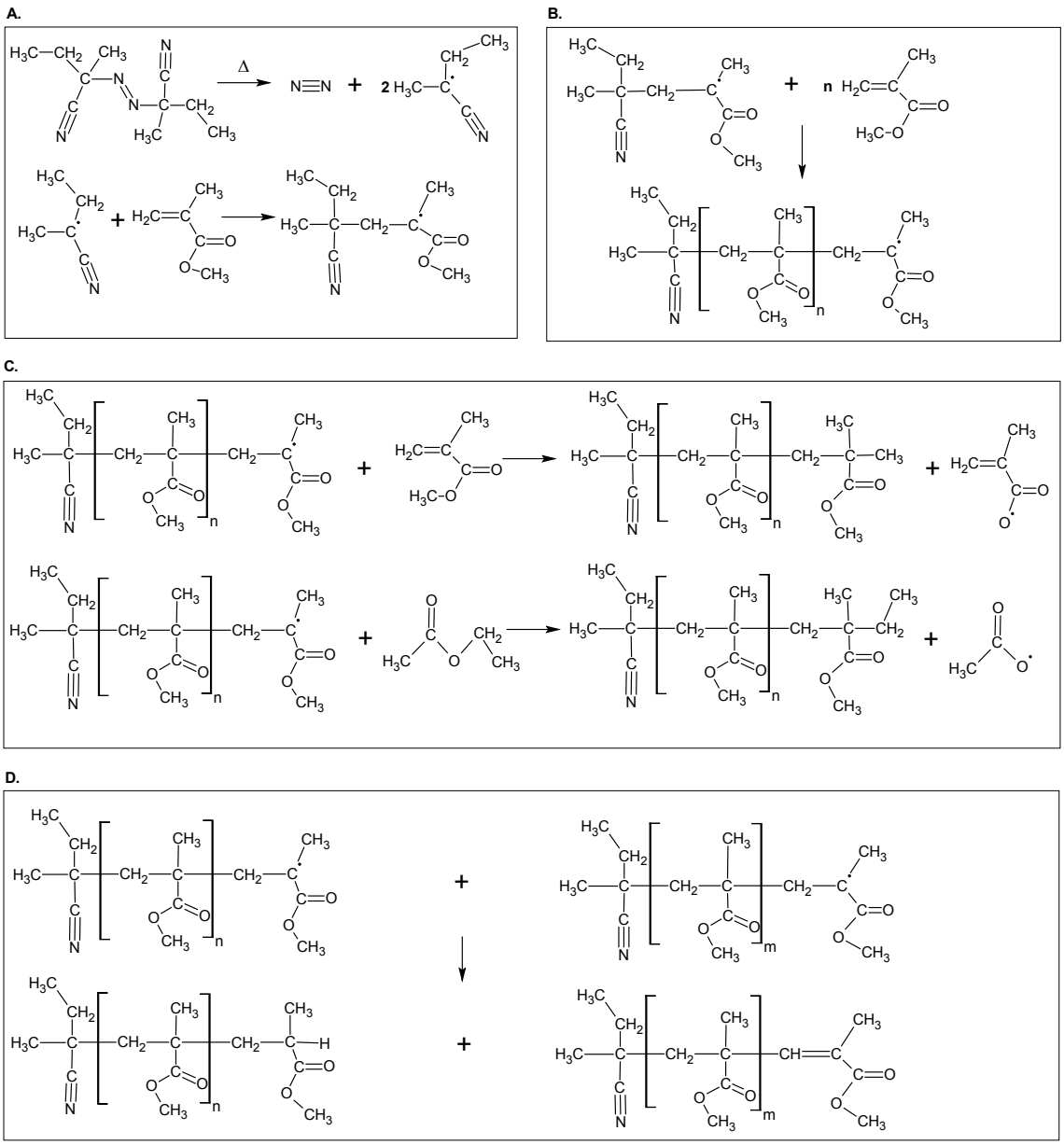


**Figure 2.1.:** *MMA free radical polymerization reaction scheme (A. - initiation, B. - propagation, C. - chain transfer to monomer and solvent, D. - termination by disproportionation). Adapted from Kranjk et al. (2001).*

only termination by disproportionation mechanism is considered. Figure 2.1 illustrates these steps showing the molecular structures of the involved compounds. The initiator used for the MMA radical polymerization reaction is the 2,2'-azobis(2-methylbutanenitrile) which decomposes thermally to form two radicals, releasing a nitrogen gas molecule in the process (Yamada and Zetterlund, 2002).

At high conversion levels, diffusion limitations become more significant than at the initial state of the reaction due to the increase in medium viscosity. Therefore, polymer radicals and monomer molecules become less mobile in the reaction medium, causing the termination and propagation rate constants ($k_{td}$ and $k_p$) to decrease (Crowley and Choi, 1997b). This phenomenon is usually referred to as "gel

**Table 2.1.:** *Kinetic scheme for the free-radical polymerization of MMA (Crowley and Choi, 1998). I is the initiator, R is the primary radical, $P_n$ is the live polymer radical with n repeating units, $D_n$ is the dead polymer with n repeating units, and S is the solvent.*

| | |
|---|---|
| Initiation: | $I \xrightarrow{k_d} 2R$ |
| | $R + M \xrightarrow{k_i} P_1$ |
| Propagation: | $P_n + M \xrightarrow{k_p} P_{n+1}$ |
| Chain transfer to monomer and solvent: | $P_n + M \xrightarrow{k_{fm}} D_n + P_1$ |
| | $P_n + S \xrightarrow{k_{fs}} D_n + P_1$ |
| Termination (disproportionation only): | $P_n + P_m \xrightarrow{k_{td}} D_n + D_m$ |

effect" and a correlation is provided in the literature to account for this condition into the disproportionation termination rate constant (see Appendix D).

## 2.2. Dynamic model formulation

Modeling is a process used in science and engineering to describe systems, and a good mathematical model must adequately describe the experimental data on a range of conditions as large as possible and must be easily manipulated in order to be useful in complex systems (Curteanu et al., 1998). A complete model for the MMA polymerization reaction system that can give information about MWD comprises mass and energy balance equations, and molecular weight moment equations. The model for this system can be found in several works (e.g., Crowley and Choi 1997b). The mathematical description of the molecular weight distribution time-dependency is given in Section 2.2.1. Then, the mass and energy balances are presented in Section 2.2.2.

### 2.2.1. Molecular weight averages and MWD from molecular weight moments

The molecular weight averages, namely the number average molecular weight ($\overline{M}_n$) and weight average molecular weight ($\overline{M}_w$) are insufficient to characterize a polymer as it is possible for two polymers with similar $\overline{M}_n$ and $\overline{M}_w$ to have distinct MWD and, consequently, substantially different physical and mechanical properties (Crowley and Choi, 1997a). Thus, there is a need for a method to compute the polymer MWD and the method of the molecular weight moments is the one considered in this study. For dead polymers, the molecular weight moments are defined by:

$$\lambda_k = \sum_{n=2}^{\infty} n^k D_n V \,, \tag{2.1}$$

where $\lambda_k$ represents the $k$th moment of a dead polymer, and $V$ is the volume of the mixture. Since the concentration of live polymer ($P_n$) is negligible compared to the concentration of dead polymer ($D_n$), then its moment is not considered for the calculation of molecular weights (Crowley and Choi, 1997b). It follows that,

$$\overline{M}_{\mathrm{n}} = M_0 \frac{\lambda_1}{\lambda_0} \qquad \text{and} \qquad \overline{M}_{\mathrm{w}} = M_0 \frac{\lambda_2}{\lambda_1} \,. \tag{2.2}$$

To calculate the MWD, the following function is used to represent the weight fraction of polymer with molecular weight in the interval $[m, n]$:

$$f_{(\mathrm{m,n})} = \frac{\sum_{i=m}^{n} i D_{\mathrm{i}} V}{\sum_{i=2}^{\infty} i D_{\mathrm{i}} V} \,. \tag{2.3}$$

The molecular weight distribution is time-dependent and the derivative of (2.3) with respect to time is given by:

$$\frac{\mathrm{d}f_{(\mathrm{m,n})}}{\mathrm{d}t} = \frac{1}{\lambda_1} \sum_{i=m}^{n} i \frac{\mathrm{d}[D_{\mathrm{i}} V]}{\mathrm{d}t} - \frac{f_{(\mathrm{m,n})}}{\lambda_1} \frac{\mathrm{d}\lambda_1}{\mathrm{d}t} \,. \tag{2.4}$$

From the kinetic scheme in Table 2.1, the kinetic rate equation for dead polymers of chain length $n$ can be derived:

$$\frac{\mathrm{d}[D_{\mathrm{n}} V]}{\mathrm{d}t} = V[k_{\mathrm{td}} P + k_{\mathrm{fm}} M + k_{\mathrm{fs}} S] P_{\mathrm{n}} \,. \tag{2.5}$$

The following equation defines the probability of propagation $\alpha$ (Crowley and Choi, 1997a),

$$\alpha = \frac{k_{\mathrm{p}} M}{k_{\mathrm{p}} M + k_{\mathrm{fm}} M + k_{\mathrm{fs}} S + k_{\mathrm{td}} P} \,. \tag{2.6}$$

The summation term in (2.4) is determined by manipulating (2.5) and (2.6) such that

$$\sum_{i=m}^{n} i \frac{\mathrm{d}[D_{\mathrm{i}} V]}{\mathrm{d}t} = V k_{\mathrm{p}} M \frac{1-\alpha}{\alpha} \sum_{i=m}^{n} i P_{\mathrm{i}} \,. \tag{2.7}$$

Using the following relationship,

$$P_{\mathrm{n}} = \alpha P_{\mathrm{n}-1} = (1-\alpha)\alpha^{n-1} P \,, \tag{2.8}$$

the term $\sum_{i=m}^{n} i P_{\mathrm{i}}$ in (2.7) is given by:

$$\sum_{i=m}^{n} i P_{\mathrm{i}} = \left[ \frac{m(1-\alpha)+\alpha}{1-\alpha} \right] \alpha^{m-1} P - \left[ \frac{(n+1)(1-\alpha)+\alpha}{1-\alpha} \right] \alpha^{n} P \,. \tag{2.9}$$

Finally, using the result in (2.9), equation (2.4) becomes

$$\frac{\mathrm{d}f_{(\mathrm{m,n})}}{\mathrm{d}t} = \frac{k_\mathrm{p} MV}{\lambda_1}\left(\left[\frac{m(1-\alpha)+\alpha}{\alpha}\right]\alpha^{m-1} - \left[\frac{(n+1)(1-\alpha)+\alpha}{\alpha}\right]\alpha^{n}\right)P - \frac{f_{(\mathrm{m,n})}}{\lambda_1}\frac{\mathrm{d}\lambda_1}{\mathrm{d}t}\,. \tag{2.10}$$

Further details on these developments can be found for instance in Crowley and Choi (1997b). To determine the evolution of the MWD, the equation (2.10) is discretized with respect to the chain length distribution. The resulting number of differential equations depends on the number of intervals considered for the weight chain length distribution (WCLD). Crowley and Choi (1997b) used 15 intervals that increase as the chain length is increased (2.11), by defining:

$$m = 2 + a(i-1)i\,, \tag{2.11a}$$

$$n = 1 + ai(i+1)\,, \tag{2.11b}$$

with $i = 1, ..., 15$. The parameter $a$ is selected such that that 99.9% of the resulting polymer has a chain length between 2 to $(1 + 15a(15 + 1))$, that is,

$$f(2, 1 + 15a(15 + 1)) = 0.999\,. \tag{2.12}$$

### 2.2.2. Mass and energy balances

The mass and energy balances for this polymerization system can be found, for instance, in the works of Silva (2005), Crowley and Choi (1997b, 1998), and Ellis et al. (1988). The partial mass balance to the monomer is expressed by:

$$\frac{\mathrm{d}X}{\mathrm{d}t} = \frac{(k_\mathrm{p} + k_\mathrm{fm})w_\mathrm{m} P}{w_{\mathrm{m},0}}\,, \tag{2.13}$$

where $X$ is the monomer conversion, $w_\mathrm{m}$ is the monomer mass fraction, $P$ is the total concentration of live polymer radicals, $w_{\mathrm{m},0}$ is the initial monomer mass fraction, and $k_\mathrm{p}$ and $k_\mathrm{fm}$ are respectively the propagation rate constant and the chain transfer to monomer rate constant. The partial mass balances to the initiator and solvent are given by:

$$\frac{\mathrm{d}w_\mathrm{i}}{\mathrm{d}t} = -k_\mathrm{d} w_\mathrm{i}\,, \tag{2.14a}$$

$$\frac{\mathrm{d}w_\mathrm{s}}{\mathrm{d}t} = -k_\mathrm{fs} w_\mathrm{s} P\,, \tag{2.14b}$$

where $w_\mathrm{i}$ is the initiator mass fraction, and $w_\mathrm{s}$ is the solvent mass fraction. The molecular weight moment equations are also included to allow the calculation of molecular weight averages and MWD. The differential equations for the the zero, first, and second order moment of dead polymer MWD ($\lambda_0$, $\lambda_1$, and $\lambda_2$) are defined

by:

$$\frac{\mathrm{d}\lambda_0}{\mathrm{d}t} = k_\mathrm{p} M V (1-\alpha) P\,, \tag{2.15a}$$

$$\frac{\mathrm{d}\lambda_1}{\mathrm{d}t} = k_\mathrm{p} M V (2-\alpha) P\,, \tag{2.15b}$$

$$\frac{\mathrm{d}\lambda_2}{\mathrm{d}t} = \frac{k_\mathrm{p} M V (\alpha^2 - 3\alpha + 4) P}{1-\alpha}\,. \tag{2.15c}$$

The energy balances to the reaction mixture and the jacket result in the following equations:

$$\frac{\mathrm{d}T_\mathrm{r}}{\mathrm{d}t} = \frac{(-\Delta H) k_\mathrm{p} w_\mathrm{m} M_\mathrm{t} P - UA(T_\mathrm{r} - T_\mathrm{j})}{\rho_\mathrm{r} C_{\mathrm{p,r}} V}\,, \tag{2.16a}$$

$$\frac{\mathrm{d}T_\mathrm{j}}{\mathrm{d}t} = \frac{F_\mathrm{j} \rho_\mathrm{j} C_{\mathrm{p,j}} (T_{\mathrm{j},0} - T_\mathrm{j}) + UA(T_\mathrm{r} - T_\mathrm{j})}{\rho_\mathrm{j} C_{\mathrm{p,j}} V_\mathrm{j}}\,, \tag{2.16b}$$

where $T_\mathrm{r}$ and $T_\mathrm{j}$ are the reactor and jacket temperature, respectively.

Now, including (2.10) for MWD evaluation, the dynamic model of the batch solution free-radical polymerization of MMA is summarized into the following set of equations:

$$\frac{\mathrm{d}X}{\mathrm{d}t} = \frac{(k_\mathrm{p} + k_\mathrm{fm}) w_\mathrm{m} P}{w_{\mathrm{m},0}}\,, \tag{2.17a}$$

$$\frac{\mathrm{d}w_\mathrm{i}}{\mathrm{d}t} = -k_\mathrm{d} w_\mathrm{i}\,, \tag{2.17b}$$

$$\frac{\mathrm{d}w_\mathrm{s}}{\mathrm{d}t} = -k_\mathrm{fs} w_\mathrm{s} P\,, \tag{2.17c}$$

$$\frac{\mathrm{d}\lambda_0}{\mathrm{d}t} = k_\mathrm{p} M V (1-\alpha) P\,, \tag{2.17d}$$

$$\frac{\mathrm{d}\lambda_1}{\mathrm{d}t} = k_\mathrm{p} M V (2-\alpha) P\,, \tag{2.17e}$$

$$\frac{\mathrm{d}\lambda_2}{\mathrm{d}t} = \frac{k_\mathrm{p} M V (\alpha^2 - 3\alpha + 4) P}{1-\alpha}\,, \tag{2.17f}$$

$$\frac{\mathrm{d}T_\mathrm{r}}{\mathrm{d}t} = \frac{(-\Delta H) k_\mathrm{p} w_\mathrm{m} M_\mathrm{t} P - UA(T_\mathrm{r} - T_\mathrm{j})}{\rho_\mathrm{r} C_{\mathrm{p,r}} V}\,, \tag{2.17g}$$

$$\frac{\mathrm{d}T_\mathrm{j}}{\mathrm{d}t} = \frac{F_\mathrm{j} \rho_\mathrm{j} C_{\mathrm{p,j}} (T_{\mathrm{j},0} - T_\mathrm{j}) + UA(T_\mathrm{r} - T_\mathrm{j})}{\rho_\mathrm{j} C_{\mathrm{p,j}} V_\mathrm{j}}\,, \tag{2.17h}$$

$$\begin{aligned}\frac{\mathrm{d}f_{(\mathrm{m,n})}}{\mathrm{d}t} = \frac{k_\mathrm{p} M V}{\lambda_1} \Bigg( \left[ \frac{m(1-\alpha)+\alpha}{\alpha} \right] \alpha^{m-1} - \\ \left[ \frac{(n+1)(1-\alpha)+\alpha}{\alpha} \right] \alpha^{n} \Bigg) P - \frac{f_{(m,n)}}{\lambda_1} \frac{\mathrm{d}\lambda_1}{\mathrm{d}t}\,,\end{aligned} \tag{2.17i}$$

**Table 2.2.:** *Parameters for the MMA polymerization reaction (Crowley and Choi, 1998).*

| $V$ | $V_j$ | $A$ | $F_j$ | $U_{eff}$ |
|---|---|---|---|---|
| 1.0 L | 3.5 L | 530.0 $cm^2$ | 1.0 $kgmin^{-1}$ | 817.0 $calK^{-1}min^{-1}$ |

**Table 2.3.:** *Initial conditions for the polymerization reaction (Crowley and Choi, 1997b).*

| $I_0$ | $\phi_{m,0}$ | $\phi_{s,0}$ |
|---|---|---|
| 0.046 $molL^{-1}$ | 0.5 | 0.5 |

In equation (2.17), the total concentration of live polymers ($P$) is determined by

$$P = \left( \frac{2 f_i k_d I}{k_{td}} \right)^{1/2}, \tag{2.18}$$

where $f_i$ is the initiator efficiency factor which corresponds to the fraction of primary radicals utilized in the chain growth (Kranjk et al., 2001).

Considering an ideal mixture and neglecting the initiator contribution, the total reactor mixture volume, $V$, is given by:

$$V = \left( \frac{w_m}{\rho_m} + \frac{w_p}{\rho_p} + \frac{w_s}{\rho_s} \right) M_t \,, \tag{2.19}$$

where $M_t$ is the total mass inside the reactor, $w_p$ is polymer mass fraction, and $\rho_m$, $\rho_p$, and $\rho_s$ are the monomer, polymer, and solvent densities, respectively.

### 2.2.3. Numerical implementation

The numerical implementation of the model described in Section 2.2 was tested and the obtained results were compared with experimental and simulated data available in the literature. Crowley and Choi (1997b) provide experimental data for the conversion evolution for this polymerization system, as well as data for the kinetic parameters, reactor characteristics, and initial conditions (see tables 2.2 and 2.3, and Appendix D). In these simulation studies the energy balance equations were not considered because the results published by Crowley and Choi (1997b) were obtained for a scenario where the reactor temperature is chosen as a task level manipulated variable.

In tables 2.2 and 2.3, $U_{eff}$ is the effective heat transfer coefficient (UA), $I_0$ is the initial concentration of the initiator, and $\phi_{m,0}$ and $\phi_{s,0}$ correspond to the monomer and solvent initial volume fraction, respectively.

With the information in these two tables, kinetic parameters in appendix D and physical properties of the compounds present in the reacting system in Appendix C, the dynamic model (2.17) was solved using the MATLAB ODE solver `ode15s`, with a sampling period of 1.0 min. The resulting monomer conversion profile is shown in Figure 2.2. As it is presented in Crowley and Choi (1997b), the reactor

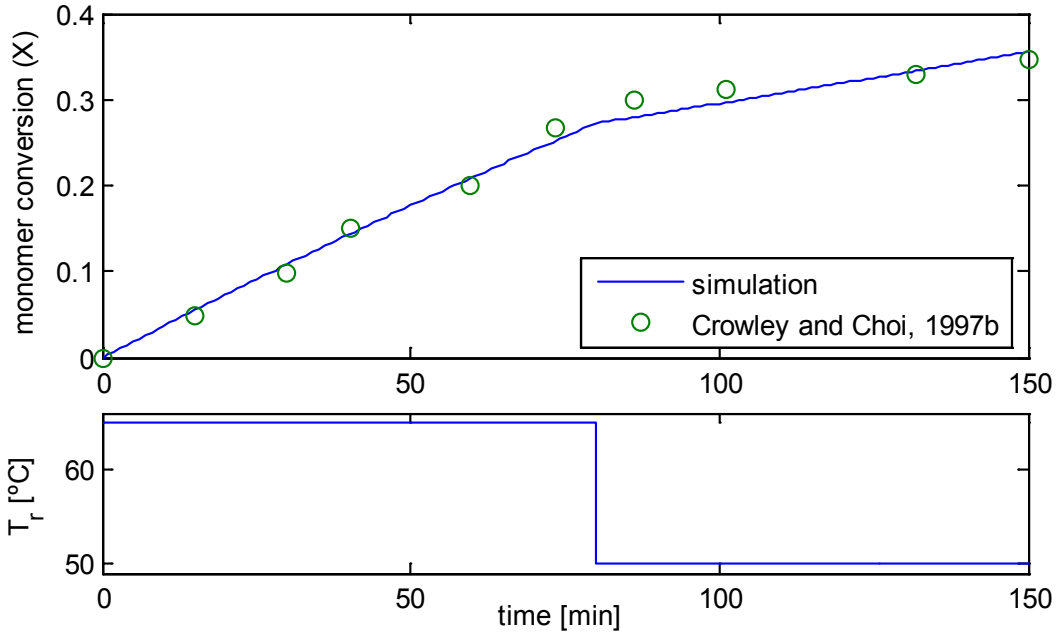


**Figure 2.2.:** *Monomer conversion model prediction.*

**Table 2.4.:** *Molecular weight averages for 50% conversion.*

| | $\overline{M}_n$ (kg mol$^{-1}$) | $\overline{M}_w$ (kg mol$^{-1}$) |
|---|---|---|
| model prediction | 55.9 | 111.6 |
| Silva, 2005 | 55.1 | 109.9 |

mixture temperature was initially set at a temperature of 65 °C. Then, when the monomer conversion is of 27%, the reactor mixture is cooled down to 50 °C in order to broaden the MWD (Crowley and Choi, 1997b). Figure 2.2 shows that the predicted monomer conversion profile is similar to the one presented by Crowley and Choi (1997b), which is in good agreement with their experimental monomer conversion values. Remark that the reactor temperature change is simulated here by a step change. Although this approach is somewhat unrealistic, from the profiles provided in Crowley and Choi (1997b) it follows that the temperature dynamics appear to be considerably faster than the monomer concentration dynamics. In Section 2.2.4 a PID loop is considered to control the reactor mixture temperature.

