ANTONIO CESAR DE OLIVEIRA INTINI

MODELING OF POLYMERIZATION OF METHYL METHACRYLATE IN HOMOGENEOUS SYSTEMS AS A FRAMEWORK FOR PROCESSES IMPROVEMENTS

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para a obtenção do título de Mestre em Ciências.

Área de Concentração: Engenharia Química

Orientador: Prof. Titular Dr. Reinaldo Giudici

São Paulo

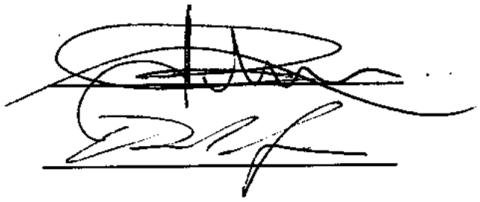
2019

Autorizo a reprodução e divulgação total ou parcial deste trabalho, por qualquer meio convencional ou eletrônico, para fins de estudo e pesquisa, desde que citada a fonte.

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 10 de Junho de 2019.

Assinatura do autor:



Assinatura do orientador:

Intini, Antonio Cesar de Oliveira

MODELING OF POLYMERIZATION OF METHYL METHACRYLATE IN HOMOGENEOUS SYSTEMS AS A FRAMEWORK FOR PROCESSES IMPROVEMENTS – Versão Corrigida – São Paulo, 2018.

133 p.

Dissertação (Mestrado) – Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Química.

1. Polimerização 2. Polímeros acrílicos 3.Cinética de reações químicas 4. Modelagem matemática. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Química II. T.

É uma tragédia o fato de que as maiores estrelas deste universo brilhem tanto e, ao mesmo tempo, tenham uma vida tão curta.

À memória de meu inesquecível amigo Lucas Stanquevisch (1988-2018).

ACKNOWLEDGMENTS

First, I would like to thank my supervisor, Professor Dr. Reinaldo Giudici, for all support and excellent and inspiring technical discussions and the opportunity of this MSc in polymer reaction engineering.

I also would like to give special thanks to Professors Dr. José Luiz Paiva, Dr. Marcelo Martins Seckler, and Dr. Roberto Guardani, for all contributions and remarks during the qualification of this project.

For the reviewed version of this dissertation, I would like to give my special thanks to the Professors Dr. Liliane Maria Ferrareso Lona and Dr. Wilson Hideki Hirota, who provided valuable suggestions and pointed out some improvement opportunities in the dissertation.

I leave my thanks to the Universidade of São Paulo, for providing the resources necessary for the MSc program, and all of its employees who, directly or indirectly, provided me all necessary support.

Another special thanks is addressed to Dr. Michael B. Cutlip, from PolyMath Software, who provided a free upgrade to the professional version of Polymath version 6.0.

Special thanks to Pery Freitas, who reviewed the final text and provided invaluable corrections and improvements in the text. I also would like to extend these thanks to my friends André Teixeira and Marcelo Cox, who reviewed the text submitted to the qualification exam.

I also would like to thank my friends and family, who supported me during this three-year journey in some way or another. I will not try to name them all here, as to not run the risk of forgetting any of them, but I feel I am a privileged person, as to count on them every time I needed.

ABSTRACT

The polymerization of methyl methacrylate (MMA) was investigated in this dissertation. Selected kinetic models from the literature were reviewed, and two new, generalized models of diffusion-limited effects (gel- and glass effects), derived from the current models were proposed and tested for bulk and solution polymerization of MMA in batch and semi-batch reactors, under isothermal and non-isothermal conditions. The newly proposed models include the capability of modeling termination by combination, radical transfer to monomer and depropagation reaction. The new and previous models were compared with experimental data of bulk and solution polymerizations of MMA, under a selection of non-steady state processes conditions (initiator and monomer feed, step changes in temperature) and compositions (initiator and chain transfer agents, regarding both type and dosages). The particular case of a non-isothermal bulk polymerization was also investigated. A simulation program (*Reactormodel*) was developed in Matlab, and its algorithm is provided.