Another test was made to verify the correctness of the numerical implementation of the model. Here the dynamic profiles of monomer conversion, polymer molecular weight averages, and MWD were compared with the simulation results obtained for this system by Silva (2005). In this case the polymerization is carried out in isothermal conditions, with a constant reactor temperature of 60 °C. Figure 2.3 shows a good agreement between the polymer MWD values obtained by the numerical simulation developed in this work and the results obtained by Silva (2005). Similarities are also verified in molecular weight averages and monomer conversion which has an approximately linear profile (see Table 2.4 and figures 2.4 and 2.5).

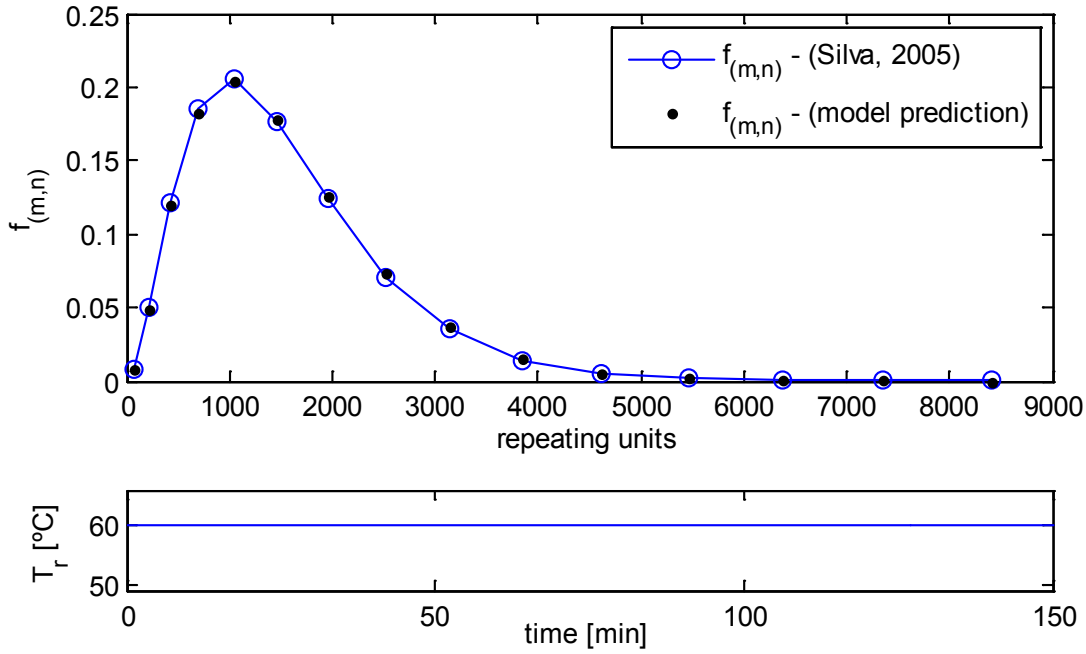


**Figure 2.3.:** *Model MWD prediction.*

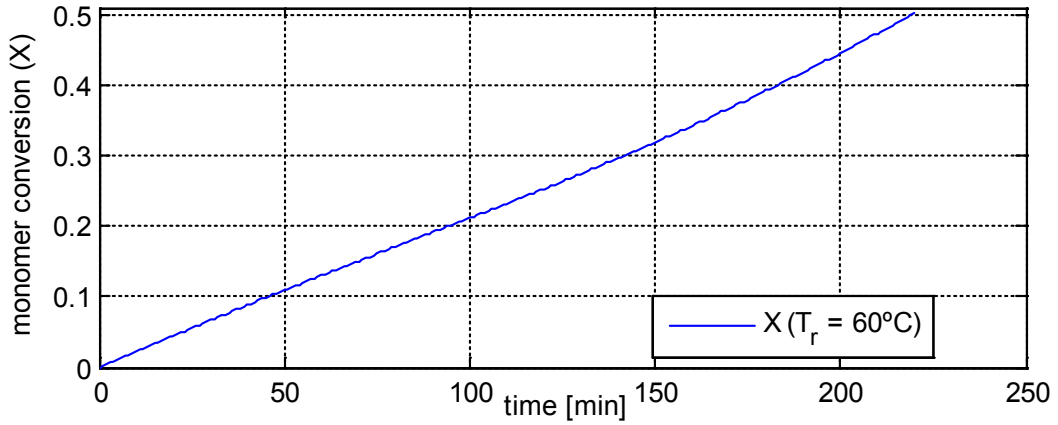


**Figure 2.4.:** *Model prediction for monomer conversion.*

### 2.2.4. Temperature control with a PID controller

Here, instead of the temperature step change illustrated in Figure 2.2, a simple feedback control approach using a PID controller is applied to change the reactor temperature by manipulating the temperature of the inlet stream to the jacket ($T_{j,0}$). The energy balance equations are now included, with the heat generation term also taken into account. The digital PID control law can be given by (Seborg et al., 2004),

$$m_{c,k} = m_{c,k-1} + K_c \left[ (\epsilon_k - \epsilon_{k-1}) + \frac{\Delta t}{\tau_I} \epsilon_k + \frac{\tau_D}{\Delta t} (\epsilon_k - 2\,\epsilon_{k-1} + \epsilon_{k-2}) \right] , \qquad (2.20)$$

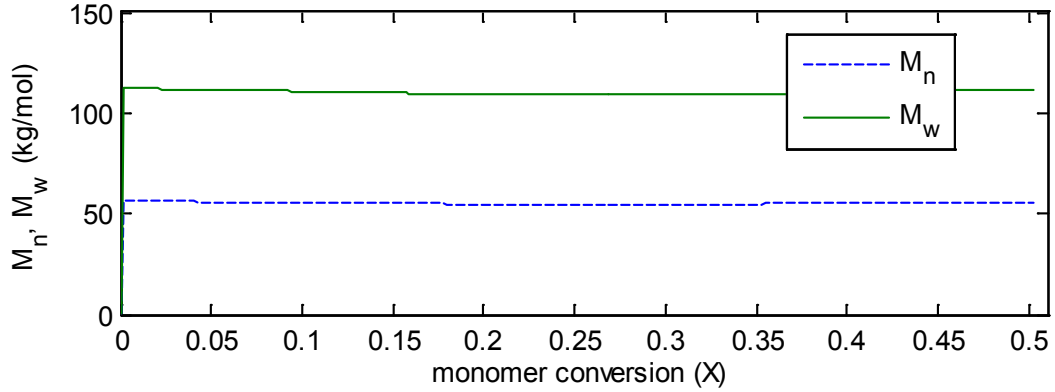


**Figure 2.5.:** *Model prediction for molecular weight averages.*

**Table 2.5.:** *PID controller parameters: controlled variable - $T_r$; manipulated variable - $T_{j,0}$.*

| $\Delta t$ | $K_c$ | $\tau_I$ | $\tau_D$ | $m_{c,max}$ | $m_{c,min}$ |
|---|---|---|---|---|---|
| 0.2 *min* | 52.5/2 | 0.93×3 min | 0.90/3 min | 80.0 °C | 25.0 °C |

which is known as the velocity form of the digital PID controller. In (2.20) $\Delta t$ is the sampling time period, $K_c$ is the controller proportional gain, $\tau_I$ is the integral time, $\tau_D$ is the derivative time, and $m_c$ is the manipulated variable. The error signal at the sampling time $k$, $\epsilon_k$, is defined by:

$$\epsilon_k = y_{sp,k} - y_k \,, \tag{2.21}$$

where $y_{sp,k}$ is the set point and $y_k$ is the measured value for the controlled variable.
The expression for the digital PID controller in (2.20) can be improved by replacing the derivate of the error by the derivative of the measured variable. This is done in order to prevent "derivative kick" phenomena, defined by a immediate large change in the controller output caused by the derivative term when a large step change is made in the setpoint. Thus, the digital PID control equation becomes:

$$m_{c,k} = m_{c,k-1} \; + \; K_c \left[ (\epsilon_k - \epsilon_{k-1}) + \frac{\Delta t}{\tau_I} \epsilon_k - \frac{\tau_D}{\Delta t} (y_k - 2\, y_{k-1} + y_{k-2}) \right] . \tag{2.22}$$

For more details about this subject, see for instance (Seborg et al., 2004).

Table 2.5 contains the PID controller parameters, where $mc_{max}$ and $mc_{min}$ represent the manipulated variable saturation points. The controller was tuned using the trial and error method explained in Seborg et al. (1989):

- The tuning process starts by eliminating integral and derivative action, that is: $\tau_I = \infty$ and $\tau_D = 0$;

- The next step involves $K_c$ tuning by starting from a small value (e.g., 0.5) and slowly increasing it until continuous cycling occurs after a small set point change. $K_c$ is then reduced by a factor of two (see Figure 2.6);

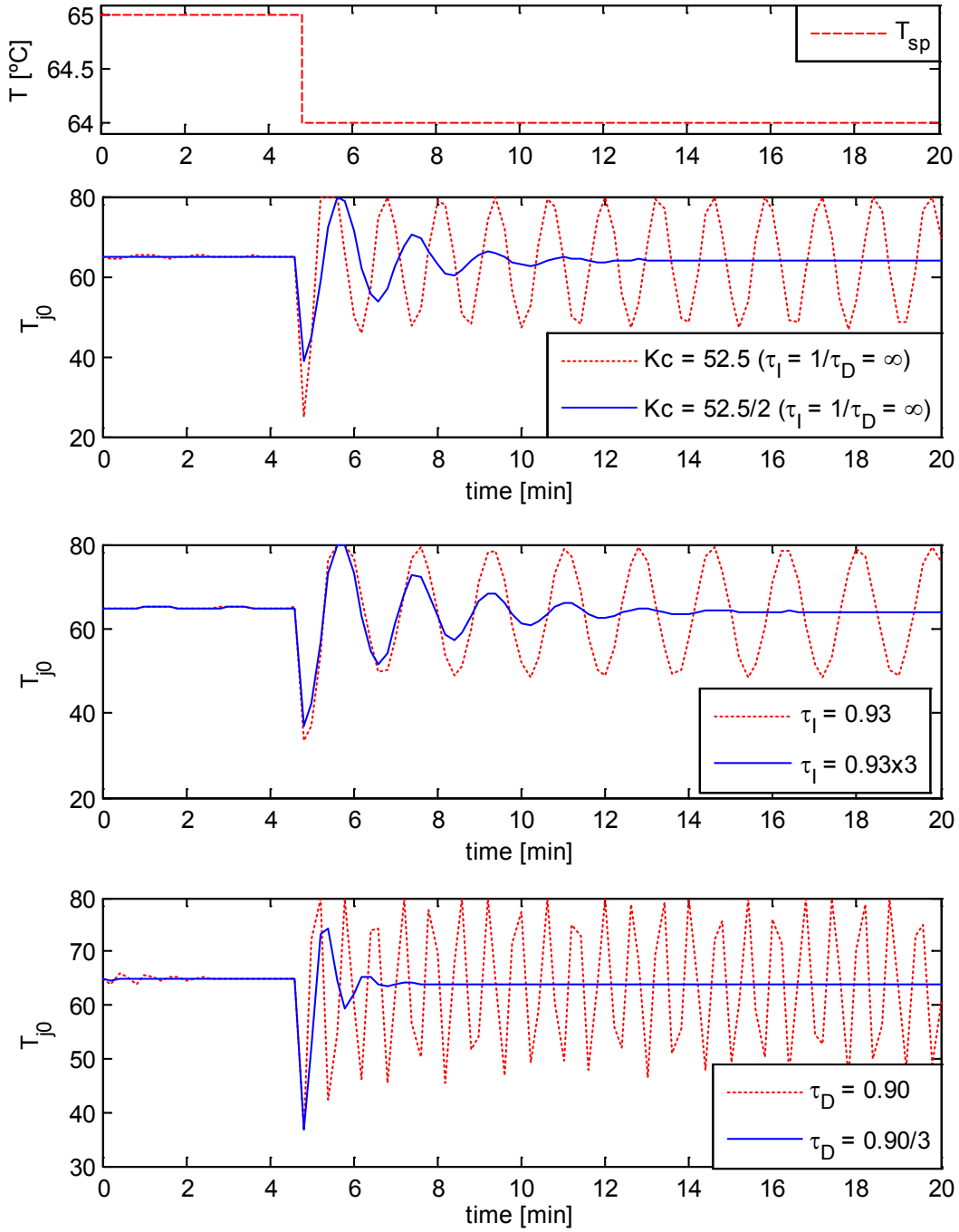


**Figure 2.6.: PID controller tuning.** *$K_c$ - Proportional gain, $\tau_I$ - Integral time, and $\tau_D$ - Derivative time.*

- Now the integral action is turned on by decreasing $\tau_I$ until continuous cycling occurs when it is then set three times the last value for $\tau_I$;
- Finally derivative action is recovered by increasing $\tau_D$ until continuous cycling occurs when it is then set three times lower than the last value.

Although the adoption of this control tuning strategy for a batch operation unit may be questionable, as it is explained along the simulation results discussion, this ap-

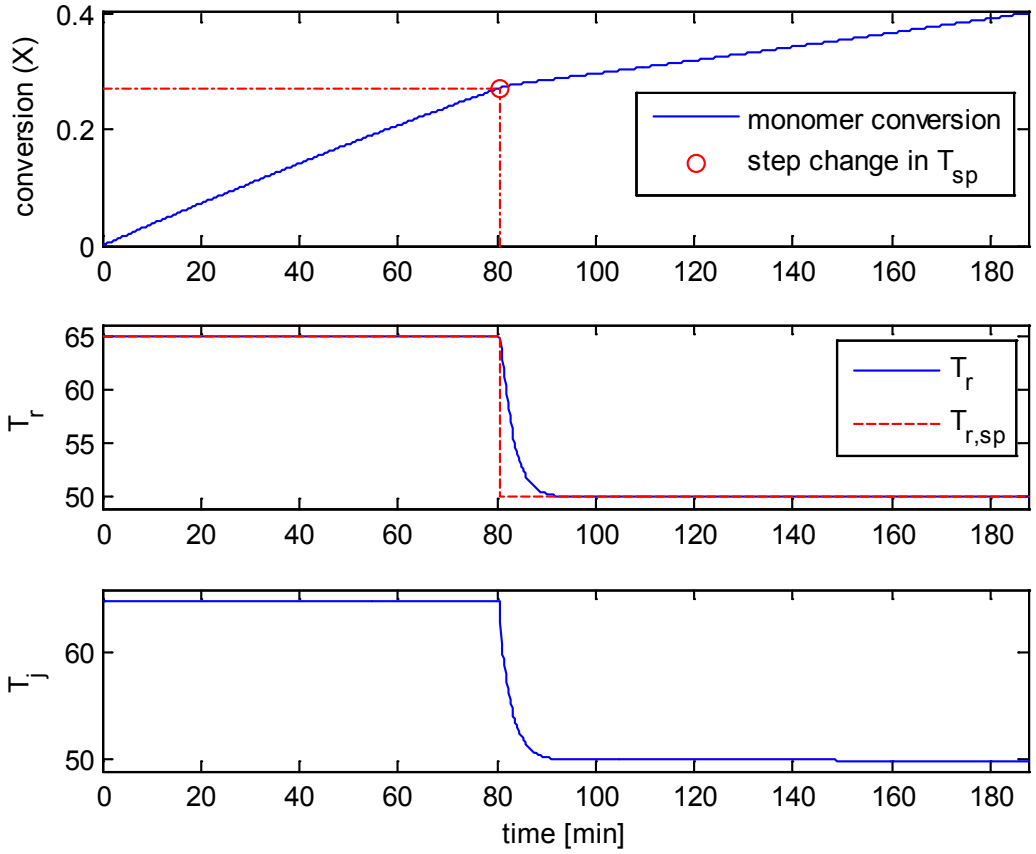


**Figure 2.7.:** *PID control of reactor temperature in the MMA batch solution polymerization reactor.*

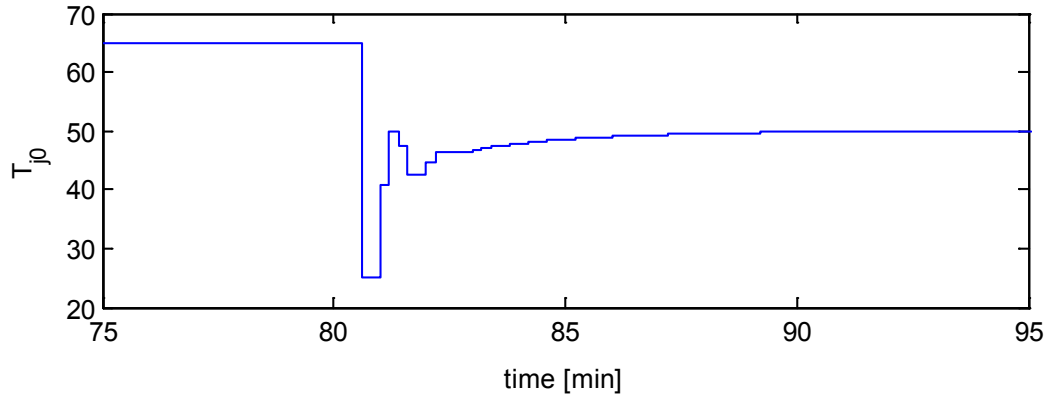


**Figure 2.8.:** *Manipulated variable ($T_{j,0}$) dynamics under the PID control.*

proach is quite acceptable in the context of this particular system. The selection of the sampling period, $\Delta t$, was also done by trial and error since there is not any systematic method to do so in batch processes. As observed by Seborg et al. (2004), the selection of the sampling period is more of an art than a science, even for continuous processes.

Figures 2.7 and 2.8 show the dynamics of the reactor temperature ($T_r$) and of the

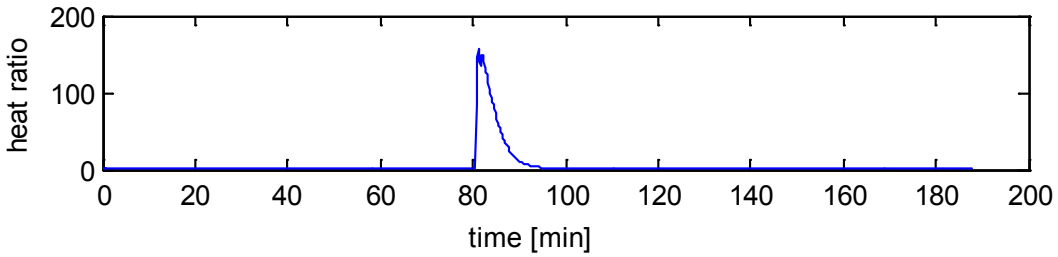


**Figure 2.9.:** *Ratio between the heat transfered and the reaction heat.*

manipulated variable ($T_{j,0}$) under the action of the PID controller. The temperature control, namely the adopted PID tuning strategy, can be performed in a similar fashion as in a continuous process, due to the fact that the released reaction heat is about 200 times smaller than the heat transfered between the jacket and the reaction mixture, as it can be observed in Figure 2.9.

One can observe from Figure 2.7 that there is no overshooting in the reactor temperature (controlled variable) which is a desirable situation because otherwise a targeted final polymer property (e.g., MWD) might not have been achieved. Therefore, it is safe to assume that the controller tuning is acceptable for this particular scenario, which is to decrease the reactor temperature from 65 to 50 °C when a 27% monomer conversion is reached.

# 3. Lab-scale Polymerization Reactor

The batch solution free-radical polymerization of methyl methacrylate (MMA) process described in Chapter 2 is now applied by simulation in the context of a jacketed lab-scale reactor vessel. As pointed out in Chapter 1, the main goal of this work is to study the thermal dynamics of a lab-scale vessel that can be used for polymerization experiments. This is motivated by the development of supporting knowledge in order to predict the thermal dynamic behavior of the lab-scale reactor vessel.

The contents of this chapter is organized as follows. First of all, a brief description of the dimensions and geometry of the lab-scale reactor are given. This is needed to define namely the heat transfer area. A description of the mathematical model of the lab-scale vessel dynamics is given, and an assessment of its heat transfer capabilities is done by simulation in batch operation mode with a water-water and a water-organic fluid system. A correlation to estimate the overall heat transfer coefficient is presented, and three different approximations to model the fluid jacked dynamics are tested – perfect mixed, plug flow, and lumped jacket model (Luyben, 1989). Results are given with profiles for the main process variables in open-loop mode. Finally, Section 3.2 describes the simulation of the closed-loop control of a batch solution free-radical polymerization of MMA, where a split-range control strategy is adopted to manipulate both a hot and cold water streams entering the reactor jacket.

#### Lab-scale reactor characteristics

The jacketed reactor vessel is supplied by Labglass Ltd and is considered to be the laboratory standard equipment. The vessel is manufactured from borosilicate with a PTFE tap, has a capacity of 250 ml, an overall height of 225 mm, with the inner and outer diameters of 75 and 105 mm, respectively, and a glass thickness of approximately 2.7 mm. A simplified diagram of the lab-scale reactor is given in Figure 3.1.