Keywords:

Methyl methacrylate, azo-bisisobutironitrile, benzoyl peroxide, glycol dimercapto acetate, dodecyl mercaptan, acrylic polymerization, batch reactor, semi-batch reactor, isothermal polymerization, non-isothermal polymerization, modeling, coupled ordinary differential equations, method of moments, optimization, Matlab, Polymath.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AIBN	Azobisisobutironitrile
AK	Achilias and Kiparissides
BPO	Benzoyl peroxide
CB	Curteanu and Bulacovschi model
CCS	Chiu, Carratt and Soong model
CTA	Chain transfer agent
DDM	Dodecylmercaptan
GDMA	Glycol Dimercapto Acetate
IA	Intermediate addition (feed)
MMA	Methyl methacrylate
ODE	Ordinary differential equations
PDI	Polydispersity index
p-MMA	Poly-methyl methacrylate
RMS	Root mean square
RSG	Ray, Saraf and Gupta model
SBGSG	Sangwai et al model
SD	Step decrease
SG	Seth and Gupta model
SI	Step increase

LIST OF SYMBOLS

i_0: Initial absolute number of moles of species i (mol)

A: Diffusion parameter to CCS model

A_{ht}: Area available for heat transfer (m^2)

B: Diffusion parameter to CCS model

C_p: Heat capacity of the reaction mixture ($J.g^{-1}.°C^{-1}$)

D_{mp}: Mutual diffusion coefficient for monomer and polymer ($cm^2.s^{-1}$)

D_n: Number of moles of (dead) polymer with chain length n (mol)

$D_{p,xy}$: Mutual diffusion coefficient for polymers with a degree of polymerization x and y ($cm^2.s^{-1}$)

D_{i0}: Intrinsic mutual diffusion coefficient for species i ($cm^2.s^{-1}$)

E_{di}: Energy required by the species i to overcome intramolecular forces upon diffusion ($J.mol^{-1}$).

f: Initiator efficiency

f_0: Intrinsic initiator efficiency

f_{CTA}: Chain transfer agent efficiency

F_i: Molar feed rate of species i ($mol.s^{-1}$)

i_{fr}: Cumulative number of moles of species i fed to the reactor (mol)

i_{FT}: Number of moles of species i in the feed tanks (mol)

H: Heaviside function

I: Number of moles of the initiator (mol)

k_d: Rate constant for initiator decomposition (s^{-1})

k_{de}: Rate constant for depropagation reaction ($L.s^{-1}$)

k_{fc}: Rate constant for chain transfer to CTA ($L.mol.s^{-1}$)

k_{fm}: Rate constant for chain transfer to monomer (L.mol^{-1}.s^{-1})

k_{fp}: Rate constant for chain transfer to polymer (L.mol^{-1}.s^{-1})

k_{fs}: Rate constant for chain transfer to solvent (L.mol^{-1}.s^{-1})

k_p: Rate constant for propagation (L.mol^{-1}.s^{-1})

k_{p0}: Intrinsic rate constant for propagation (L.mol^{-1}.s^{-1})

k_p^{eff}: Effective rate constant for propagation (L.mol^{-1}.s^{-1})

k_t: Rate constant for termination (L.mol^{-1}.s^{-1})

k_{t0}: Intrinsic rate constant for termination (L.mol^{-1}.s^{-1})

k_{tc}: Rate constant for termination by combination (L.mol^{-1}.s^{-1})

k_{td}: Rate constant for termination by disproportionation (L.mol^{-1}.s^{-1})

N_A: Avogrado's number

m_i: Mass of species i (g)

M: Number of moles of monomer (mol)

$[M]$: Monomer concentration (mol.L^{-1})

$[M]_{eq}$: Monomer equilibrium concentration (mol.L^{-1})

Mn: Number-average molecular weight (g.mol^{-1})

M_{fr}: Number of mols fed to the reactor (mol)

Mw: Weight-average molecular weight (g.mol^{-1})

Mw_{crit}: Critical weight-average molecular weight (g.mol^{-1})

P_n: Instantaneous degree of polymerization

R: Ideal gas constant (J. °C $^{-1}$.mol^{-1})

R_i^*: Number of moles of free radical of species i (mol)

R_n^*: Number of moles of free radicals with length n (mol)

r_n: Number-average chain length

r_w: Weight-average chain length

S: Number of moles of solvent (mol)

t: Time (s)

T: Temperature (°C)

Tg: Glass transition temperature (°C)

Ts: Surrounding temperature

$\hat{V}_i^*$: Specific critical hole free volume of the monomer (i=m) and polymer (i=p)

V_f: Free-volume fraction

$V_{f_{crit}}$: Critical free-volume fraction

V_{fi}: Free volume of the species i

U: Overall heat transfer coefficient ($J.m^{-2}.\ °C^{-1}$)

V_r: Volume of the reaction mixture (L)

X: Conversion (mol fraction of monomer converted in the reactor)

X_m: Conversion (mass fraction of polymer in the reactor)

ΔH_p: Enthalpy of propagation reaction ($J.mol^{-1}$)

γ: Overlap factor

ζ_p: Radius of interaction for propagation (Å)

ζ_t: Radius of interaction for termination (Å)

$\theta_{f:}$: Initiator diffusion efficiency parameter to SG model

θ_p: Propagation diffusion parameter to CCS and derived models

θ_t: Termination diffusion parameter to CCS and derived models

θ_f: Initiator diffusion parameter to SG and derived models

λ_n: n^{th} moment of the polymeric radicals (mol)

$[\lambda_n]$: n^{th} moment of the polymeric radicals in concentration ($mol.L^{-1}$)

ξ: Ratio of the critical molar volume of the monomer jumping unit to the critical molar volume of the polymer

ρ_i: Density of species i ($g.cm^{-3}$)

φ_i: Volume fraction of species i

ωi: Model hybridization parameters

SUMMARY

CHAPTER I

1. INTRODUCTION

This chapter provides an overview of motivations, objective, and general considerations for this dissertation. The structure of the essay is also summarized.

1.1. Motivation

Vinyl polymers, of which acrylic polymers are highlighted in this study, are broadly used in the coatings and adhesive industry in large scale due to their excellent cost-benefit. Acrylic polymers can be used in applications such as decorative (acrylic dispersions, including copolymers with vinyl acetate and styrene) and industrial and automotive coatings (e.g., top coatings and clear coatings with high mechanical and chemical resistance).

These polymers are usually synthesized in solution (solventborne) or dispersions (waterborne dispersions). There is a variety of functional monomers, which makes it easier to include sites for crosslinking reactions and design the best polymer architecture for the intended use.

The industry devotes significant investments to the research and development (R&D) of such polymers. It is a common practice in the industry to perform the development of new products through trial-and-error or by statistical analysis by using instruments such as the design of experiments (DOE). On the other hand, there is a lack of a software application based on first principles, which could provide a way to hypothesize and calculate new compositions for acrylic polymers.

One of the reasons for this lack of technical solutions in polymer science is the complexity of the kinetic, diffusion, and thermodynamic problems present in these systems. Even though the computational power has evolved to an extent in which large systems of coupled ordinary differential equations (ODEs) can be solved in a few seconds in a modern desktop computer, the actual modeling of the phenomena is still the primary challenge in this area. Particularly in the polymerization of acrylic polymers, the main challenge is the modeling of

diffusional effects, which will affect the degree of polymerization throughout the polymerization time.

The understanding of these phenomena would allow the development of a laboratory software, which would help formulators to predict the outcome of new compositions, study thermal behavior (including the prevention of runaway reactions) and take advantage of diffusional and kinetic effects to optimize the process parameters. The knowledge gathered in one area, such as bulk polymerization, can be used in other polymerization conditions (e.g., solution and emulsion polymerization).