## 3.1. Lab-scale reactor open-loop thermal study

From the application of the energy conservation principle, and after several reasonable and well justified approximations, suitable for liquid systems, the simplified energy balance for a perfectly mixed non-reacting continuous tank system with a constant inlet flow rate, and a heat transfer rate to or from its surroundings, is of the form (Denn, 1987):

$$\rho V C_{\mathrm{p}} \frac{\mathrm{d}T}{\mathrm{d}t} = \rho F C_{\mathrm{p}} (T_{\mathrm{f}} - T) + Q \,, \tag{3.1}$$

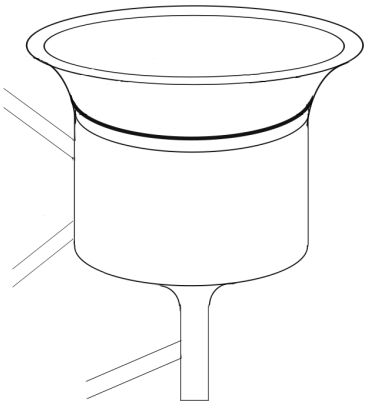

**Figure 3.1.:** *Lab-scale reactor.*

where $\rho$ is the density of the liquid mixture, $V$ is the tank liquid volume, $C_p$ is the heat capacity of the mixture, $T$ is the temperature of the fluid inside the vessel, $F$ is the inlet flow rate, $T_f$ is the feed stream temperature, and $Q$ is the heat flux to the heating/cooling system. For instance, in the particular case of a perfectly mixed jacketed liquid system,

$$Q = U\,A\,(T_j - T)\,, \tag{3.2}$$

where $U$ is the overall heat transfer coefficient, $A$ is the heat transfer area, and $T_j$ is the jacket fluid temperature. One emphasizes that (3.1) is obtained under the assumption that the properties $\rho$ and $C_p$ are constant.

Equation (3.1) is applied to both the fluid inside the reactor, and the fluid in the jacket. In the case of the lab-scale reactor operation in batch mode, the first term of the right hand side of the equation (3.1) vanishes (i.e., $F = 0$) in the energy balance equation related to the mass of the fluid inside the reactor.

### 3.1.1. Perfectly mixed jacket

In a perfectly mixed cooling or heating jacket model the temperature inside the jacket ($T_j$) is considered to be spatially constant. This is a good approximation for relatively high flow rates where the thermal fluid temperature does not change significantly as it goes through the jacket (Luyben, 1989). In this case, and under the assumption there are no changes in the composition of the fluid inside the tank, the dynamic model of the lab-scale batch vessel is given by

$$\rho_r V_r C_{p,r} \frac{dT_r}{dt} = UA(T_j - T_r)\,, \tag{3.3a}$$

$$\rho_j V_j C_{p,j} \frac{dT_j}{dt} = \rho_j F_j C_{p,j}(T_{j,0} - T_j) + UA(T_r - T_j)\,, \tag{3.3b}$$

where $U$ is the overall heat transfer coefficient, $A$ is the heat transfer area, and the subscripts $r$ and $j$ refer to the property that belongs to the fluid inside the reactor and inside the jacket, respectively. Also, the model 3.3 is derived under the assumption that there are no heat losses.

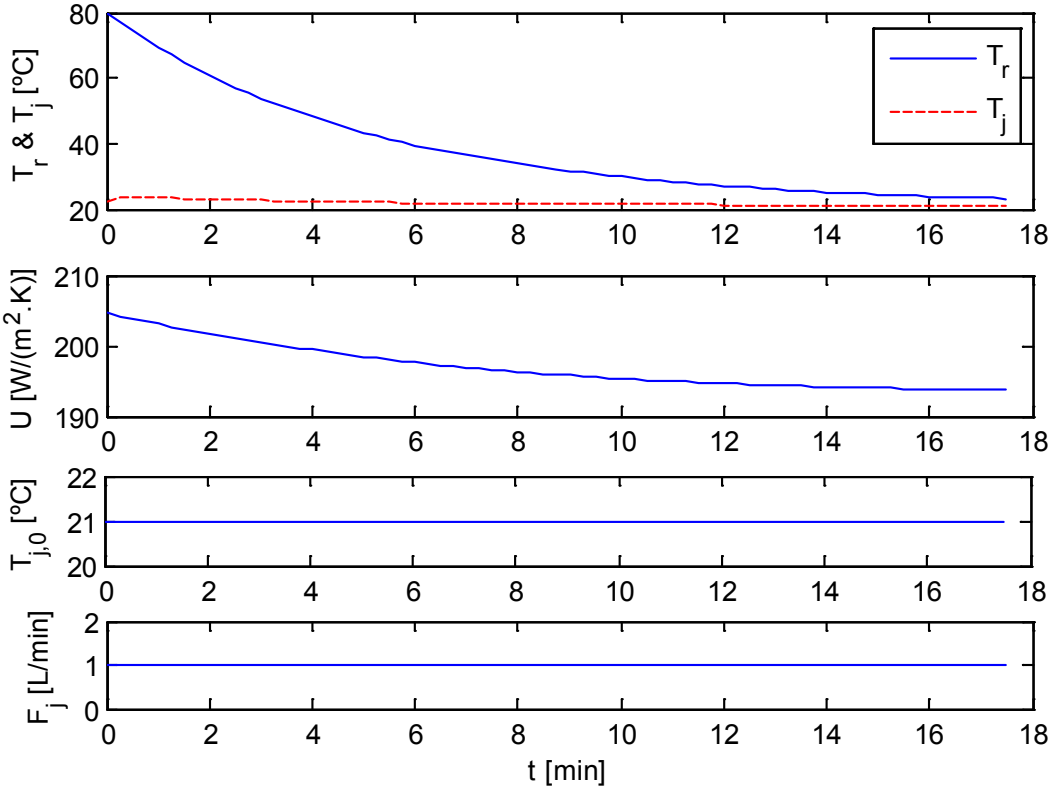


**Figure 3.2.: Water-water system.** *$T_r$ and $T_j$ profiles for the cooling simulation.*

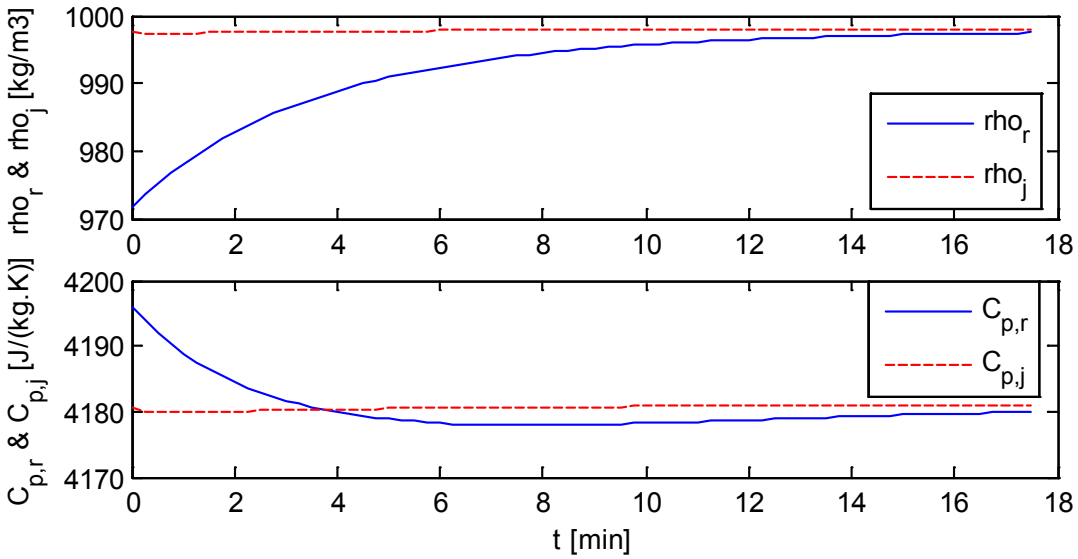


**Figure 3.3.: Water-water system.** *Physical properties of the fluids inside the reactor and jacket for the cooling simulation.*

### Water - Water system

A simulation with water in both the reactor and the jacket was carried out in MATLAB, using `ode15s` as the integrator and a sampling period of $\Delta t = 15\,\text{s}$, for cooling

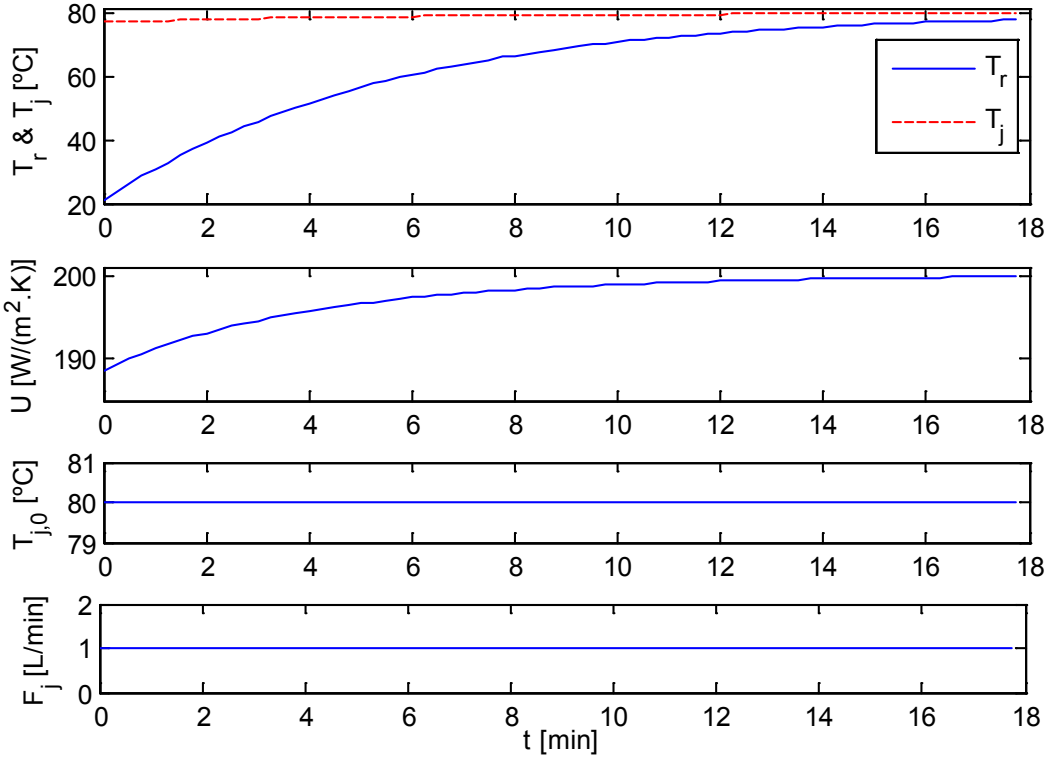


**Figure 3.4.: Water-water system.** *$T_r$ and $T_j$ profiles for the heating simulation.*

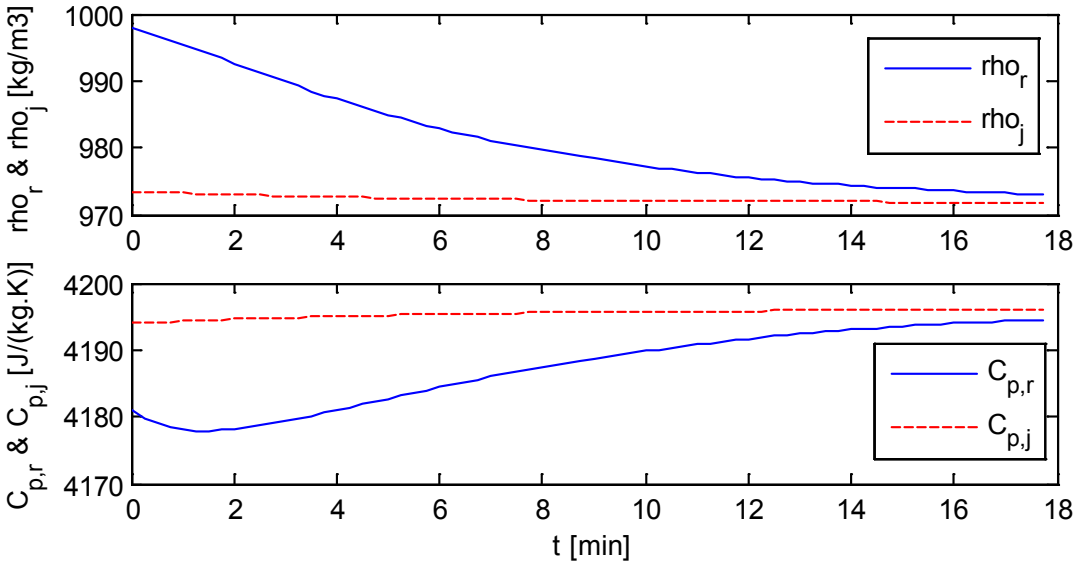


**Figure 3.5.: Water-water system.** *Physical properties of the fluids inside the reactor and jacket for the heating simulation.*

and heating scenarios (figures 3.2 and 3.4, respectively). The calculation of the overall heat transfer coefficient, heat transfer area, and fluid properties is described respectively in Appendix A, B, and C. Figure 3.2 represents the case where there is an excess of heat inside the reactor and, therefore, there is a need to remove it. The

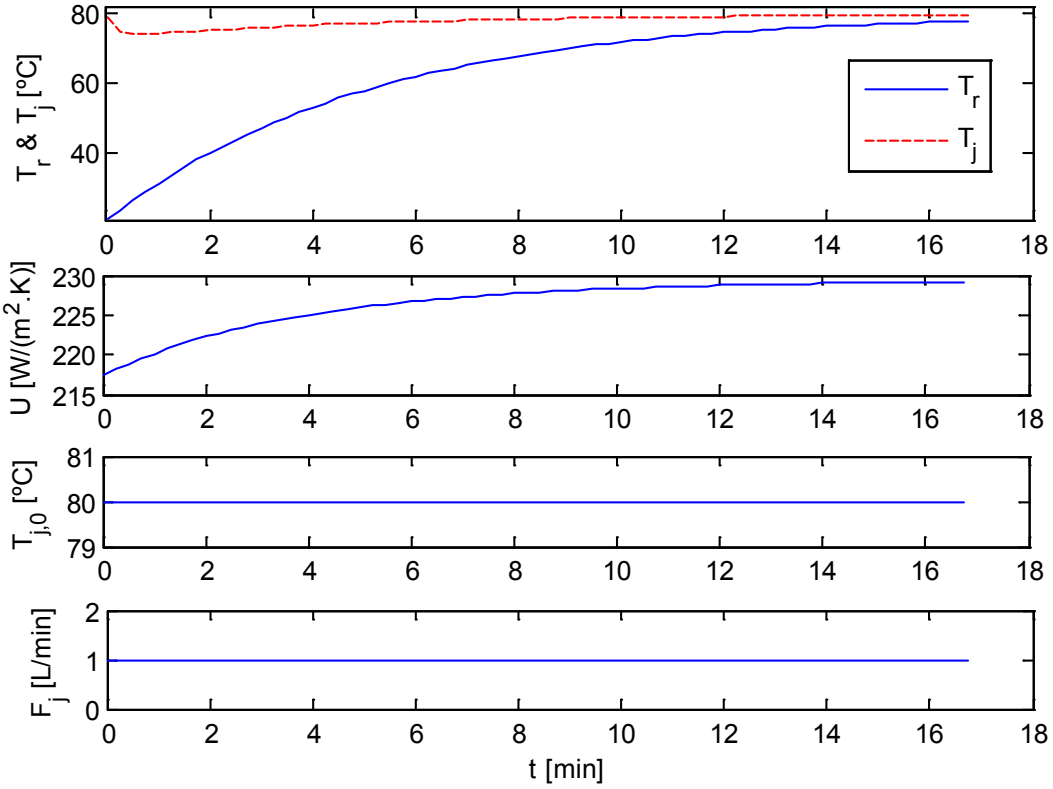


**Figure 3.6.: Water-organic system.** *$T_r$ and $T_j$ profiles for the heating simulation.*

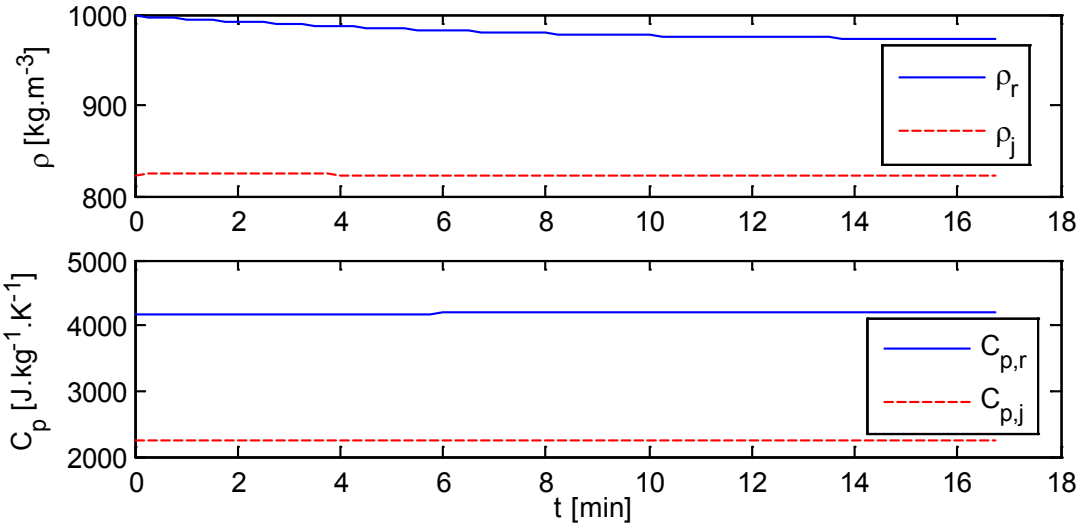


**Figure 3.7.: Water-organic system.** *Physical properties of the fluids inside the reactor and jacket for the heating simulation.*

reactor was considered to be initially at 80°C and the cooling water at 21°C. The profiles in figure 3.2 show that the tank system temperature stabilizes after around 18 minutes. The stabilization criterion adopted in the algorithm is a change in $T_r$ of less than 0.1°C between the temperatures of two successive sampling time instants.

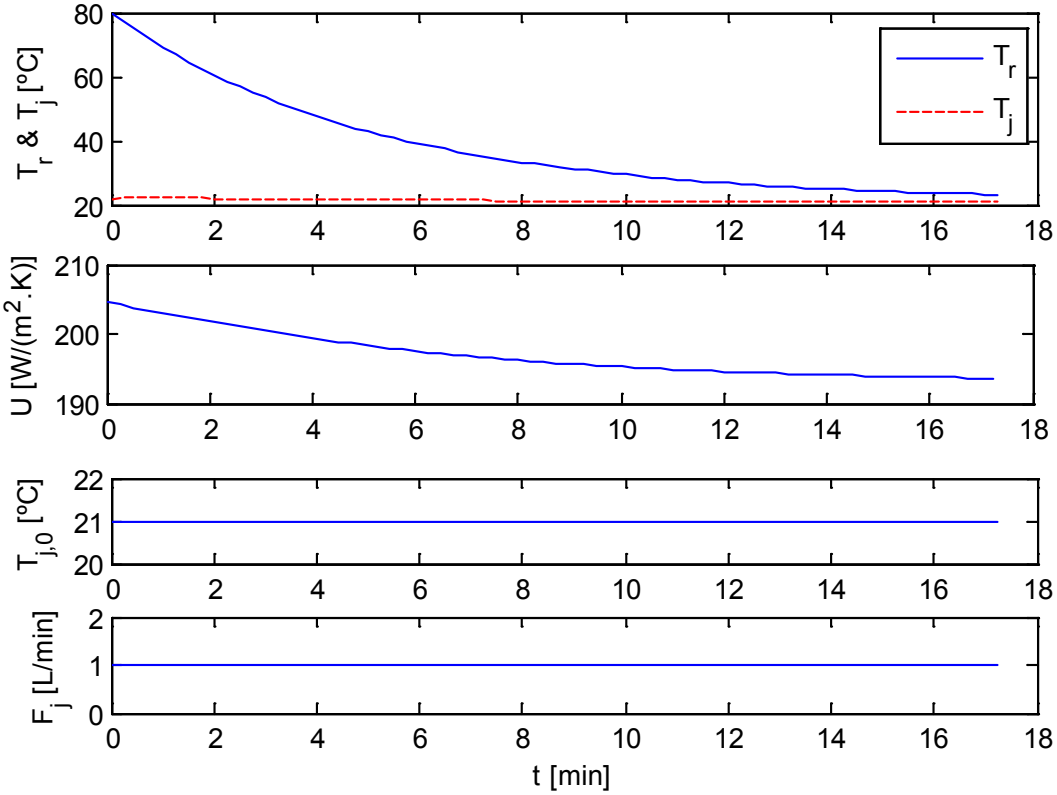


**Figure 3.8.: Water-water system.** *Plug flow cooling jacket with the cooling fluid at 80°C.*

On the other hand, Figures 3.4 illustrates the case where there is a need to provide heat to the reactor. Here, the temperatures $T_r$ and $T_j$ were interchanged from the previous simulation, i.e., $T_r = 21°C$ and $T_j = 80°C$. Both the overall heat transfer coefficient, $U$, and the physical properties of the fluids, such as density and heat capacity, are considered temperature dependent in these simulations (see figures 3.3 and 3.5).

#### Water - Organic system

Figures 3.6 and 3.7 show a case where an organic fluid is utilized instead of water. The organic fluid is known as *Duratherm 450*, it has a specific heat of 2.202 kJ $kg^{-1}K^{-1}$ at 60°C and can be used for heating or cooling duties in a temperature range of 0 to 230°C (DURATHERM, 2011). One can conclude that in this lab-scale system there is not a great advantage in using the organic fluid ,nstead of water. However, if temperatures above 100°C are needed then there is a clear advantage over the use of water.