1.2. Objective

The objective of this dissertation is the development of a modeling framework for the polymerization of acrylic monomers, which is then used to compare existing and new proposed polymerization models. The monomer chosen for this study is methyl methacrylate (homopolymerization).

This modeling framework consists of an algorithm developed in Matlab, which is flexible enough for testing different reaction sets and diffusional models. With this framework, some models available in the literature are tested and compared.

Moreover, two new models with improved adjustments of the data are also provided and compared with the existing models, and their limitations are discussed. These optimized model take into consideration all of the essential reactions involved. One particular case is the radical transfer constant to monomers, which could potentially lead to the determination of particle-exiting radicals in emulsion polymerization.

The properties calculated in this study are conversion (X_m), number average molecular weight (Mn), and weight average molecular weight (Mw).

1.3. Structure of the Dissertation

This dissertation is organized as follows:

Chapter 2 provides the literature review of the polymerization field, focusing on methyl methacrylate, developing the historical timeline and disclosing the evolution, achievements, and limitations found in this area.

Chapter 3 discloses the mathematical modeling of the polymerization of acrylic monomers, which are the basis for the algorithm (the algorithm itself is provided in the appendix section).

Chapter 4 provides a brief explanation of the program and the experimental data used in this study. It also provides an overview of the code developed in this dissertation, named *Reactormodel*.

Chapter 5 presents the outcome of the simulations carried out with the different models, comparing their fitness to the available data. The model is then tested with additional experimental data, and the results are discussed.

Chapter 6 provides conclusions and recommendations for future work.

CHAPTER II

2. LITERATURE REVIEW

This chapter provides a review of the knowledge field related to the polymerization of acrylic monomers, with a particular focus on methyl methacrylate (MMA). Relevant polymerization models developed in this area are presented and discussed.

2.1. Introduction

Vinyl polymers, synthesized by addition polymerization, are one of the most versatile classes of polymers in the industry. Fittig and Engelhorn first synthesized first reports of styrene polymerization date back to 1839, and polymethacrylic acid in early 1880 (Flory, 1953). Acrylates, methacrylates and their copolymers (e.g., with styrene, vinyl acetate) are a particular subset of vinyl polymers, broadly used in the adhesives and coatings industry. Advances in this area date back to early 1901, with Otto Rohm's Ph.D. thesis on acrylates and methacrylates, which subsequently served as a basis for the flourishing of the industrial application through Röhm & Haas enterprises in Germany and the United States (Fazenda, 2009).

Acrylate and methacrylate monomers, such as methyl methacrylate (Figure 1) can be polymerized with free radicals generated by initiators, such as peroxides and azo-compounds in oil phase polymerization (mass, solution and suspension polymerization) and persulfates in the water phase (emulsion polymerization). The efficiency of the initiator is dependent on the medium and its chemical nature.

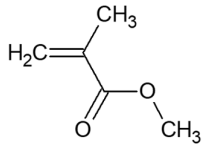


Figure 1: Methyl methacrylate (MMA)

Another source of radicals is the thermal self-initiation of the monomers (Walling; Briggs, 1946). This mechanism may be relevant depending on the monomers selected. The polymerization mechanism is favored by the absence

of oxygen in the atmosphere of the system (Flory, 1953), and it is standard practice the use of polymerization inhibitors as to ensure the proper storage of the monomer (Foord, 1940). The presence of inhibitors implies in an induction time in the polymerization process, which will consume some of the first radicals in the system. As the content of inhibitors is minimal, this mechanism may or may not be considered. In research works, it is also common practice to purify monomers before use to avoid such interference. In industrial practice, however, the monomer is used as received, thus with inhibitor, without purification.

Other possible routes for initiation include electron transfer-redox initiating systems, chemical initiation by metals or metal-containing compounds, redox initiators for suspension polymerizations, high-energy radiation initiation, and photoinitiation (Mishra; Yagci, 1998).