### 3.1.2. Plug flow cooling jacket

The plug flow model for the cooling jacket may reveal to be a better approximation for low flow rates where the temperature of the jacket fluid changes significantly

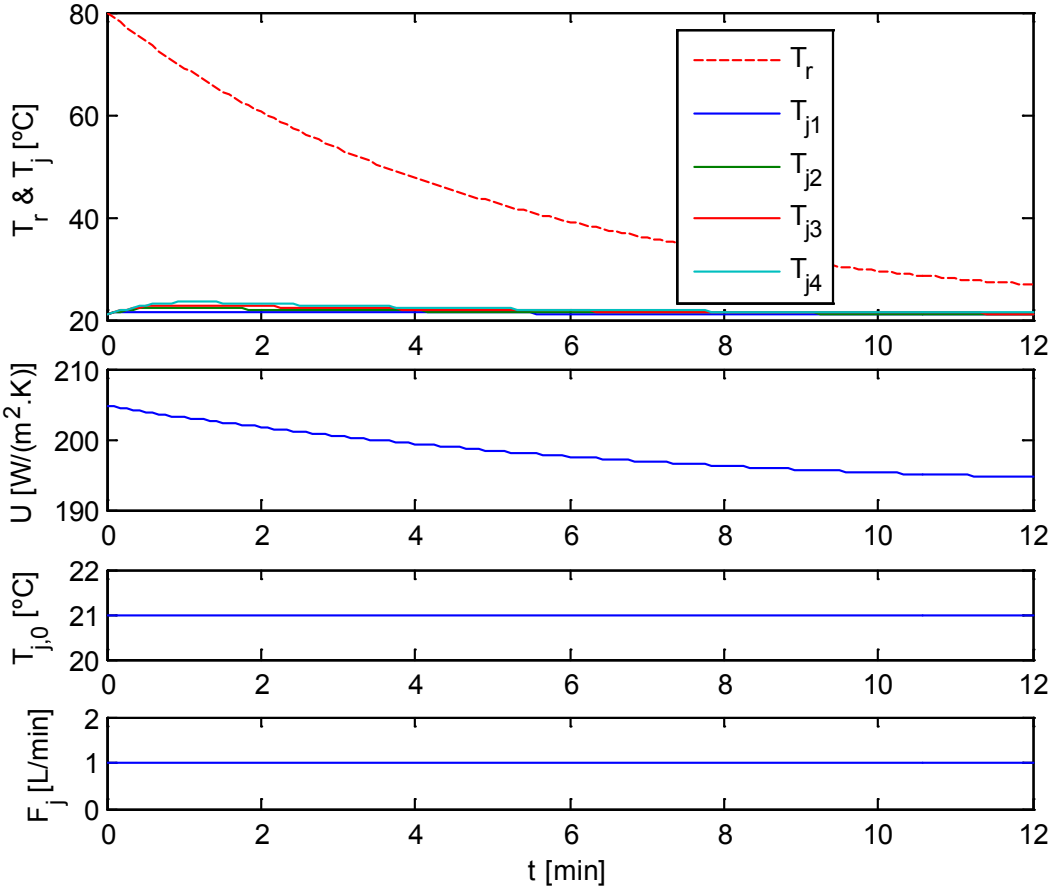


**Figure 3.9.: Water-water system.** *Lumped jacket with the cooling fluid at 80°C.*

along the jacket and the flow pattern is more similar to a plug flow in contrast to what was considered in 3.1.1. The model for this situation is presented by equation (3.4) and differs from the previous one because here the temperature inside the jacket ($T_\mathrm{j}$) is approximated by the average temperature between the inlet ($T_\mathrm{j,exit}$) and outlet ($T_\mathrm{j,0}$) cooling water (Luyben, 1989),

$$\rho_\mathrm{r} V_\mathrm{r} C_\mathrm{p,r} \frac{\mathrm{d}T_\mathrm{r}}{\mathrm{d}t} = -UA(T_\mathrm{r} - T_\mathrm{j}), \tag{3.4a}$$

$$\rho_\mathrm{j} V_\mathrm{j} C_\mathrm{p,j} \frac{\mathrm{d}T_\mathrm{j}}{\mathrm{d}t} = \rho_\mathrm{j} F_\mathrm{j} C_\mathrm{p,j}(T_\mathrm{j,0} - T_\mathrm{j,exit}) + UA(T_\mathrm{r} - T_\mathrm{j}), \tag{3.4b}$$

$$T_\mathrm{j} = \frac{T_\mathrm{j,0} + T_\mathrm{j,exit}}{2}, \tag{3.4c}$$

where $T_\mathrm{j,0}$ and $T_\mathrm{j,exit}$ are the inlet and outlet cooling fluid temperature, and $T_\mathrm{j}$ in this case represents an average jacket temperature. Figure 3.8 shows, however, for these particular set of conditions, $T_\mathrm{j}$ profiles are similar and the stabilization is reached approximately at the same time as in the case of the perfectly mixed jacket model (see Figure 3.2).

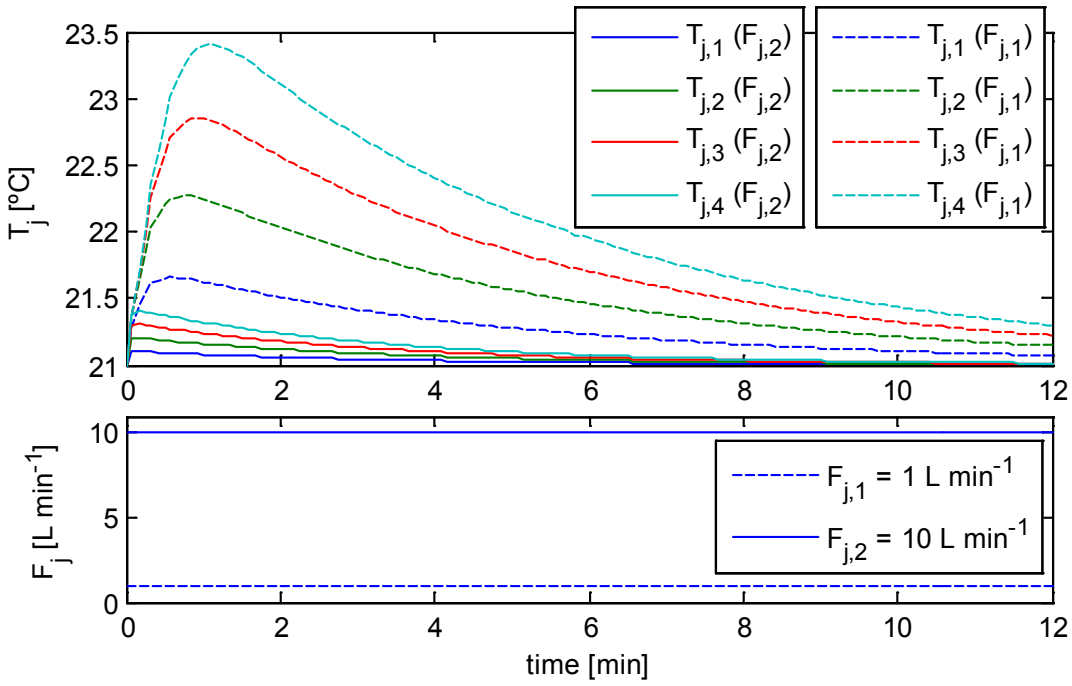


**Figure 3.10.: Water-water system.** *Lumped temperatures ($T_{j,i}$, $i = 1, 4$).*

### 3.1.3. Lumped jacket model

In this model, the jacket is broken up into a number of perfectly mixed "lumps", which means that the temperature inside each lump is uniform, although its value may be changing in time (Luyben, 1989),

$$\rho_r V_r C_{p,r} \frac{dT_r}{dt} = \sum_{i=1}^{n} \frac{1}{n} UA(T_r - T_{j,i}), \tag{3.5a}$$

$$\frac{1}{n}\rho_j V_j C_{p,j} \frac{dT_{j,1}}{dt} = \rho_j F_j C_{p,j}(T_{j,0} - T_{j,1}) + \frac{1}{n} UA(T_r - T_{j,1}), \tag{3.5b}$$

$$\vdots$$

$$\frac{1}{n}\rho_j V_j C_{p,j} \frac{dT_{j,n}}{dt} = \rho_j F_j C_{p,j}(T_{j,n-1} - T_{j,n}) + \frac{1}{n} UA(T_r - T_{j,n}), \tag{3.5c}$$

where $i = 1, \cdots, n$ represents each individual "lump" into which the system was divided. Figure 3.9 shows the simulation results for this model, where the jacket was divided into four lumps of equal volume. Figure 3.10 makes more visible the difference between the lump temperatures and evidences the relevance of jacket flow rate for this model. Two flow rates are utilized ($F_j = 1$ and $F_j = 10\,\mathrm{L\,min^{-1}}$) and it is quite visible that this model is far more relevant for low flow rates where the convective heat transfer mechanism is less significant.

#### Response time analysis

Figure 3.11 represents the dependency of stabilization time on the flow rate ($F_j$) and on the cooling jacket fluid inlet temperature ($T_{j,0}$) . The limits for $T_{j,0}$ was established

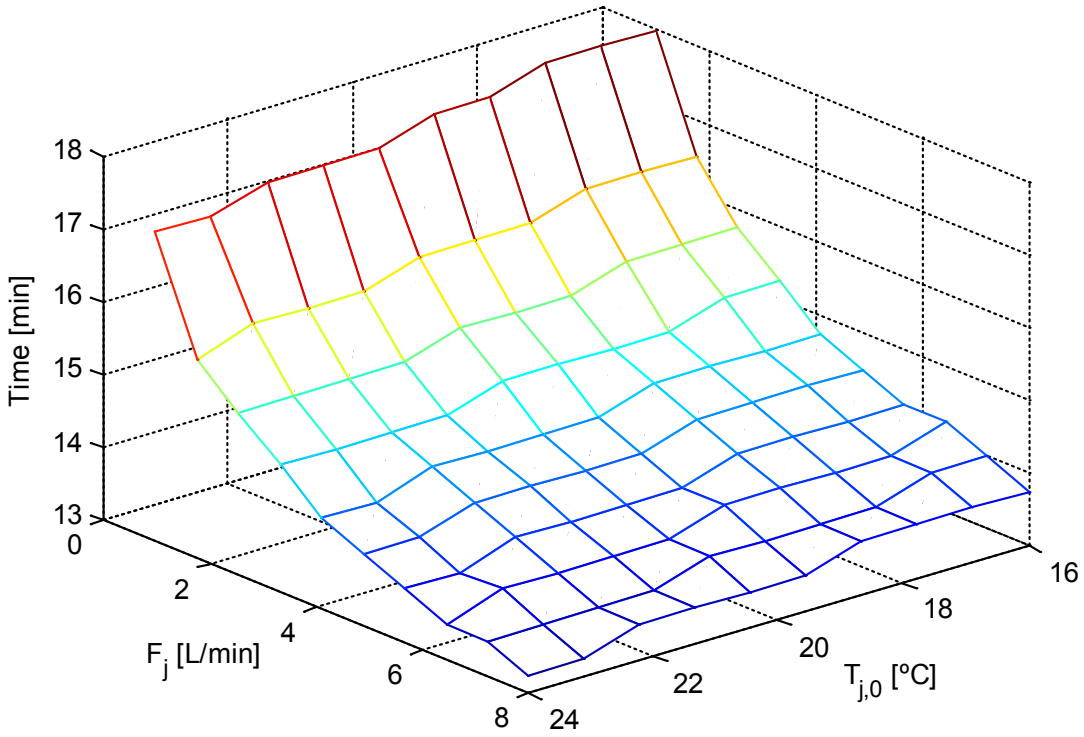


**Figure 3.11.:** *Stabilization time dependence on cooling fluid flow rate and temperature.*

based on tap water average temperature in the winter and in the summer. With the operating window considered, it is evident that the stabilization time depends more extensively on $F_j$ than on $T_{j,0}$.

#### Effect of the overall heat transfer coefficient

Under the current reactor design and operating conditions, the significance of considering a overall heat transfer coefficient dependent temperature is minimum on the reactor temperature dynamics, as the reactor temperature profile is essentially the same whether a constant or a temperature-dependent $U$ is considered for the simulation (Figure 3.12).

## 3.2. MMA polymerization in the lab-scale reactor

In Section 2.2 a mathematical model was developed that accounts for the MMA conversion calculation, the thermal dynamics through energy balance equations and the polymer weight distribution as well. Its implementation was validated in Section 2.2.3. Here this model is applied to simulate the MMA polymerization reaction within the context of the lab-scale reactor. Energy balances are also considered which means that all differential equations in (2.17) are included in the lab-scale reactor model formulation, neglecting heat losses to the surroundings. The parameters used for the MMA polymerization reaction are given in Table 3.2 and the kinetic constants

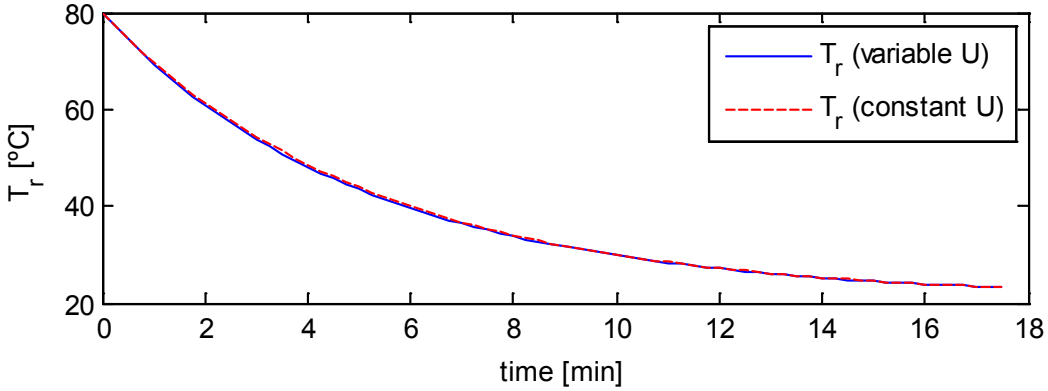


**Figure 3.12.:** *Variable overall heat transfer coefficient (U) significance.*

are the same as the ones presented in Section 2.2.3 (see also Appendix D). The initial conditions for the polymerization reaction are given in Table 2.3. The overall heat transfer coefficient is considered to be of 350 $\mathrm{W\,m^{-2}K^{-1}}$, which is a typical value for jacketed glass vessels (Fletcher, 1981).

### 3.2.1. Split-range control

A split-range control strategy is implemented in order to maintain the reactor temperature ($T_\mathrm{r}$) at the desired values. This type of control, which is essentially a PID control, is employed when several manipulated variables are used to control a single controlled variable (Seborg et al., 2004). For the current polymerization system, the use of this type of controller is justified by the necessity of both cooling and heating the reactor. Therefore, $T_\mathrm{r}$ is controlled by the hot and cold water flow rate in the jacketed reactor, according to the controller signal ($m_\mathrm{c}$) which varies from 0 to 100%. The split-range controller is designed such that when $m_\mathrm{c}$ is less than 50%, the hot stream is turned off while the cold stream is used to control $T_\mathrm{r}$. When $m_\mathrm{c}$ is greater than 50%, it is the other way around. It is necessary to introduce a modification in the jacket fluid energy balance (2.17h) in order to account for the hot and cold streams contributions, such that:

$$\frac{\mathrm{d}T_\mathrm{j}}{\mathrm{d}t} = \frac{F_\mathrm{hot}}{V_\mathrm{j}}(T_\mathrm{hot} - T_\mathrm{j}) + \frac{F_\mathrm{cold}}{V_\mathrm{j}}(T_\mathrm{cold} - T_\mathrm{j}) + \frac{UA(T_\mathrm{r} - T_\mathrm{j})}{\rho_\mathrm{j} C_{\mathrm{p,j}} V_\mathrm{j}}, \tag{3.6}$$

where $T_\mathrm{hot}$, $T_\mathrm{cold}$, $F_\mathrm{hot}$ and $F_\mathrm{cold}$ are respectively the temperatures and flow rates of the hot and cold streams entering the jacket. There is also a need to define a range for the manipulated variables ($F_\mathrm{hot}$ and $F_\mathrm{cold}$), as presented in Table 3.1. The values

**Table 3.1.:** *Range for the manipulated variables, $F_{hot}$ and $F_{cold}$.*

| | $F_{hot}$ / L min$^{-1}$ | $F_{cold}$ / L min$^{-1}$ |
|---|---|---|
| minimum | 0 | 0 |
| maximum | 10.0 | 10.0 |
| range = maximum - minimum | 10.0 | 10.0 |

**Table 3.2.:** *Parameters for the MMA polymerization reaction.*

| $V$ | $V_j$ | $A$ | $F_j$ | $U_{eff}$ |
|---|---|---|---|---|
| 250 mL | 170 mL | 174 cm$^2$ | 5.0 L min$^{-1}$ | 365.4 J K$^{-1}$min$^{-1}$ |

for the manipulated variables are set according to:

$$F_{hot} = \begin{cases} F_{hot,range} \times \dfrac{m_c - m_{c,split}}{100 - m_{c,split}}, & \text{if} \quad m_c - m_{c,split} > 0\,, \\ 0\,, \quad \text{if} \quad m_c - m_{c,split} \leqslant 0\,, \end{cases} \tag{3.7a}$$

$$F_{cold} = \begin{cases} F_{cold,range} \times \dfrac{m_{c,split} - m_c}{m_{c,split}}, & \text{if} \quad m_{c,split} - m_c > 0\,, \\ 0\,, \quad \text{if} \quad m_{c,split} - m_c \leqslant 0 \end{cases} \tag{3.7b}$$

where $F_{hot,range}$ and $F_{cold,range}$ are given in Table 3.1 and $m_{c,split}$ is the point where the split occurs. In this study $m_{c,split} = 50\,\%$.

## 3.2.2. Simulation results

### Closed-loop response to a temperature setpoint change

Figures 3.14 to 3.21 show the simulation results for the closed-loop system using the split-range control strategy, with a MMA solution free-radical polymerization reaction taking place in a laboratory batch reactor. Similarly to the procedure in Section 2.2.3, the mixture temperature setpoint is changed from 65 to 50°C when a 27% conversion is attained. As it can be observed in figures 3.20 and 3.18, this decrease in $T_r$ broadens the molecular weight distribution of the resulting polymer and increases its average molecular weights. However the reactor temperature will not go down instantaneously, but rather evolves according to heat transfer limitations, under the controller actuation (Figure 3.14). The controller was tuned based on the trial and error method described in section 2.2.4, obtaining the parameters presented in Table 3.3 and producing the controller command signal in Figure 3.22. Figure 3.13 illustrates the application of the mentioned method for the controller tuning.

Figure 3.21 represents the controller output signal in the interval $[10, 60]$min and it shows that despite of the fact that the system appeared to be in "stead state" at this period (as suggested by Figure 3.22), the cold flow rate is actually not null. The controller keeps the cold stream valve open just enough to coold down the reactor.