One way of controlling molecular weight is the use of chain transfer agents (CTAs), molecules with labile hydrogen of which the most used category are the thiols (O'Brien; Gornick, 1955), generally referred as mercaptans. Other components of the system, such as the solvents and the polymer, may also be able to transfer radicals. When these side reactions are relevant, applicable equations must be included in the system.

2.2. Kinetic models

Acrylic polymerization is very complex in nature. Usually its mechanism involves a large number of concurring parallel reactions. These reactions consist of initiator decomposition, polymerization initiation, propagation, depropagation, and termination, as well as radical chain transfer to species in the reaction medium (monomer, solvents, and polymer). Models can be developed by determining a set of coupled ordinary differential equations (ODEs) to calculate the rate of each of the reactions in the system.

Kinetic constants may vary during the polymerization, as a result of the diffusion limitations of the system: Figure 2 presents a typical behavior of the changes in termination rate constant as a function of monomer conversion in bulk polymerization of MMA.

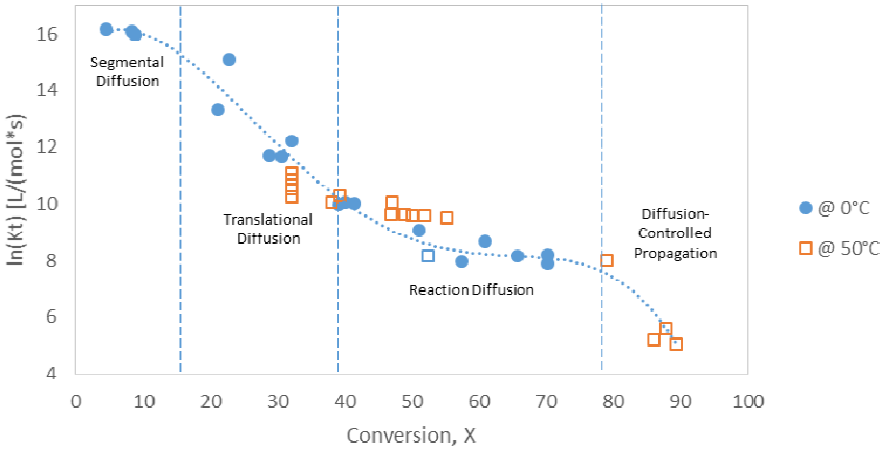


Figure 2 - Variation of k_t as a function of conversion for methyl methacrylate; data and classification of diffusion phases as proposed by Achilias (2007).

In the early stages of polymerization, the reaction is limited by the segmental diffusion of polymer chains (reaction sites must approach each other). With the advance of polymer fraction in the reactor, translational diffusion of the center of mass of polymer chains is slowed due to viscous effects. This behavior continues up to the point where the polymer chains will mostly move its radical end by the addition of a monomer unit to the chain, via propagation (reaction-diffusion). This diffusion proceeds until the conversion reach a critical point, where even the diffusion via propagation is halted. This condition is the glassy state of the mixture, which is only observable at temperatures below the Tg (Achilias, 2007).

Because of these behaviors, the polymerization will diverge from the ideal model of constant kinetics (at constant temperature), and the kinetic constants must be adjusted to account for these deviations.

The long chain hypothesis, which establishes that the kinetic reaction parameters do not depend on the degree of polymerization of the radical chain, is the standard practice for models of MMA polymerization. This assumption allows the use of a single kinetic parameter for each of the reactions involving radicals. Models accounting for each radical size (chain length dependent propagation and termination) were also developed, but these present particular challenges, such as the need to compute each equation (Achilias, 2007).

2.3. Diffusion Effects in Addition Polymerization

The effects of non-ideality of addition polymerization, such as the ones related to diffusion controlled reactions observed in the polymerization of methyl methacrylate, are the gel effect, glass effect, and cage effect. These effects and the relevant literature are reviewed in the following sections.