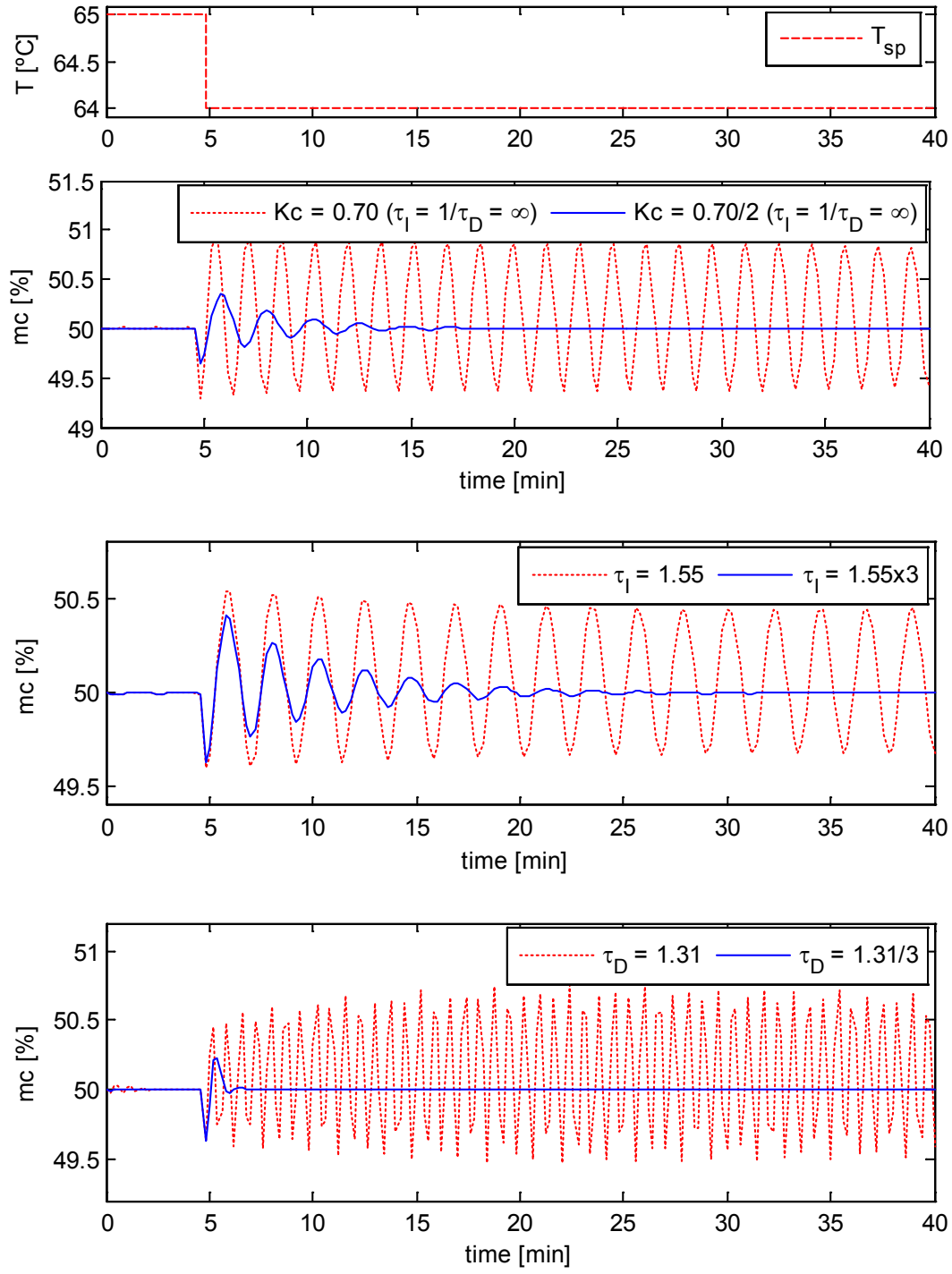


**Figure 3.13.: PID controller tuning.** *$K_c$ - Proportional gain, $\tau_I$ - Integral time, and $\tau_D$ - Derivative time.*

Also, one can observe a controller compensation of the temperature overshoot below the desired setpoint by opening the hot stream for a short time period. The relevance of the derivative kick phenomenon referred in Section 2.2.4 can be observed by comparing figures 3.22 and 3.23. Figure 3.23, which result from the digital con-

**Table 3.3.:** *PID controller parameters: controlled variable - $T_r$; manipulated variables - $F_{hot}$ and $F_{cold}$. The application of the tuning method described in Section 2.2.4 is illustrated in Figure 3.13.*

| $\Delta t$ | $K_c$ | $\tau_I$ | $\tau_D$ | $m_{c,max}$ | $m_{c,min}$ | $m_{c,split}$ |
|---|---|---|---|---|---|---|
| 0.2 min | 0.70/2 $^{\circ}C^{-1}$ | 1.55x3 min | 1.31/3 min | 100% | 0% | 50% |

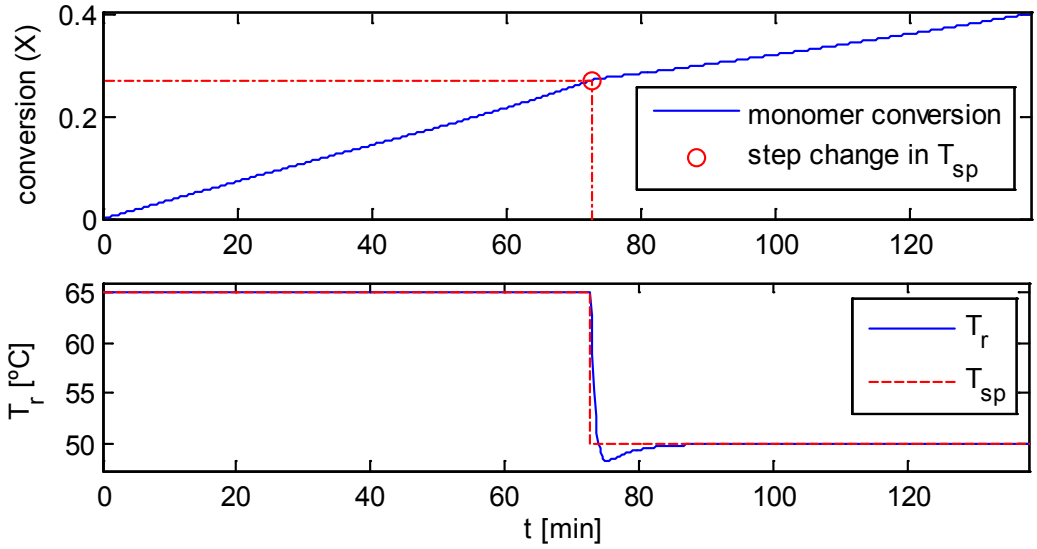


**Figure 3.14.:** *Monomer conversion and reactor temperature evolution.*

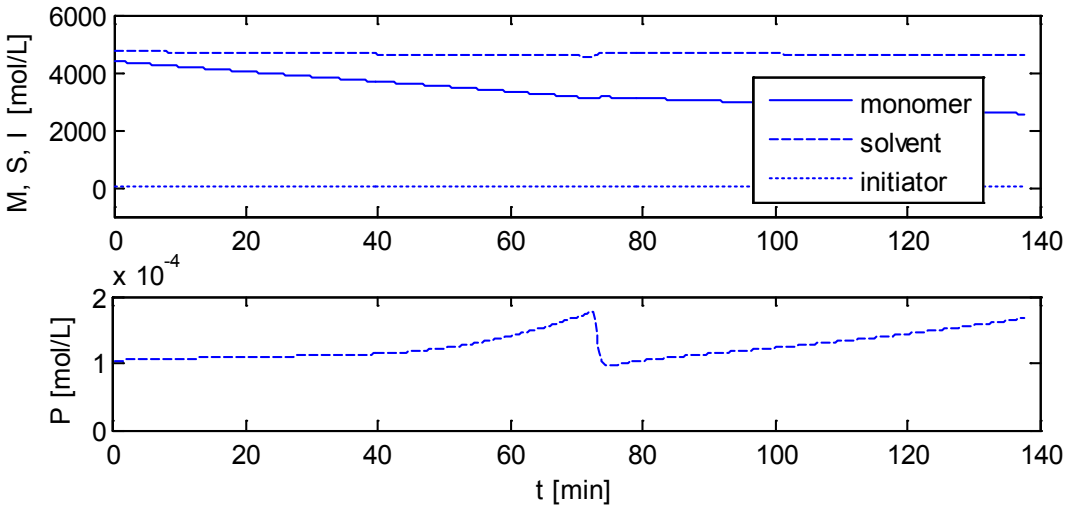


**Figure 3.15.:** *Monomer, solvent, initiator and live radical molar concentrations.*

trol equation that contemplates the setpoint in the derivative action, shows a much abrupt variation in the controller output signal compared to the profile in Figure 3.22 where the derivative kick is eliminated (see Equation 2.20).

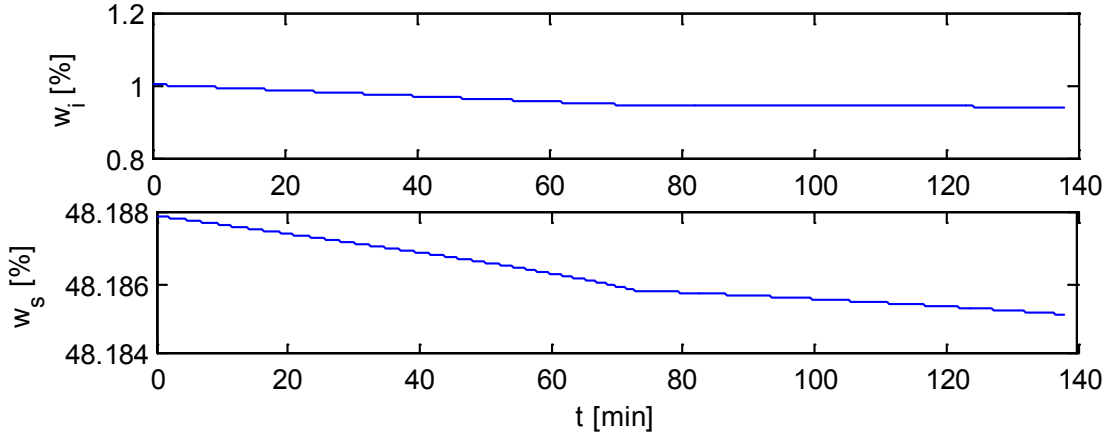


**Figure 3.16.:** *Mass fraction of the solvent and the initiator.*

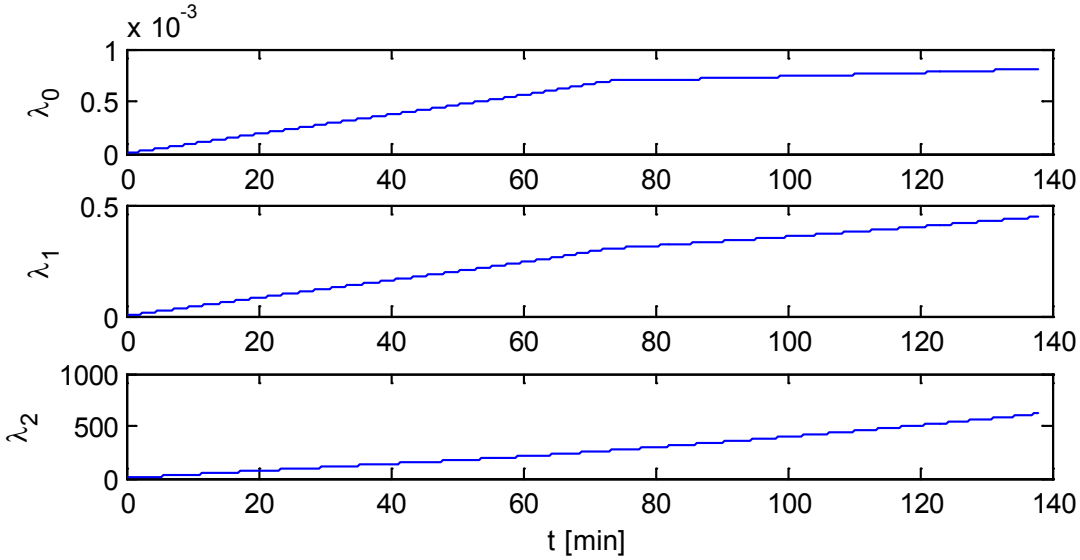


**Figure 3.17.:** *Molecular weight moments.*

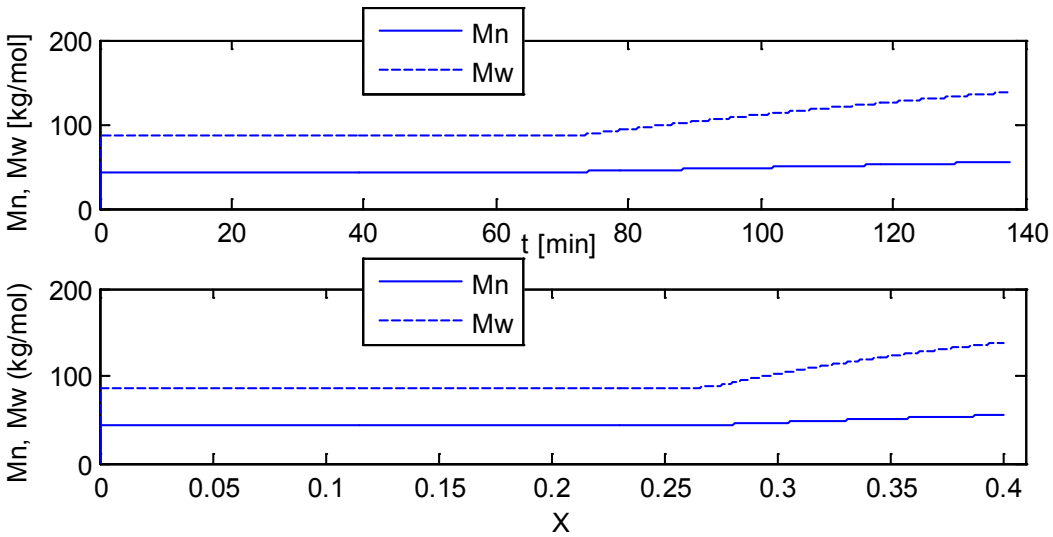


**Figure 3.18.:** *Number and weight average molecular weights.*

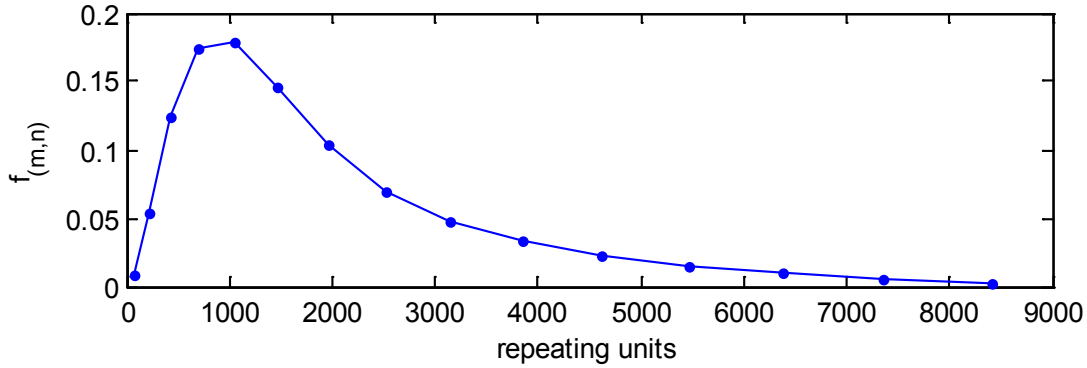


**Figure 3.19.:** *Molecular weight distribution at the final instant of the reaction.*

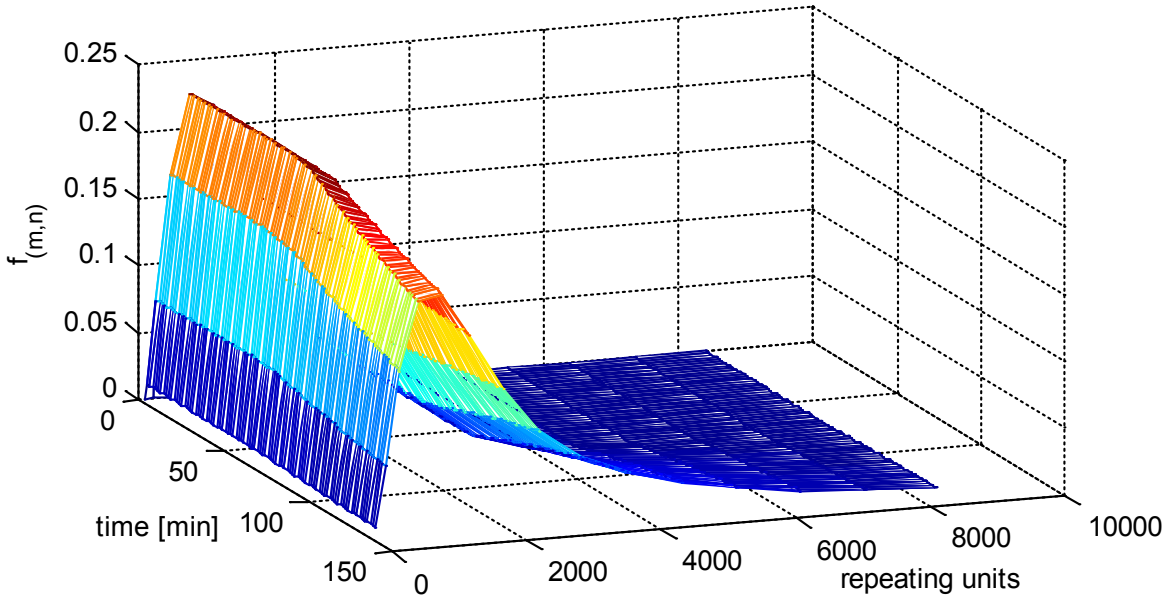


**Figure 3.20.:** *Evolution of the molecular weight distribution during the reaction time.*

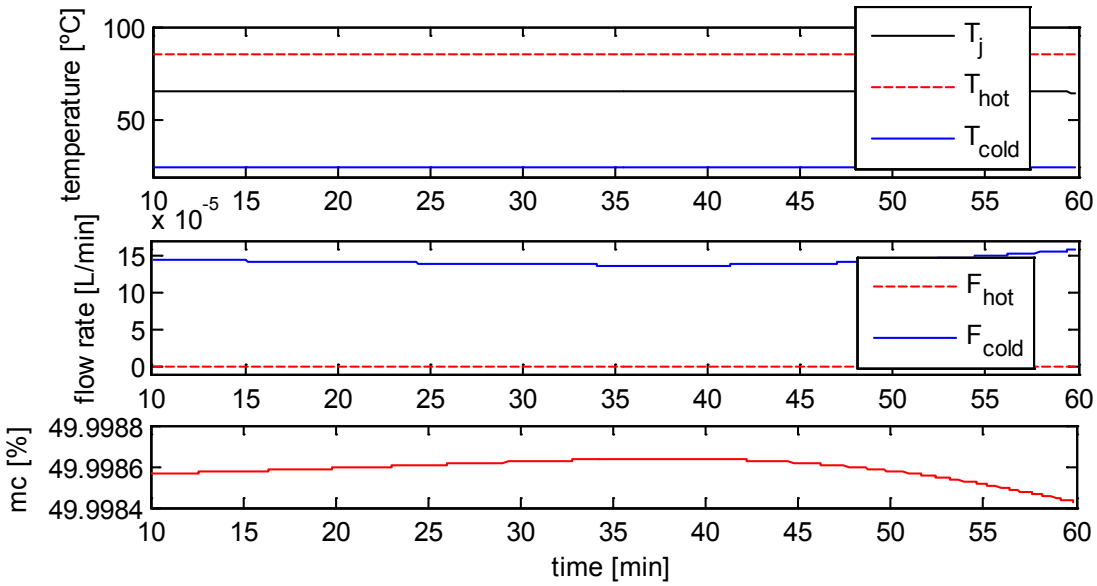


**Figure 3.21.:** *Jacket temperature ($T_j$), hot and cold fluid flow rate in the jacket ($T_{hot} = 85°C$ and $T_{cold} = 25°C$) and controller command signal (mc) in the interval $t = [10, 60]$min.*

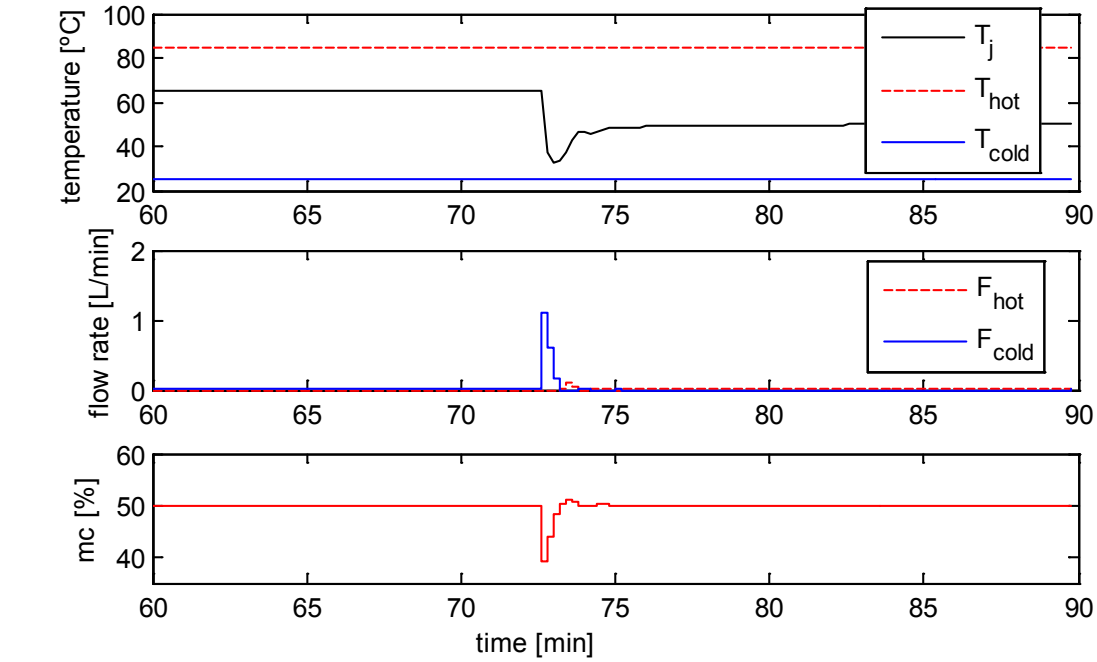


**Figure 3.22.:** *Jacket temperature ($T_j$), hot and cold fluid flow rate in the jacket ($T_{hot} = 85°C$ and $T_{cold} = 25°C$) and controller command signal (mc) profiles.*

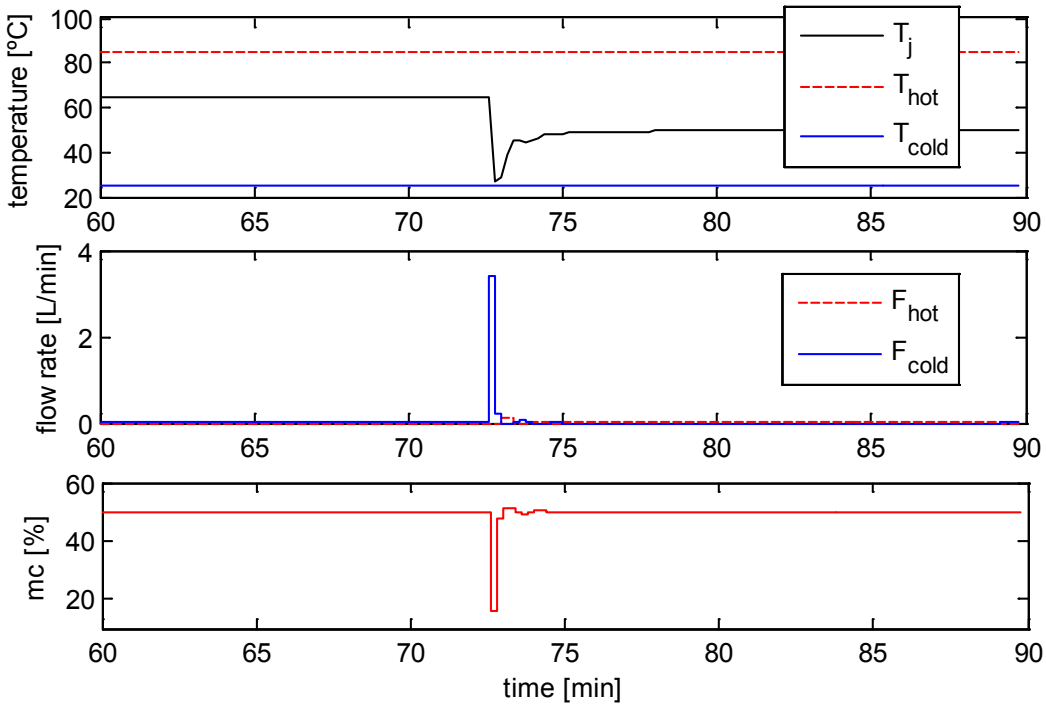


**Figure 3.23.:** *Jacket temperature ($T_j$), hot and cold fluid flow rate in the jacket ($T_{hot} = 85°C$ and $T_{cold} = 25°C$) and controller command signal (mc) profiles with derivative kick.*

# 4. Conclusions

A mathematical model based on the work of Crowley and Choi (1997b), for MMA solution polymerization, occurring in a batch laboratory reactor is presented. The model incorporates energy balance equations as well as a method for the MWD calculation. The simulations involve a thermal study of the lab-scale reactor in Section 3.1, validation of the model implementation in Section 2.2.3 and, finally, simulation of MMA polymerization occurring in the laboratory reactor (Section 3.2). The simulations performed in this study for the laboratory reactor will be useful when the reactor is installed, since thermal responsiveness data are available in this work for several scenarios, allowing good insight of the reactor thermal behavior. The three models considered for the jacket, namely, perfectly mixed, plug flow and lumped cooling jacket, are equivalent in terms of thermal dynamics, except for low jacket flow rates where the lumped cooling jacket model suggests a significant gradient in temperature along the jacket (see Section 3.1.3). With the MMA polymerization model is also presented, from which conversion and molecular weight distribution are determined. The model takes into consideration the gel effect discussed in Section 2.1 and the heat of reaction as well. The performed simulations may be used to control the reactor temperature according to a predetermined trajectory that leads to the desired polymer properties.

The thermal study results, from Section 3.1, denote that it takes around 18 min to heat the lab-scale reactor from 21 to 80°C or to cool it down from 80 to 21°C. Nevertheless, this is an approximate result as the heat losses to the exterior are neglected and the fluid inside the reactor is water that has different thermodynamic properties than the MMA reaction mixture. A good mixture can be achieved as the reactor has a small size (250 mL capacity), promoting a good heat transfer between the fluids in the reactor and in the jacket, and making the perfectly mixed model a suitable one for the system. However, the results cannot be directly transposed to a geometrically similar industrial scale reactor, as the relative importance of the transport phenomena may be different from one scale to another, as pointed out by Faísca (2002).

## 4.1. Future work

The computational framework developed in this work shows that one can track the setpoint for the reactor temperature and predict the MWD and monomer conversion profiles, for a specific time horizon. However, it would be interesting to develop nonlinear model predictive control based methodologies to determine the optimal

reactor temperature setpoints in order to target desired MWD and monomer conversion, and also to minimize the reaction time, as demonstrated by Silva and Oliveira (2002). Also, the modeling study must be extended to include the effect of the thermal capacitance of the total mass of glass of which the lab-scale reactor is made of. Regarding the upcoming perspective of performing polymerization experiments with this reactor, it is of paramount importance to assess the instrumentation and equipment needs to fit the reactor in order to control the reactor temperature and to ensure its safety.

# Bibliography


Adebekun, D. K. and Schork, F. J. (1989). Continuous solution polymerization reactor control. 1. nonlinear reference control of methyl methacrylate polymerization. *Ind. Eng. Chem. Res.*, 28(9):1308–1324.

Arora, P., Jain, R., Mathur, K., Sharma, A., and Gupta, A. (2010). Synthesis of polymethyl methacrylate (pmma) by batch emulsion polymerization. *African Journal of Pure and Applied Chemistry*, 4(8):152–157.

Cao, E. (2009). *Heat Transfer in Process Engineering*. McGraw-Hill.

Crowley, T. J. and Choi, K. Y. (1997a). Calculation of molecular weight distribution from molecular weight moments in free radical polymerization. *Ind. Eng. Chem. Res.*, 36(5):1419–1423.

Crowley, T. J. and Choi, K. Y. (1997b). Discrete optimal control of molecular weight distribution in a batch free radical polymerization process. *Ind. Eng. Chem. Res.*, 36(9):3676–3684.

Crowley, T. J. and Choi, K. Y. (1998). Experimental studies on optimal molecular weight distribution control in a batch-free radical polymerization process. *Chemical Engineering Science*, 53(15):2769–2790.

Curteanu, S., Bulacovschi, V., and Lisa, C. (1998). Free radical polymerization of methyl methacrylate: Modeling and simulation by moment generating function. *Iranian Polymer Journal*, 7(4):152–157.

Debab, A., Chergui, N., and Bertrand, J. (2011). An investigation of heat transfer in a mechanically agitated vessel. *Journal of applied fluid mechanics*, 4(2).

Denn, M. M. (1987). *Process modeling*. Harlow : Longman Scientific & Technical, Boston, MA.

DURATHERM (2011). Duratherm 450, heat transfer fluid. `http://www.heat-transfer-fluid.com/heat-transfer-fluid/duratherm-450`, accessed in May 2011.

Ellis, M. F., Taylor, T. W., Gonzalez, V., and Jensen, K. F. (1988). Estimation of the molecular weight distribution in batch polymerization. *AIChE*, 34(8):1341–1353.

Faísca, N. P. (2002). Polimerização em Suspensão do VCM. Faculdade de Ciências e Tecnologia, Universidade de Coimbra.

Fletcher, P. (1981). Heat transfer coefficients for stirred batch reactor design. *The Chemical Engineer*, pages 33–37.

Kell, G. S. (1975). Density, Thermal Expansivity, and Compressibility of Liquid Water from 0 °C to 150 °C: Correlations and Tables for Atmospheric Pressure and Saturation Reviewed and Expressed on 1968 Temperature Scale. *Journal of Chemical and Engineering Data*, 20(1):97–105.

Kranjk, M., Poljansek, I., and Golob, J. (2001). Kinetic modeling of methyl methacrylate free-radical polymerization initiated by tetraphenyl biphosphine. *Polymer*, 42(9):4153–4162.

Luyben, W. L. (1989). *Process Modeling, Simulation and Control for Chemical Engineers*. McGraw-Hill chemical engineering series, Schaum's outline series in civil engineering. McGraw-Hill, 2nd edition.

McCabe, W. L., Smith, J. C., and Harriott, P. (1993). *Unit Operations of Chemical Engineering*. Chemical and Petroleum Engineering Series. McGraw-Hill, New York, N.Y., 5th edition.

Poling, B. E., Thomson, G. H., Friend, D. G., Rowley, R. L., and Wilding, W. V. (2008). Physical and chemical data. In Perry, R. H. and Green, D. W., editors, *Perry's Chemical Engineers' Handbook*. McGraw-Hill, USA.

Rantow, F. S. and Soroush, M. (2005). Optimal control of a high-temperature semibatch solution polymerization reactor. In *American Control Conference 2005, Portland, OR, USA*.

Reid, R. C., Prausnitz, J. M., and Poling, B. E. (1987). *The Properties of Gases and Liquids*. McGraw-Hill, New York, 4th edition.

Schork, F. J., Deshpande, P. B., and Leffew, K. W. (1993). *Control of Polymerization Reactors*. Marcel Dekker, Inc., 1st edition.

Seborg, D. E., Edgar, T. F., and Mellichamp, D. A. (1989). *Process dynamics and control*. Wiley series in chemical engineering. Wiley, 1st edition.

Seborg, D. E., Edgar, T. F., and Mellichamp, D. A. (2004). *Process dynamics and control*. Wiley series in chemical engineering. Wiley, 2nd edition.

Silva, D. C. M. (2005). *Controlo Predictivo Não-linear de Processos Químicos - Aplicação a Sistemas de Polimerização Descontínuos*. PhD thesis, Faculdade de Ciências e Tecnologia, Universidade de Coimbra, Coimbra, Portugal.

Silva, D. C. M. and Oliveira, N. M. C. (2002). Optimization and nonlinear model predictive control of batch polymerization systems. *Computers and Chemical Engineering*, 26(4-5):649–658.

Yamada, B. and Zetterlund, P. B. (2002). General chemistry of radical polymerization. In Matyjaszewski, K. and Davis, T. P., editors, *Handbook of Radical Polymerization*. John Wiley & Sons, Inc., USA.

# Appendices

# A. Overall heat transfer coefficient

In section 3.1 a thermal study of the lab-scale reactor was done, where the overall heat transfer coefficient needed to be determined. This chapter illustrates how to estimate its value for an agitated jacketed vessel.

## A.1. Internal heat transfer coefficient

Equation (A.1) is an empirical correlation (Debab et al., 2011) for the calculation of $h_{\mathrm{i}}$, where $N_u$, $R_e$ and $P_r$ are the Nusselt, Reynolds and Prandtl numbers, respectively.

$$N_u = \theta_{10} R_e^{\theta_{20}} P_r^{\theta_{30}} V_i^{\theta_{40}} . \tag{A.1}$$

$$N_u = \frac{h_{\mathrm{i}} D_{\mathrm{i}}}{k} , \tag{A.2a}$$

$$R_e = \frac{\rho N d^2}{\mu} , \tag{A.2b}$$

$$P_r = \frac{\mu C_{\mathrm{p}}}{k} , \tag{A.2c}$$

$$V_i = \frac{\mu}{\mu_{\mathrm{w}}} , \tag{A.2d}$$

where $D_{\mathrm{i}}$ is the reactor internal diameter, $k$ is the thermal conductivity of the liquid inside the reactor, $\rho$ is the liquid density, $N$ is the impeller rotational speed, $d$ is the impeller diameter, $\mu$ is the fluid dynamic viscosity at the bulk mean temperature, $\mu_{\mathrm{w}}$ is the liquid dynamic viscosity at the wall temperature and $C_{\mathrm{p}}$ is the liquid specific heat. $h_{\mathrm{i}}$ is determined by substituting (A.2) in (A.1)

**Table A.1.:** *The typical values for the empirical correlation constants (Debab et al., 2011).*

| $\theta_{10}$ | $\theta_{20}$ | $\theta_{30}$ | $\theta_{40}$ |
|---|---|---|---|
| 0.54 | 2/3 | 1/3 | 0.14 |

## A.2. External heat transfer coefficient calculation

The heat transfer coefficient in the jacket side can be determined by equation (A.3) (Cao (2009))

$$\frac{h_o D_H}{k} = 1.86(R_e P_r \frac{D_H}{L})^{0.33} (\frac{\mu}{\mu_w})^{0.14}, \tag{A.3}$$

where $D_H$ is the hydraulic diameter given by

$$D_H = 4R_H, \tag{A.4}$$

with L the reactor height ($h_1$ from figure B.1) and $R_H$ the ratio between the jacket fluid flow area and the internal tube perimeter. In this case, the internal tube corresponds to the reaction vessel. The Reynolds number for this case is given by equation (A.5), while for $P_r$ calculation the equation (A.2c) still remains valid.

$$R_e = \frac{\rho u D_H}{\mu}. \tag{A.5}$$

## A.3. Overall heat transfer coefficient

The overall heat transfer coefficient based on the internal area ($U_i$) can be determined by equation (A.6) (McCabe et al., 1993).

$$\frac{1}{U_i} = \frac{1}{h_i} + \frac{x_w}{k_m}\frac{D_i}{\bar{D_L}} + \frac{1}{h_o}\frac{D_i}{D_o}, \tag{A.6}$$

where $x_w$ is the wall thickness, $k_m$ is the wall thermal conductivity and $\bar{D_L}$ is the logarithmic mean of the internal and external diameter, given by:

$$\bar{D_L} = \frac{D_i - D_o}{log(D_i/D_o)}. \tag{A.7}$$

# B. Heat transfer area

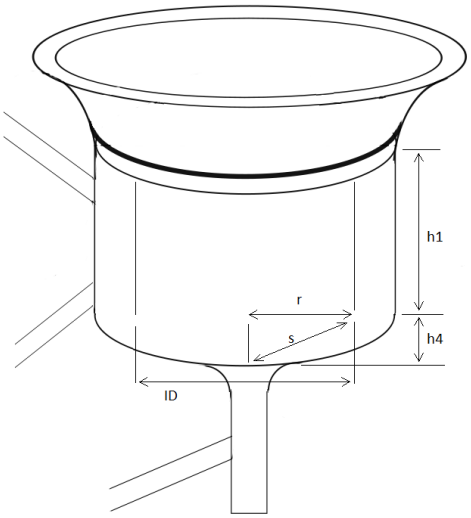


**Figure B.1.:** *Representation of the lab-scale polymerization reactor.*

The heat transfer area is given by:

$$A = A_{\mathrm{L}} + A_{\mathrm{B}}\,, \tag{B.1}$$

where $A_{\mathrm{L}}$ is the lateral surface area and $A_{\mathrm{B}}$ is the base surface area. $A_{\mathrm{L}}$ is given by

$$A_{\mathrm{L}} = \pi D_{\mathrm{i}} h_1\,, \tag{B.2}$$

while $A_{\mathrm{B}}$ is approximated by a cone lateral surface area.

$$A_{\mathrm{B}} = \pi\, r\, s \quad \text{where} \quad s = \sqrt{r^2 + {h_4}^2}\,. \tag{B.3}$$

The values of $D_{\mathrm{i}}$, $h_1$ and $h_4$, are given in Table B.1. Therefore, the heat transfer area is:

$$A = \left(\pi \times 75.0 \times 53.5 + \pi \times 37.5 \times \left(\sqrt{37.5^2 + 15.5^2}\right)\right) \mathrm{mm}^2 = 174\,\mathrm{cm}^2\,. \tag{B.4}$$

**Table B.1.:** *Lab-scale reactor dimensions.*

| $D_{\mathrm{i}}$ | $h_1$ | $h_4$ |
|---|---|---|
| 75.0 mm | 53.5 mm | 15.5 mm |

# C. Physical properties of fluids

The information provided here are about thermophysic properties needed for the simulations in sections 2.2.3, 2.2.4, 3.1 and 3.2. The molecular weight of the polymerization reagents are given in table C.1.

## C.1. Water

### C.1.1. Heat capacity

The water heat capacity in $\mathrm{J\,kg^{-1}K^{-1}}$ is given by equation C.1, with $[T] = {}^{\circ}\mathrm{C}$ (Reid et al., 1987).

$$C_{\mathrm{p}}(T) = 4185.5 \left( 0.996185 + 2.874 \times 10^{-4} \left( \frac{T+100}{100} \right)^{5.26} + 0.011160 \times 10^{-0.036T} \right) . \tag{C.1}$$

### C.1.2. Density

Equation (C.2) gives the water density in $\mathrm{kg\,m^{-3}}$ for a temperature range from 0 to 150°C at atmospheric pressure, according to Kell (1975).

$$\rho(T) = \Big( 999.83952 + 16.945176T - 7.9870401 \times 10^{-3}T^2 - 46.170461 \times 10^{-6}T^3 + 105.56302 \times 10^{-9}T^4 - 280.54253 \times 10^{-12}T^5 \Big) \Big/ \Big( 1 + 16.879850 \times 10^{-3}T \Big) . \tag{C.2}$$

**Table C.1.:** *Molecular weight of monomer (methyl methacrylate), solvent (Ethyl acetate) and initiator (2,2'-azobis(2-methylbutanenitrile)) (Ellis et al. (1988)).*

| **compound** | MMA | ethyl acetate | 2,2'-azobis(2-methylbutanenitrile) |
|---|---|---|---|
| $[\mathbf{M}] = \mathrm{g\,mol^{-1}}$ | 100.12 | 88.10 | 192.00 |

### C.1.3. Viscosity

The water dynamic viscosity, $[\mu] = \mathrm{Pa\,s}$, can be calculated from equation (C.3) (Reid et al. (1987)), with $[T] = {}^{\circ}\mathrm{C}$.

$$\mu(T) = 10^{-3} \times exp\left(-24.71 + 4209\frac{1}{T+273.15} + 0.04527(T+273.15) - 3.376\times10^{-5}(T+273.15)^2\right). \tag{C.3}$$

### C.1.4. Thermal conductivity

According to Poling et al. (2008), the water thermal conductivity, $[k] = \mathrm{W\,m^{-1}K^{-1}}$, is given by equation (C.4), with $[T] = {}^{\circ}\mathrm{C}$.

$$k(T) = -0.432 + 0.0057255\times(T+273.15) - 0.000008078\times(T+273.15)^2 + 1.861\times10^{-09}\times(T+273.15)^3. \tag{C.4}$$

## C.2. Ethyl acetate

### C.2.1. Heat capacity

From the *National Institute of Standards and Technology* (NIST) database, the solvent heat capacity was found to be

$$C_{\mathrm{p,s}} = 170.59\ \left(\mathrm{J\,mol^{-1}K^{-1}}\right). \tag{C.5}$$

### C.2.2. Density

$$\rho_{\mathrm{s}} = 925 - 1.237(T-273.15)\ \left(\mathrm{kg\,m^{-3}}\right), \tag{C.6}$$

with $[T] = {}^{\circ}\mathrm{C}$ (Ellis/etal:1988).

## C.3. Methyl methacrylate (MMA)

### C.3.1. Heat capacity

$$C_{\mathrm{p,m}} = 114.1 + 6.8299T\ \left(\mathrm{J\,kg^{-1}K^{-1}}\right), \tag{C.7}$$

with $[T] = {}^{\circ}\mathrm{C}$ (NIST).

### C.3.2. Density

$$\rho_{\mathrm{m}} = 965.4 - 1.09(T-273.15) - 9.7\times10^{-4}(T-273.15)^2\ \left(\mathrm{kg\,m^{-3}}\right), \tag{C.8}$$

with $[T] = {}^{\circ}\mathrm{C}$ (Ellis et al., 1988).

## C.4. Poly(methyl methacrylate) (PMMA)

### C.4.1. Heat capacity

$$C_{\mathrm{p,p}} = (0.265 + 1.39 \times 10^{-3} T) 4.184 \times 10^{3} \ \left(\mathrm{J\,kg^{-1}K^{-1}}\right), \tag{C.9}$$

with $[T] = {}^{\circ}\mathrm{C}$ (NIST).

### C.4.2. Density

$$\rho_{\mathrm{p}} = \frac{\rho_{\mathrm{m}}}{0.754 - 9 \times 10^{-4}(T - 273.15)} \ \left(\mathrm{kg\,m^{-3}}\right), \tag{C.10}$$

with $[T] = {}^{\circ}\mathrm{C}$ and $\rho_{\mathrm{m}}$ given by equation (C.8) (Ellis et al., 1988).

# D. Kinetic Parameters

Table D.1 contains the kinetic parameters for the MMA solution polymerization, with $k_d$ being the initiator decomposition rate constant, $k_p$ the propagation rate constant, $k_{fm}$ the chain transfer to monomer rate constant, $k_{fs}$ the chain transfer to solvent rate constant and $k_{td,0}$ disproportionation termination rate constant.

### Gel effect

The following correlation enables the inclusion of the gel effect into the disproportionation termination rate constant, as described in Crowley and Choi (1997b). For that purpose, the free and critical free volumes must be calculated from the following expressions:

$$v_f = 0.025 + 0.001(T - 167)\phi_m + 0.001(T - 181)\phi_s + 0.00048(T - 387)\phi_p\,, \quad \text{(D.1a)}$$

$$v_{fcr} = 0.186 - 2.96 \times 10^{-4}(T - 273.16)\,. \quad \text{(D.1b)}$$

$$g_{t1} = 0.10575 exp(17.15 v_f - 0.01715(T - 273.16))\,, \quad \text{(D.2a)}$$

$$g_{t2} = 2.3 \times 10^{-6} exp(75 v_f)\,. \quad \text{(D.2b)}$$

$$g_t = 0.5\left((g_{t1} - g_{t2})\, tanh\left(150\left(v_f - v_{fcr}\right)\right) + g_{t1} + g_{t2}\right)\,, \quad \text{(D.3)}$$

$$k_{td} = g_t \times k_{td,0}\,, \quad \text{(D.4)}$$

with $\phi_m$, $\phi_s$ and $\phi_p$ the monomer, solvent and polymer volume fractions, $v_f$ and $v_{fcr}$ the free volume and the critical free volume, respectively, $g_t$ the gel effect parameter and $[T] = \mathrm{K}$.

### Heat of MMA polymerization

According to Adebekun and Schork (1989), MMA polymerization reaction is exothermic, with a heat of reaction ($\Delta H$) of -13.8 kcal/mol.