2.3.1. Gel Effect

The studies from Schulz and Harborth (1947), and Trommsdoff, Köhle, and Lagally (1948) were the first ones to demonstrate a kinetic effect, which causes the increase of the rate of polymerization throughout the reaction, known as gel effect (also known as autoacceleration and Trommsdorff effect). Schulz and Harborth (1947) demonstrated the effect of the polymer fraction in the reaction medium by polymerizing MMA at 50°C with benzoyl peroxide, varying the concentration of benzene (which acts as a solvent in this system). The experiment shows a sudden sharp increase in the rate of polymerization of MMA, at a defined conversion level, when it is polymerized in bulk (no benzene). The effect is minimized when experiments are carried with increasing amounts of solvent. At concentrations of benzene higher the 60%, autoacceleration is virtually negligible.

This behavior is explained by the decrease in the rate of termination in the system under the presence of less or no solvent. This decrease is the consequence of the lower mobility of polymer chains due to the increased viscosity of the system.

The termination kinetic constant k_t is limited by segmental diffusion at the early stages of the polymerization, when there are relatively no restrictions for the movement of polymer chains. As the polymerization progresses, termination reaction becomes limited by the translational diffusion of polymer chains. The net result is the decrease of termination reaction (due to decrease k_t). With decreased termination in the system, the radical concentration increases, and propagation is favored.

The gel effect is observable by the increased rate of conversion (X), determined by the derivative dX/dt, where $d^2X/dt^2 > 0$ (Figure 3).

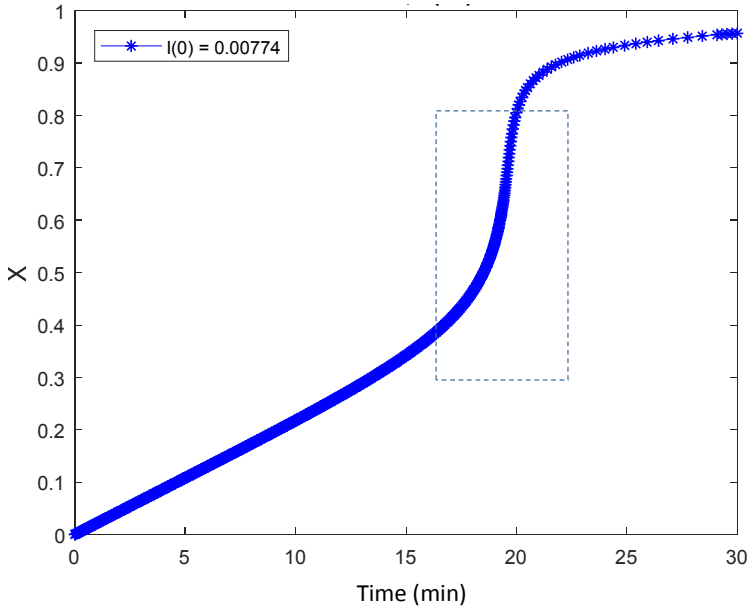


Figure 3: Gel effect in MMA polymerization at 90°C with AIBN (initial concentration = 0.00774 mol/L)

2.3.2. Glass Effect

Propagation is a reaction not subjected to the same restrictions as in termination reaction because one of its components (monomer) is a small molecule capable of diffusing easily in the reaction mixture. Because of this behavior, k_p remains relatively constant throughout the reaction.

However, in polymerization reaction carried out below the glass transition temperature (Tg) of the polymer, and at high polymer fractions, another effect will take place and decrease k_p: the glass effect. The system as a whole becomes glassy, thus reducing the mobility of all species, including the small molecules. This effect is the cause of conversions lower than 100% if T<Tg and polymer volume fraction (φ_p) in the reactor mixture is close to 1. The glass effect is observable by the decreased rate of conversion, determined by the derivative dX/dt, where $d^2X/dt^2 < 0$ (Figure 4), when the monomer conversion does not reach values close to 100% at longer polymerization times.

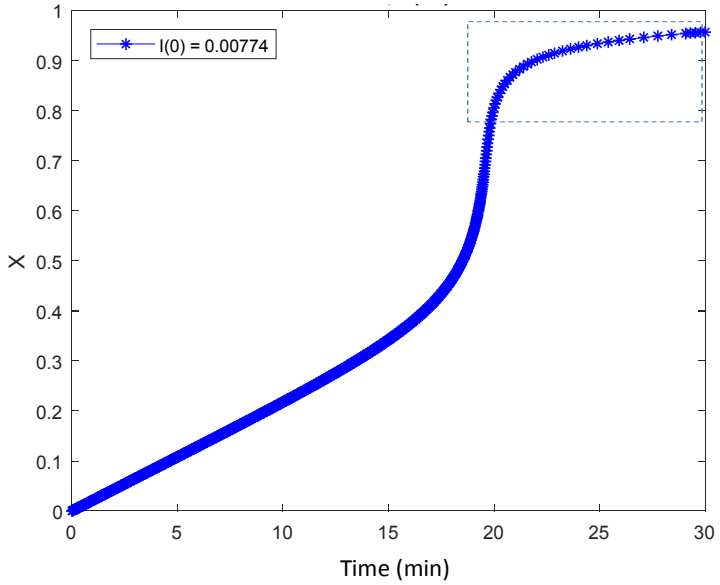


Figure 4: Glass effect in MMA polymerization at 90°C with AIBN (initial concentration = 0.00774 mol/L)

Under this condition, the glassy medium will affect the mobility of even small molecules such as the monomer, slowing down the propagation reaction (and all other reactions taking place).

In models based on free volume theory, it is standard practice to adopt two different equations for the temperature ranges below and above Tg to calculate the available free volume in the mixture so that the model can reproduce the behavior in both conditions (Soh; Sundberg, 1982; Achilias; Kiparissides, 1988).

2.3.3. Cage Effect

Initiators decompose in the reaction mixture upon heating, thus generating radicals, which are required to initiate the polymerization of vinylic monomers. However, it can be observed that the efficiency of the initiators is lower than 100%. When the radicals are generated, they have to diffuse into the reaction mixture, in order to react with a monomer molecule and initiate the polymerization. At the same time, some of these radical fragments may recombine on unwanted reactions, forming inert molecules and decreasing the overall efficiency of the system.

One example is the initiator azobisisobutyronitrile (AIBN), which releases nitrogen at decomposition (Figure 5). The product radical, cyanpropyl, can

generate two different products (tetramethylsuccinenitrile and ketamine) which are inert in the medium (Pryor, 1967). This undesirable reaction decreases the efficiency of the initiator, as less radical fragments are available to react with monomer.

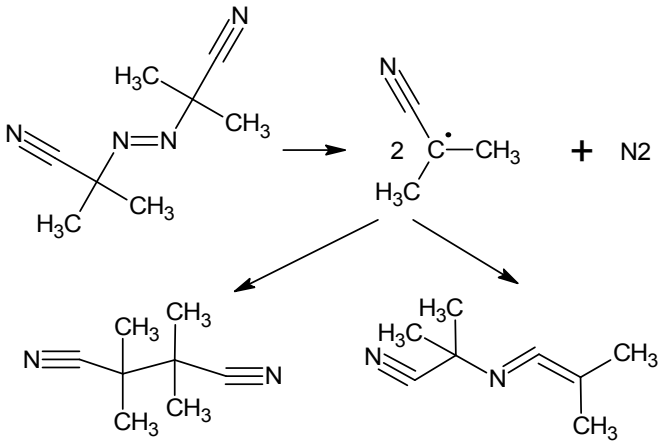


Figure 5. AIBN undesirable decomposition products

In order to consider this effect, an efficiency factor *f* is commonly used in polymerization models. In the case of AIBN, this efficiency factor in bulk polymerization is reported in the literature as *f*=0.58 (Chiu; Carratt; Soong, 1983), meaning that nearly half of the initiator is lost. Recent models also provide ways to calculate variable f over the course of the reaction (Seth; Gupta, 1995).