**Table D.1.:** *Kinetic parameters for the MMA solution polymerization, with $[T] = K$ (Crowley and Choi (1997b)).*

| **parameter** | **value** | **unit** |
|---|---|---|
| $k_{\mathrm{d}}$ | $1.14 \times 10^{19} exp((-4.184 \times 34277)/(RT))$ | $\mathrm{min}^{-1}$ |
| $k_{\mathrm{p}}$ | $4.20 \times 10^{5} exp((-4.184 \times 6300)/(RT))$ | $\mathrm{m}^3\mathrm{mol}^{-1}\mathrm{min}^{-1}$ |
| $k_{\mathrm{fm}}$ | $1.75 \times 10^{10} exp((-4.184 \times 17957)/(RT))$ | $\mathrm{m}^3\mathrm{mol}^{-1}\mathrm{min}^{-1}$ |
| $k_{\mathrm{fs}}$ | $6.95 \times 10^{7} exp((-4.184 \times 15702)/(RT))$ | $\mathrm{m}^3\mathrm{mol}^{-1}\mathrm{min}^{-1}$ |
| $k_{\mathrm{td},0}$ | $1.06 \times 10^{8} exp((-4.184 \times 2800)/(RT))$ | $\mathrm{m}^3\mathrm{mol}^{-1}\mathrm{min}^{-1}$ |

# E. MATLAB simulation programs

## E.1. Thermal study

### E.1.1. model function

```
%-------------------------------------------------------------------------
% Model for the water-only system
% A. Perfectly mixed cooling jacket
%-------------------------------------------------------------------------

   function dxdt = tankmodel( t, x, modeldata )

   global fluidproperties;

% state variables

   Tr = x(1);
   Tj = x(2);

% system parameters

   U  = modeldata.U;
   A  = modeldata.A;
   V  = modeldata.V;
   Vj = modeldata.Vj;

% input variables

   Tj0 = modeldata.Tj0;
   Fj  = modeldata.Fj;


% physical properties

   rhor   = rho( Tr );
   Cpr    =  Cp( Tr );
   miur   = visc( Tr );
   kappar = tc( Tr );


   rhoj   = rho( Tj );
   Cpj    =  Cp( Tj );
   miuj   = visc( Tj );
   kappaj = tc( Tj );
```

```
  % pass this info to the main program

  fluidproperties.rhoj    = rhoj;
  fluidproperties.rhor    = rhor;
  fluidproperties.Cpj     = Cpj;
  fluidproperties.Cpr     = Cpr;
  fluidproperties.miuj    = miuj;
  fluidproperties.miur    = miur;
  fluidproperties.kappaj  = kappaj;
  fluidproperties.kappar  = kappar;


% energy balances

  heattransfer = U * A * (Tj - Tr);

  dTrdt = heattransfer / ( rhor * Cpr * V );

  dTjdt = ( Fj * rhoj * Cpj * (Tj0 - Tj) - heattransfer  ) / ( rhoj * Cpj * Vj );

  dxdt  = [ dTrdt; dTjdt ];

  end



%-------------------------------------------------------------------------
% physical properties of water
%-------------------------------------------------------------------------

  function C = Cp(T) %[J/(kg*K)] heat capacity
      C = 4185.5*(0.996185 + 2.874e-4*((T+100)/100).^5.26+0.011160*10.^(-0.036*T));
  end

  function D = rho(T) %[kg/m^3] density
      D = (999.83952+16.945176*T-7.9870401e-3*T.^2 ...
            -46.170461*10^-6*T.^3 + 105.56302*10^-9*T.^4 ...
         -280.54253*10^-12*T.^5)/(1+16.879850*10^-3*T);
  end

  function miu = visc(T) %[Pa*s] dynamic viscosity
      miu = 1e-3*exp(-24.71 + 4209*(1./(T+273.15)) + 0.04527*(T+273.15)...
          - 3.376*10^-5*(T+273.15).^2);
  end

  function k = tc(T) %[W/(m*K)] thermal conductivity
    k = -0.432 + 0.0057255*(T+273.15) - 0.000008078*(T+273.15).^2 + ...
        1.861E-09*(T+273.15).^3;
  end
```

## E.1.2. Main program

```
%-------------------------------------------------------------------------
% Program for the water-only system
% A. Perfectly mixed cooling jacket
%-------------------------------------------------------------------------

  clear all
  clc
  clf

% disable paging of the output in the MATLAB Command Window

  more off

%-------------------------------------------------------------------------

  global fluidproperties;

% initial states

  ns = 2;              % number of state variables

  Tr = 80.0;           % °C reactor temperature
  Tj = 22.0;           % °C jacket temperature

  xplant = [ Tr; Tj ]; % initial conditions

% inputs

  Tj0 = 21.0       ;   % °C
  Fj  = 1/60/1000.;    % m^3/s jacket flow rate

  modeldata.Tj0 = Tj0; % pass the information into the input variable "modeldata"
  modeldata.Fj  = Fj;  %

% parameters

  U  = 250;                  % W/(m^2*K) (Just a symbolic value. The correct value
                             % for U is calculated inside the for-cycle bellow
  Di = 75e-3; D0 = 105e-3;      % m reactor internal and external diameters
  L1 = 53.5e-3;                 % m reactor height
  A_lat = pi*Di*53.5*1e-3;      % m^2 lateral area
  A_bas = pi*(75/2)*40.6*1e-6;  % m^2 base area
  A  = A_lat + A_bas;           % m^2 heat transfer area
  V  = 250e-6;                  % m^3 volume of the liquid inside the reactor
  Vj = 170e-6;                  % m^3 jacket volume
  k_glass = 1.05;               % W/(m*K) glass thermal conductivity

  modeldata.U  = U;
  modeldata.A  = A;
  modeldata.V  = V;
  modeldata.Vj = Vj;
```

```
% ------------------------------------------------------------------------
% time simulation specifications
% ------------------------------------------------------------------------

   tinit  =    0.0;    % s
   tfinal =  1500.0;   % s
   dt     =   15.0;    % s   sampling time

% number of simulation iterations

   kmax = length( tinit : dt : tfinal );

% pre-allocate memory for results storage arrays

   resxmeas  = zeros(ns, kmax);
   resTj0    = zeros( 1, kmax);
   resFj     = zeros( 1, kmax);
   restime   = zeros( 1, kmax);
   resCpj    = zeros( 1, kmax);
   resCpr    = zeros( 1, kmax);
   resrhoj   = zeros( 1, kmax);
   resrhor   = zeros( 1, kmax);
   resmiuj   = zeros( 1, kmax);
   resmiur   = zeros( 1, kmax);
   reskappar = zeros( 1, kmax);
   reskappaj = zeros( 1, kmax);
   resU      = zeros( 1, kmax);


  %% ------------------------------------------------------------------------
% ode solver OPTIONS:
% force max stepsize for ODE solver and tell the ODE solver that only
% nonnegative solutions are acceptable
% set AbsTol and RelTol; default values are 1e-6 and 1e-3
% ------------------------------------------------------------------------

   tstep = 0.5*dt;

   ODEoptions = odeset( 'MaxStep',     tstep, ...
                        'NonNegative', 1:ns , ...
                        'AbsTol',      1e-6 , ...
                        'RelTol',      1e-4 );

% ========================================================================
% LOOP dynamic system simulation
% ========================================================================

   for k = 1:kmax

%     current time instant t0
```

```
        t0 = (k - 1)*dt;

%       ------------------------------------------------------------------
%       PLANT state estimates/measurements
%       here we assume that all the states are measured
%       ------------------------------------------------------------------
%       save measurements for NMPC problem initialization

        xmeas  = xplant;

%       ------------------------------------------------------------------
%       call the controller
%       ------------------------------------------------------------------

%       ( ... )

%       ------------------------------------------------------------------
%       set disturbances if any
%       ------------------------------------------------------------------

%       ( ... )

%       ------------------------------------------------------------------
%       record current process/plant data
%       ------------------------------------------------------------------

        resxmeas(:,k) = xmeas;
        resTj0(k)     = Tj0;
        resFj(k)      = Fj*1000*60.; % m3/s --> L/min
        restime(k)    = t0/60.;      %   s  -->  min

%       print to terminal

        fprintf( '␣%4i␣␣%8.2f␣␣%10.6f␣%10.6f␣%10.6f␣%10.6f␣\n', ...
                 k, t0/60., xmeas(1), xmeas(2), Tj0, Fj*1000*60. );

%       get fluids properties at t0

        dxdt = tankmodel( t0, xplant, modeldata );

        resrhoj(k)    = fluidproperties.rhoj;
        resrhor(k)    = fluidproperties.rhor;
        resCpj(k)     = fluidproperties.Cpj;
        resCpr(k)     = fluidproperties.Cpr;
        resmiuj(k)    = fluidproperties.miuj;
        resmiur(k)    = fluidproperties.miur;
        reskappar(k)  = fluidproperties.kappar;
        reskappaj(k)  = fluidproperties.kappaj;

        %-----------------------------------
        %hi calculation:
        d = 40e-3;          %[m] impeller diameter
```

```
        N = 2000*2*pi/60; %impeller rotational speed
        Re_r = resrhor(k)*N*d^2/resmiur(k);        %Reynolds number inside the reactor
        Pr_r = resmiur(k)*resCpr(k)/reskappar(k); %Prandtl number inside the reactor
        hi = (reskappar(k)/Di)* 0.54* Re_r^(2/3)* Pr_r^(1/3); %W.m^-2K^-1

        %h0 calculation
        xw = 2.7e-3; %[m] glass thickness
        A_f = (pi/4)*((D0-xw)^2 - (Di+xw)^2); Pi = pi*Di; % m^2 flow area
        R_H = A_f/Pi; D_eq = 4*R_H;                        % m hydraulic radius
        u = (Fj)/A_f; % m/s jacket fluid flow speed
        Re_j = resrhoj(k)*u*D_eq/resmiuj(k);       %Reynolds number
        Pr_j = resmiuj(k)*resCpj(k)/reskappaj(k); %Prandtl number
        h0 = (reskappar(k)/D_eq)* 1.86* (Re_j* Pr_j*D_eq/L1)^0.33; %W.m^-2K^-1

        DL_bar = (Di-D0)/log(Di/D0); % m diameter logarithmic mean
        U = (1/hi + xw/k_glass*(Di/DL_bar) + 1/h0*Di/D0)^(-1); % W.m^-2K^-1(see McCabe)

        modeldata.U  = U;
        resU(k) = U;
        %-----------------------------------------


        kplotmax = k;  % this may be useful for plotting data

%       ------------------------------------------------------------------------
%       PLANT:
%       ------------------------------------------------------------------------

        x0plant = xplant;

        tnew = t0 + dt;

        odemodel = @(t0, x0plant) ...
                    tankmodel(t0, x0plant, modeldata );

        [ soltime, solxnew ] = ode15s( odemodel, [t0 tnew], x0plant, ODEoptions );

%       get the new state variables

        xplant = solxnew(end,:)';

        %Tolerance for thermal establilization is 0.1 °C:
        tol_T = 0.1;
        if abs(xplant(1)-resxmeas(1,k)) < tol_T && ...
                abs(xplant(2)-resxmeas(2,k)) < tol_T
            break;
        end

    end

% ========================================================================
% end of LOOP
```

```
% ========================================================================


%% plot results

  figure(1)

  subplot(7,1,1:3)
  plot(restime(1:kplotmax),resxmeas(1,1:kplotmax)); hold on;
  plot(restime(1:kplotmax),resxmeas(2,1:kplotmax),'r--');
  %xlabel('t (min)')
  ylabel('T_r & T_j [°C]')
  legend('T_r','T_j')
  hold off

  subplot(7,1,4:5)
  plot(restime(1:kplotmax),resU(1:kplotmax));
  xlabel('t [min]')
  ylabel('U [W/(m^2.K)]')

  subplot(7,1,6)
  plot(restime(1:kplotmax),resTj0(1:kplotmax)); hold on;
  %xlabel('t [min]')
  ylabel('T_{j,0} [°C]')
  hold off

  subplot(7,1,7)
  stairs(restime(1:kplotmax),resFj(1:kplotmax)); hold on;
  xlabel('t [min]')
  ylabel('F_j [L/min]')
  hold off



  figure(2)

  subplot(2,1,1)
  plot(restime(1:kplotmax),resrhor(1:kplotmax)); hold on;
  plot(restime(1:kplotmax),resrhoj(1:kplotmax),'r--');
  %xlabel('t [min]')
  ylabel('rho_r & rho_j [kg/m3]')
  legend('rho_r','rho_j',4)
  hold off

  subplot(2,1,2)
  plot(restime(1:kplotmax),resCpr(1:kplotmax)); hold on;
  plot(restime(1:kplotmax),resCpj(1:kplotmax),'r--');
  xlabel('t [min]')
  ylabel('C_{p,r} & C_{p,j} [J/(kg.K)]')
  legend('C_{p,r}','C_{p,j}')
```

```
  hold off

% end of file
```

## E.2. MMA polymerization reaction study

### E.2.1. Model function

```
%-------------------------------------------------------------------------
% Model for the MMA polymerization system
% - Perfectly mixed cooling jacket
%-------------------------------------------------------------------------

function dxdt = tankmodel( t, x, modeldata)

  global fluidproperties;


% state variables

  X   =  x(1);      % monomer conversion
  wi  =  x(2);      % polymer mass fraction
  ws  =  x(3);      % initiator mass fraction

  mu0 =  x(4);      % 0th order inactive polymer chain distribution moment
  mu1 =  x(5);      % 1st order inactive polymer chain distribution moment
  mu2 =  x(6);      % 2nd order inactive polymer chain distribution moment

  Tr  =  x(7);      % reaction mixture temperature
  Tj  =  x(8);      % jacket temperature

  fmn =  x(9:end); % polymer mass fraction with chain length between m and n


% system parameters

  Mt   = modeldata.Mt;
  Vj   = modeldata.Vj;

  R    = modeldata.R;
  UA   = modeldata.UA;
  dH   = modeldata.dH;
  fi   = modeldata.fi;

  Mm   = modeldata.Mm;
  Ms   = modeldata.Ms;
  Mi   = modeldata.Mi;

  nint = modeldata.nint;
  m    = modeldata.m;
  n    = modeldata.n;
```

```
  wm0  = modeldata.wm0;

%  Fj   = modeldata.Fj;

  Fhot  = modeldata.Fhot;
  Thot  = modeldata.Thot;
  Fcold = modeldata.Fcold;
  Tcold = modeldata.Tcold;

% physical properties

  wm = wm0*(1 - X);

  wp = 1 - ws - wm;

  rhom = rho_m(Tr); rhop = rho_p(Tr); rhos = rho_s(Tr); rhoj = rho_j(Tr);
  rhor = wm*rhom + ws*rhos + wp*rhop;

  Cpm  = Cp_m(Tr); Cps = Cp_s(Tr); Cpp = Cp_p(Tr);
  Cpr  = wm*Cpm + ws*Cps + wp*Cpp;
  Cpj  = Cp_j(Tj);

% kinetic parameters for MMA solution polymerization (Crowley & Choi, 1997)

  TK = Tr + 273.15; % conversion from °C to K
  RTK = R * TK;
  %Crowley & Choi 1997b
  kd  = 1.14 * 10^19*exp( -4.184*34277/RTK);     % min^-1          initiator decomposi
  kp  = 4.20 * 10^5 *exp( -4.184*6300/RTK);      % m^3/(mol.min)   propagation
  kfm = 1.75 * 10^10*exp( -4.184*17957/RTK);     % m^3/(mol.min)   chain to monomer tra
  kfs = 6.95 * 10^7 *exp( -4.184*15702/RTK);     % m^3/(mol.min)   chain to solvent tra
  kt0 = 1.06 * 10^8 *exp( -4.184*2800/RTK);      % m^3/(mol.min)   termination rate co


% mixture volume

  V = ( wm/rhom + wp/rhop + ws/rhos) * Mt;       %m^3

% volume fractions of M, S, P, I

  phim = wm * Mt / (rhom*V);
  phip = wp * Mt / (rhop*V);
  phis = 1 - phim - phip;

% free volume and critical free volume

  Vf    = 0.025 + 0.001*(TK - 167)*phim + 0.001*(TK - 181)*phis + 0.00048*(TK - 387)*p
  Vfcri = 0.186 - 2.96 * 10^-4*(TK - 273.16);

% gel effect parameter

  gt1 = 0.10575 * exp( 17.15*Vf - 0.01715*(TK - 273.16) );
```

```
   gt2 = 2.3 * 10^-6 * exp( 75*Vf );

   gt  = 0.5*( (gt1 - gt2) * tanh( 150*(Vf - Vfcri) ) + gt1 + gt2 );

% termination reaction rate constant

   kt  = gt * kt0;   % m^3/(mol.min)

% concentrations  of monomer M, solvent S, initiator I, radicals Rad

   CM   = wm * Mt/(Mm*V);
   CS   = ws * Mt/(Ms*V);
   CI   = wi * Mt/(Mi*V);

   CRad = sqrt( 2*fi * kd * CI / kt );

% pass info to the main program

   fluidproperties.Cpr  = Cpr;
   fluidproperties.Cpj  = Cpj;
   fluidproperties.rhor = rhor;
   fluidproperties.rhoj = rhoj;
   fluidproperties.rhom = rhom;
   fluidproperties.rhos = rhos;
   fluidproperties.rhop = rhop;
   fluidproperties.CM   = CM;
   fluidproperties.CS   = CS;
   fluidproperties.CI   = CI;
   fluidproperties.CRad = CRad;


% mass balance

   alfa = kp*CM / ( kp*CM + kfm*CM + kfs*CS + kt*CRad );

   dXdt  =  ( kp + kfm ) * wm * CRad / wm0;

   dwidt = -kd * wi;

   dwsdt =  -kfs * ws * CRad;

   fluidproperties.alfa = alfa;


% number average and weight average molecular weights are calculated from the three
% leading moments of the weight chain length distribution for the dead polymer.

   dmu0dt =  kp * CM * V * (1 - alfa) * CRad;
   dmu1dt =  kp * CM * V * (2 - alfa) * CRad;
   dmu2dt = (kp * CM * V * ( alfa^2 - 3*alfa + 4) * CRad)/(1 - alfa);

% dynamics of the function that defines the weight fraction of the polymer within the
```

```
% chain length interval (m,n); they are integrated together with other
% kinetic equations at a discrete number of chain length intervals to calculate the
% entire weight chain length distribution (WCLD). One can choose any number of
% chain length intervals.

  for j = 1:nint

      factor = kp * CRad * CM * V / mu1;

      term1  =  (     m(j)   * (1 - alfa) + alfa ) * alfa^(m(j) - 1) / alfa ;
      term2  =  ( (n(j) + 1) * (1 - alfa) + alfa ) * alfa^n(j) / alfa ;

      term3  = fmn(j) * dmu1dt/mu1;

      dfmndt(j) = factor * (term1 - term2) - term3 ;

%     dfmndt(j)  = (((m(j)*(1-alfa)+alfa)/alfa)*alfa^(m(j)-1) - (((n(j)+1)*(1-alfa)+.
%         alfa)/alfa)*alfa^n(j) - (2-alfa)*fmn(j)).*(kp*CM*CRad./mu1);
  end

% energy balance

  heattransfer = UA * (Tr - Tj);

  dTrdt = ((-dH)*kp*wm*Mt*CRad - heattransfer) / ( rhor * Cpr * V );

  dTjdt = Fhot  * ( Thot  - Tj ) / Vj + Fcold * ( Tcold - Tj ) / Vj...
      + heattransfer  / ( rhoj * Cpj * Vj );


% derivatives

  dxdt = [ dXdt dwidt dwsdt dmu0dt dmu1dt dmu2dt dTrdt dTjdt dfmndt]';

end


%-------------------------------------------------------------------------
% physical properties monomer, polymer and solvent (T [oC])
%-------------------------------------------------------------------------

  function C = Cp_s(Tc) %[J/(kg*K)] (NIST)
    Ms = 88.10/1000;
    C = 170.59/Ms; %[J/(mol*K)] to %[J/(kg*K)]
  end
  function rhos = rho_s(Tc) %[kg/m^3]
      TK = Tc + 273.15;
      rhos = 925 - 1.237 * (TK-273.15);
  end

  function C = Cp_m(Tc) %[J/(kg*K)] (NIST)
      TK = Tc + 273.15;
```

```
    C = 114.1 + 6.8299*TK;
end
function rhom = rho_m(Tc) %[kg/m^3]
    TK = Tc + 273.15;
    rhom = 965.4 - 1.09 * (TK-273.15) - 9.7*10^-4 * (TK-273.15)^2;
end

function C = Cp_p(Tc) %[J/(kg*K)] [298 to 463 K] (NIST)
    TK = Tc + 273.15;
    C = (0.265 + 1.39*10^-3*TK)*4.184e3; %cal/(g*K) to J/(kg*K)
end
function rhop = rho_p(Tc) %[kg/m^3]
    TK = Tc + 273.15;
    rhop = rho_m(TK)/( 0.754 - 9*10^-4 * (TK-273.15) );
end



%-------------------------------------------------------------------------
% physical properties of water 450 (T [°C])
%-------------------------------------------------------------------------
function C = Cp_j(T) %[J/(kg*K)] (prof thesis)
    C = 4185.5*(0.996185 + 2.874e-4*((T+100)/100).^5.26+0.011160*10.^(-0.036*T));
end

function D = rho_j(T) %[kg/m^3] (prof thesis)
    D = (999.83952+16.945176*T-7.9870401e-3*T.^2 ...
          -46.170461*10^-6*T.^3 + 105.56302*10^-9*T.^4 ...
       -280.54253*10^-12*T.^5)/(1+16.879850*10^-3*T);
end
```

## E.2.2. Main program

```
%-------------------------------------------------------------------------
% MMA polymerization system simulation program
% - Perfectly mixed cooling jacket
%-------------------------------------------------------------------------

clear all
clc
clf

% disable paging of the output in the MATLAB Command Window

more off

% time/date

year   = datestr( now, 11 );
month  = datestr( now, 5 );
day    = datestr( now, 7 );
hour   = hour( datestr(now) );
minute = minute( datestr(now) );
```

```
% second = second( datestr(now) );


  fidroot      = 'datafile';
  fidextension = '.txt';

  fidcase1 = 'Xwpwimus';
  fidcase2 = 'fmn';

  fidT = 'CrCh1997b';

  filename1 = sprintf( '%s%s%s%s', fidroot, fidT, fidcase1, fidextension);
  filename2 = sprintf( '%s%s%s', fidroot, fidcase2, fidextension);

  fid = fopen( 'pid-parameters','wt' );
  fid1 = fopen( filename1,'wt' );
  fid2 = fopen( filename2,'wt' );

  fprintf( fid1, '%% %s %s %s %02d:%02d \n', year, month, day, hour, minute );
  fprintf( fid2, '%% %s %s %s %02d:%02d \n', year, month, day, hour, minute );

% ------------------------------------------------------------------------------

  global fluidproperties;


% molecular weights, kg/mol

  Mm = 100.12/1000;
  Ms =  88.10/1000;
  Mi = 192.00/1000;

% initial operating conditions - preliminary calculations

  Tr = 65;            % Initial temperature inside the reactor (°C)
  [ rhom, rhop, rhos, rhoi ] = ppyrhoMMA( Tr );

  phim = 0.50;
  phis = 0.50;
  phii = 1 - phim - phis;

  V  = 250e-6; %[m^3]
  Vj = 170e-6; %[m^3]

  massS = V * phis * rhos;
  massM = V * phim * rhom;
  CI0 = 0.046e3; %mol/m^3
  massI = CI0*Mi*V;

  Mt = massS + massM + massI;  % total mass, kg

  dH = -13.8e3*4.184;          % [J/mol]
```

```
% initial states

   X    = 0.0;        %  monomer conversion
   wp   = 0.0;        %  polymer mass fraction

   wi   = massI/Mt; %  initiator mass fraction
   ws   = massS/Mt; %  solvent mass fraction

   wm0 = massM/Mt; %  monomer mass fraction

   mu0 = 1.0e-9;       %  0th order inactive polymer chain distribution moment
   mu1 = 1.0e-9;       %  1st order inactive polymer chain distribution moment
   mu2 = 1.0e-9;       %  2nd order inactive polymer chain distribution moment

   Tj = 65;            % Initial temperature in the jacket(°C)

% number of intervals for MWD calculation
% the adjustable parameter a  must satisfy f( 2, 1 + nint * a * (nint + 1))) =
0.999

   nint = 15;

   a = 35;

   for i = 1:nint
       m(i) = 2 + a * (i - 1) * i;
       n(i) = 1 + a * (i + 1) * i;

       fprintf( ' m(%2i ) = %4i         n(%2i ) = %4i \n', i, m(i), i, n(i) );
   end

   fmn   =  zeros(1,nint);

   xplant = [ X wi ws mu0 mu1 mu2 Tr  Tj  fmn ]';


   ns = length( xplant ); % number of state variables


% ------------------------------------------------------------------------------
% inputs
% ------------------------------------------------------------------------------


   fprintf( fid1, '%% T = %3.1f C\n', Tr);
   fprintf( fid2, '%% T = %3.1f C\n', Tr);

   modeldata.nint   = nint;
   modeldata.m      = m;
   modeldata.n      = n;
```

```
modeldata.wm0    = wm0;


% ------------------------------------------------------------------------
% parameters
% ------------------------------------------------------------------------

R  = 8.31446; % J/(mol.K)        universal gas constant
A  = 174e-4;  % m^2              reactor heat transfer area
U  = 350;     % (J/s)/(m^2.K)    overall heat transfer constant
UA = U*A*60;  % (J/min)/(K)      effective U

fi = 0.21;  % initiator efficiency factor


modeldata.R  = R;
modeldata.UA = UA;
modeldata.dH = dH;

modeldata.Mt = Mt;
modeldata.Vj = Vj;
modeldata.fi = fi;

modeldata.Mm = Mm;
modeldata.Ms = Ms;
modeldata.Mi = Mi;



% inputs
Thot  = 85.0;  % °C       hot stream temperature  (manipulated variable)
Tcold = 25.0;  % °C       cold stream temperature (manipulated variable)
Fhot  = 0;     % m^3/min  hot stream flow rate to the jacket
Fcold = 0;     % m^3/min  cold stream flow rate to the jacket

mcsplit = 50;  % above this percentage we have hot stream

mc = 50;       % controller command signal from 0 to 100%

modeldata.Thot  = Thot;
modeldata.Tcold = Tcold;

modeldata.Fhot  = Fhot;
modeldata.Fcold = Fcold;

Fhotmax  = 10/1000;   % L/min --> m3/min
Fhotmin  = 0;
Fhotrange = Fhotmax - Fhotmin;

Fcoldmax = 10/1000;   % L/min --> m3/min
Fcoldmin = 0;
```

```
Fcoldrange = Fcoldmax - Fcoldmin;



%-------------------------------------------------------------------------
% time simulation specifications
%-------------------------------------------------------------------------

tinit  =   0.0;   % min
tfinal = 250.0;   % min
%   tfinal = 40.0;  % min (controller tuning)
dt     =   0.2;   % min   sampling time



%-------------------------------------------------------------------------
% controller parameters
% controlled variable:  Tr
% controller output:    mc
%-------------------------------------------------------------------------

Trsetpoint = 65.0;     %   °C

global pid
pid.dt    = dt;       %   min
pid.Kc    = 0.70/2;   %   /°C
pid.tauI  = 1.55*3;   %   min
pid.tauD  = 1.31/3;   %   min
pid.erro1 =  0.;
pid.erro2 =  0.;
pid.medida1 =  Tr;
pid.medida2 =  Tr;

pid.mc    = mc;       % %

mcmax     = 100;      % %
mcmin     =   0;      % %

pid.mcmax = mcmax;
pid.mcmin = mcmin;


%-------------------------------------------------------------------------

fprintf( fid, '%% dt       %8.5f \n', pid.dt );
fprintf( fid, '%% Kc       %8.5f \n', pid.Kc );
fprintf( fid, '%% tauI     %8.5f \n', pid.tauI );
fprintf( fid, '%% TauD     %8.5f \n', pid.tauD );

%-------------------------------------------------------------------------
```

```
% number of simulation iterations

  kmax = length( tinit : dt : tfinal );

% pre-allocate memory for results storage arrays

  resxmeas  = zeros(ns, kmax);
  restime   = zeros( 1, kmax);

  resCpr    = zeros( 1, kmax);
  resCpj    = zeros( 1, kmax);

  resrhor   = zeros( 1, kmax);
  resrhoj   = zeros( 1, kmax);

  resrhom   = zeros( 1, kmax);
  resrhop   = zeros( 1, kmax);
  resrhos   = zeros( 1, kmax);

  resCM     = zeros( 1, kmax);
  resCS     = zeros( 1, kmax);
  resCI     = zeros( 1, kmax);
  resCRad   = zeros( 1, kmax);

  resaveMn  = zeros( 1, kmax);
  resaveMw  = zeros( 1, kmax);

  resThot  = zeros( 1, kmax);
  resTcold = zeros( 1, kmax);
  resmc    = zeros( 1, kmax);
  resFhot  = zeros( 1, kmax);
  resFcold = zeros( 1, kmax);
  resTrsp  = zeros( 1, kmax);

  resalfa  = zeros( 1, kmax);

%-------------------------------------------------------------------------
% ode solver OPTIONS:
% force max stepsize for ODE solver and tell the ODE solver that only
% nonnegative solutions are acceptable
% set AbsTol and RelTol; default values are 1e-6 and 1e-3
%-------------------------------------------------------------------------

  tstep = 0.5*dt;

  ODEoptions = odeset( 'MaxStep',     tstep, ...
                       'NonNegative', 1:ns , ...
                       'AbsTol',      1e-6 , ...
                       'RelTol',      1e-4 );
```

```
% ================================================================================
% LOOP dynamic system simulation
% ================================================================================

   for k = 1:kmax

%      current time instant t0

       t0 = (k - 1)*dt;


%      ------------------------------------------------------------------------
%      PLANT state estimates/measurements
%      here we assume that all the states are measured
%      ------------------------------------------------------------------------

       xmeas  = xplant;


%      ------------------------------------------------------------------------
%      call the controller
%      ------------------------------------------------------------------------

       %setpoint change
       if xmeas(1) > 0.27
          Trsetpoint = 50.0; % ºC
       end

%        %controller tuning:
%        if k == 25
%           Trsetpoint = 64; % C
%        end


%      controller

       Tmedida = xmeas( 7 );

       mc = controllerPIDnoDkick( Trsetpoint, Tmedida );
%        mc = controllerPID( Trsetpoint, Tmedida );

%      0 < mc < 100 %


       Fhot  = Fhotrange  * max( 0, (mc - mcsplit) / (100 - mcsplit) );
       Fcold = Fcoldrange * max( 0, (mcsplit - mc) / (mcsplit)      );

       modeldata.Fhot  = Fhot;
       modeldata.Fcold = Fcold;
```

```
%     ------------------------------------------------------------------------
%     set disturbances if any
%     ------------------------------------------------------------------------

%     ( ... )

%     ------------------------------------------------------------------------
%     record current process/plant data
%     ------------------------------------------------------------------------
      resThot(k)  = Thot;
      resTcold(k) = Tcold;
      resmc(k)    = mc;
      resFhot(k)  = Fhot;
      resFcold(k) = Fcold;
      resTrsp(k)   = Trsetpoint;

      resxmeas(:,k) = xmeas;
      restime(k)    = t0;

%     get fluids properties at t0

      dxdt = tankmodel( t0, xplant, modeldata );

      resCpr(k)  = fluidproperties.Cpr;
      resCpj(k)  = fluidproperties.Cpj;

      resrhor(k) = fluidproperties.rhor;
      resrhoj(k) = fluidproperties.rhoj;
      resrhom(k) = fluidproperties.rhom;
      resrhop(k) = fluidproperties.rhop;
      resrhos(k) = fluidproperties.rhos;

      resCM(k)   = fluidproperties.CM;
      resCS(k)   = fluidproperties.CS;
      resCI(k)   = fluidproperties.CI;
      resCRad(k) = fluidproperties.CRad;

      resalfa(k) = fluidproperties.alfa;

%     acumulated average numerical and mass molecular weights

      resaveMn(k) = Mm * xmeas(5)/xmeas(4);
      resaveMw(k) = Mm * xmeas(6)/xmeas(5);

%     print to terminal

      fprintf(       ' %4i  %8.2f  %8.5f  %8.5f  %8.5f  %10.6f %10.6f %12.6f ', ...
              k, t0, xmeas(1), xmeas(2), xmeas(3), xmeas(4), xmeas(5), xmeas(6) );
      fprintf(       ' %10.6f  %10.6f  %10.6f  %10.3e ', ...
                     resCM(k)/1000, resCS(k)/1000, resCI(k)/1000, resCRad(k)/1000 );
      fprintf(       ' %10.6f  %10.6f  %10.6f  %10.6f %10.6f %10.6f \n', ...
                     resaveMn(k), resaveMw(k), resrhor(k), resrhom(k), resrhop(k), re
```

```
%     print to files

      fprintf( fid1, ' %4i  %8.2f  %8.5f  %8.5f  %8.5f  %10.6f %10.6f %12.6f ', ...
               k, t0, xmeas(1), xmeas(2), xmeas(3), xmeas(4), xmeas(5), xmeas(6) );
      fprintf( fid1, ' %10.6f  %10.6f  %10.6f  %10.3e ', ...
                      resCM(k)/1000, resCS(k)/1000, resCI(k)/1000, resCRad(k)/1000 );
      fprintf( fid1, ' %10.6f  %10.6f  %10.6f  %10.6f %10.6f %10.6f \n', ...
                      resaveMn(k), resaveMw(k), resrhor(k), resrhom(k), resrhop(k), resrh

      fprintf( fid2, ' %4i  %8.2f ', k, t0 );
      for i = 1:nint-1
          fprintf( fid2, ' %12.4f ', xmeas(6 + i) );
      end
      fprintf( fid2, ' %12.4f \n', xmeas(end) );


%     this may be useful for plotting data

      kplotmax = k;


%     ------------------------------------------------------------------------
%     PLANT:
%     ------------------------------------------------------------------------

      x0plant = xplant;

      tnew = t0 + dt;

      odemodel = @(t0, x0plant) ...
                    tankmodel(t0, x0plant, modeldata );

      [ soltime, solxnew ] = ode15s( odemodel, [t0 tnew], x0plant, ODEoptions );

%     get the new state variables

      xplant = solxnew(end,:)';


%    Stopping criterion
     if xmeas( 1 ) > 0.40
        fprintf( ' \n' );
        fprintf( ' \n' );
        fprintf( ' desired conversion reached  X = %10.6f \n', xmeas( 1 ) );
        fprintf( ' \n' );
        break
     end


  end
```

```
% ================================================================================
% end of LOOP
% ================================================================================



  fclose(fid1);
  fclose(fid2);


%% plot results

  figure(1)

  subplot(5,1,[1 3])
  plot(restime(1:kplotmax), resxmeas(1,1:kplotmax)'); hold on;
  for i = 2:kplotmax
      if resTrsp(i) == resTrsp(i-1)
      else
          time_change = restime(i); Trsp_change = resTrsp(i);
          break
      end
  end
  plot(time_change,0.27,'ro'); line([0 time_change],[0.27 0.27],'LineStyle','-.','colo
  line([time_change time_change],[0 0.27],'LineStyle','-.','color','r'); hold off
  xlabel('time [min]')
  ylabel('conversion (X)')
  legend('monomer conversion','step change in T_{sp}')
  axis([restime(1) restime(kplotmax) ...
      min(resxmeas(1,1:kplotmax)) max(resxmeas(1,1:kplotmax))])

  subplot(5,1,[4 5])
  plot(restime(1:kplotmax), resxmeas(7,1:kplotmax)'); hold on;
  stairs(restime(1:kplotmax), resTrsp(1:kplotmax)','r--'); hold off;
  xlabel('t [min]')
  ylabel('T_r [°C]')
  legend('T_r','T_{sp}')
  Tr = resxmeas(7,1:kplotmax);
  axis([restime(1) restime(kplotmax) min(Tr)-1 max(Tr)+1])


  figure(2)

  subplot(2,1,1)
  plot(restime(1:kplotmax), resxmeas(2,1:kplotmax)*100')
  xlabel('t [min]')
  ylabel('w_i [%]')

  subplot(2,1,2)
  plot(restime(1:kplotmax), resxmeas(3,1:kplotmax)*100')
  xlabel('t [min]')
  ylabel('w_s [%]')
```

```
figure(3)

subplot(3,1,1)
plot(restime(1:kplotmax), resxmeas(4,1:kplotmax)')
xlabel('t [min]')
ylabel('\lambda_0')

subplot(3,1,2)
plot(restime(1:kplotmax), resxmeas(5,1:kplotmax)')
xlabel('t [min]')
ylabel('\lambda_1')
subplot(3,1,3)
plot(restime(1:kplotmax), resxmeas(6,1:kplotmax)')
xlabel('t [min]')
ylabel('\lambda_2')


figure(4)

subplot(2,1,1)
plot(restime(1:kplotmax), resaveMn(1:kplotmax) ); hold on;
plot(restime(1:kplotmax), resaveMw(1:kplotmax), '--' )
xlabel('t [min]')
ylabel('Mn, Mw [kg/mol] ')
legend('Mn','Mw','Location','Best')
hold off;

subplot(2,1,2)
plot(resxmeas(1,1:kplotmax), resaveMn(1:kplotmax) ); hold on;
plot(resxmeas(1,1:kplotmax), resaveMw(1:kplotmax), '--' )
xlabel('X')
ylabel('Mn, Mw (kg/mol) ')
legend('Mn','Mw','Location','Best')
hold off;


figure(5)

subplot(5,1,[1 3])
plot(restime(1:kplotmax), resCM(1:kplotmax) ); hold on;
plot(restime(1:kplotmax), resCS(1:kplotmax), '--' );
plot(restime(1:kplotmax), resCI(1:kplotmax), ':' );
ylabel('M, S, I [mol/L] ')
legend('monomer', 'solvent', 'initiator', 'Location','Best')
hold off;

subplot(5,1,[4 5])
plot(restime(1:kplotmax), resCRad(1:kplotmax), '--' )
xlabel('t [min]')
ylabel('P [mol/L] ')
```

```
figure(6)

finalfmn = resxmeas(9:end, kplotmax);
finalfmncriterion = sum( finalfmn );
fprintf( '␣\n' );
fprintf( '␣\n' );
fprintf( '␣final␣fmn␣criterion␣>␣0.999␣?␣%10.6f␣\n', finalfmncriterion );
fprintf( '␣\n' );

plot([n], [finalfmn],'.-')
xlabel('repeating␣units')
ylabel('f_{(m,n)}')


figure(7)
[RESTIME,MN] = meshgrid(restime(1:8:kplotmax),[n]);
mesh(RESTIME,MN, [resxmeas(9:end,1:8:kplotmax)])
xlabel('time␣[min]')
ylabel('repeating␣units')
zlabel('f_{(m,n)}')


figure(8)

subplot(3,1,1)
plot(restime(1:kplotmax),resxmeas(8,1:kplotmax),'k'); hold on;
plot(restime(1:kplotmax),resThot(1:kplotmax),'r--');
plot(restime(1:kplotmax),resTcold(1:kplotmax),'b-');
xlabel('time␣[min]')
ylabel('temperature␣[°C]')
legend('T_j', 'T_{hot}', 'T_{cold}')
hold off

subplot(3,1,2)
stairs(restime(1:kplotmax),resFhot(1:kplotmax)*1000, 'r--'); hold on;
stairs(restime(1:kplotmax),resFcold(1:kplotmax)*1000,'b');
xlabel('time␣[min]')
ylabel('flow␣rate␣[L/min]')
legend('F_{hot}','F_{cold}')
hold off

subplot(3,1,3)
stairs(restime(1:kplotmax),resmc(1:kplotmax), 'r'); hold on;
xlabel('time␣[min]')
ylabel('mc␣[%]')
hold off


figure(9)
```

```
subplot(3,1,1)
plot(restime(50:300),resxmeas(8,50:300),'k'); hold on;
plot(restime(50:300),resThot(50:300),'r—');
plot(restime(50:300),resTcold(50:300),'b–');
xlabel('time␣[min]')
ylabel('temperature␣[ºC]')
legend('T_j', 'T_{hot}', 'T_{cold}')
hold off

subplot(3,1,2)
stairs(restime(50:300),resFhot(50:300)*1000, 'r—'); hold on;
stairs(restime(50:300),resFcold(50:300)*1000,'b');
xlabel('time␣[min]')
ylabel('flow␣rate␣[L/min]')
legend('F_{hot}','F_{cold}')
hold off

subplot(3,1,3)
stairs(restime(50:300),resmc(50:300), 'r'); hold on;
xlabel('time␣[min]')
ylabel('mc␣[%]')
hold off


figure(10)
plot(restime(1:kplotmax),resalfa(1:kplotmax))
xlabel('time␣[min]')
ylabel('\alpha')


figure(11)

subplot(5,1,[1 3])
plot(restime(300:450), resxmeas(1,300:450)'); hold on;
for i = 2:kplotmax
    if resTrsp(i) == resTrsp(i-1)
    else
        time_change = restime(i); Trsp_change = resTrsp(i);
        break
    end
end
plot(time_change,0.27,'ro'); line([0 time_change],[0.27 0.27],'LineStyle','-.','color',
line([time_change time_change],[0 0.27],'LineStyle','-.','color','r'); hold off
xlabel('time␣[min]')
ylabel('conversion␣(X)')
legend('monomer␣conversion','step␣change␣in␣T_{sp}')
axis([55 restime(450) ...
    min(resxmeas(1,300:450)) max(resxmeas(1,300:450))])

subplot(5,1,[4 5])
plot(restime(300:450), resxmeas(7,300:450)'); hold on;
stairs(restime(300:450), resTrsp(300:450)','r—'); hold off;
```

```
xlabel('t [min]')
ylabel('T_r [ºC]')
legend('T_r','T_{sp}')
Tr = resxmeas(7,300:450);
axis([55 restime(450) min(Tr)-1 max(Tr)+1])

figure(12)
subplot(3,1,1)
plot(restime(300:450),resxmeas(8,300:450),'k'); hold on;
plot(restime(300:450),resThot(300:450),'r--');
plot(restime(300:450),resTcold(300:450),'b-');
xlabel('time [min]')
ylabel('temperature [ºC]')
legend('T_j', 'T_{hot}', 'T_{cold}')
hold off

subplot(3,1,2)
stairs(restime(300:450),resFhot(300:450)*1000, 'r--'); hold on;
stairs(restime(300:450),resFcold(300:450)*1000,'b');
xlabel('time [min]')
ylabel('flow rate [L/min]')
legend('F_{hot}','F_{cold}')
hold off

subplot(3,1,3)
stairs(restime(300:450),resmc(300:450), 'r'); hold on;
xlabel('time [min]')
ylabel('mc [%]')
hold off

% end of file
```

## E.2.3. PID controller function

```
%-------------------------------------------------------------------------
% @ 2006  DEQ-FCTUC -- last modification: 2011-04-28
%-------------------------------------------------------------------------
% digital PID -- velocitity form
% E.g., see Seborg et al. (2004), page 201
%
% mc_k = mc_{k-1} + Kc * (e_k - e_{k-1}) + Kc * DeltaT * e_k / tau_I
%                 + (e_k - 2 * e_{k-1} + e_{k-2}) * Kc * tau_D / DeltaT
%
% Code elaborated in Octave.
%
% In the main program, the following parameters need to be defined:
%
% pid.dt   - sampling time
% pid.Kc   - proportional gain (P)
% pid.tauI - integral time     (I)
% pid.tauD - derivative time   (D)
%
% About activation/deactivation of I and D actions:
```

```
%
% - to cancel I action, select a very high value for tauI,
%   for instance, pid.tauI = 10000000.
%
% - to cancel D action, make pid.tauD = 0.
%
%
% Nota: to "switch of" the controller, just define Kc = 0 in the main program.
%
%-------------------------------------------------------------------------

  function mc = controllerPID( setpoint, medida )

  global pid

% error calculation

  erro     = setpoint - medida;

% contribution of each action

  actionP = erro - pid.erro1;
  actionI = pid.dt / pid.tauI * erro;
  actionD = -( medida - 2 * pid.medida1 + pid.medida2 ) * pid.tauD / pid.dt;

% controller signal calculation

  mc       = pid.mc + pid.Kc * ( actionP + actionI + actionD );

% check if mc is in the admissible range

  if mc < pid.mcmin
     mc = pid.mcmin;
  end

  if mc > pid.mcmax
     mc = pid.mcmax;
  end

% save the previous error values and present the just calculated controller signal:

  pid.erro2 = pid.erro1;
  pid.erro1 = erro;
  pid.medida2 = pid.medida1;
  pid.medida1 = medida;

  pid.mc     = mc;

% end of file
```

### E.2.4. Function for the calculation of monomer, polymer and solvent densities

```
%-------------------------------------------------------------------------
% physical properties of monomer, polymer & solvent
%-------------------------------------------------------------------------

function [ rhom, rhop, rhos, rhoi ] = ppyrhoMMA( Tc )      % kg/m^3 or g/L

TK = Tc + 273.15;

rhom = 965.4 - 1.09 * (TK-273.15) - 9.7*10^-4 * (TK-273.15)^2;

rhop = rhom/( 0.754 - 9*10^-4 * (TK-273.15) );

rhos = 925 - 1.237 * (TK-273.15);

rhoi = 0.94e3; %http://www.pharmaceutical-sales.net/products/p1/...
%22-Azodi(2-methylbutyronitrile)/

end
```