The viscosity of the medium affects the efficiency of the initiator (Matyjaszewski, 2002). This suggests the phenomenon be more relevant in bulk than in solution polymerization, and in less critical in emulsion polymerization, where the radical will form and propagate in the low-viscous water phase.

2.4. Diffusion models for k_p, k_t and f.

Several authors investigated the diffusion problems present in the polymerization of addition polymers such as polymethylmethacrylate (p-MMA). Different approaches were tried, with increasing advances, which brought a better understanding of the phenomena and allowing the use of mathematical models for simulation. These models have been categorized in the literature according to their structure.

2.4.1. Fickian Model

Achilias (2007) provided an extensive review of diffusion-controlled polymerization models. In this article, equations based on Smoluchowski diffusion-controlled reaction between two solid spherical particles, contextualized in the polymerization of addition polymers, are provided for k_p and k_t:

$$\frac{1}{k_p} = \frac{1}{k_{p0}} + \frac{1}{4\pi N_A \zeta_p D_{mp}} \quad (1)$$

$$\frac{1}{k_t} = \frac{1}{k_{t0}} + \frac{1}{4\pi N_A \zeta_t D_{p,xy}} \quad (2)$$

In these equations, k_{p0} and k_{t0} are the intrinsic rate constant for propagation and termination reactions. N_A is the Avogrado's constant. ζ_p and ζ_t are the radii of interaction respectively for propagation and termination. D_{mp} and $D_{p,xy}$ are the mutual diffusion coefficient, respectively, between monomer and polymer, and between polymers chains with a degree of polymerization x and y.

According to equations (1) and (2), the rate kinetic pseudo-constants k_p and k_t are a function of the intrinsic rates and a diffusion factor. When diffusion tends to infinite, k_t tends to k_{t0}. The lower the diffusion, the lower k_t and k_p. Kinetic models such as (1) and (2) were wrongly classified as "parallel", due to their resemblance to the parallel circuit equation in electric systems (Vivaldo-Lima; Hamielec; Wood, 1994; Gao; Penlidis, 1996); actually, these equations state the resistance to the process as a sum of two resistances in series, the resistance to the reaction in series with the resistance to diffusion, thus should be called "series", not "parallel".

There are several approaches to obtain the mutual diffusion coefficients for (1) and (2). Thus, equations are the basis for several models in the literature. In this dissertation, some important cases are reviewed. The models are named here after the authors by acronyms with their initials.

2.4.2. CCS Model

One of the first attempts to adjust experimental data with a Fickian model such as equations (1) and (2) is the work of Chiu, Carratt, & Soong (1983). This

group proposed a new model for k_p and k_t, based on diffusion coefficient's dependence on concentration and temperature (Fujita; Kishimoto; Kinya, 1960):

$$\frac{1}{k_p} = \frac{1}{k_{p0}} + \frac{\theta_p \lambda_0}{\exp(\frac{2.3\varphi_m}{A + B\varphi_m})} \quad (3)$$

$$\frac{1}{k_t} = \frac{1}{k_{t0}} + \frac{\theta_t \lambda_0}{\exp(\frac{2.3\varphi_m}{A + B\varphi_m})} \quad (4)$$

The model parameters were determined by fitting the experimental data of Marten and Hamielec (1979) for bulk polymerization of MMA with AIBN at 50°C, 70°C and 90°C, with two initial initiator loadings (0.01548 mol/L and 0.0258 mol/L). The values of the parameters θ_p(T), θ_t(T, I_0), A(T) and B(T) are provided in a table in the referred article. φ_m is the volume fraction of monomer. This model provides a good fit with the experimental data (Figure 6) but has the disadvantage of being based on adjusted parameters specific for each condition.

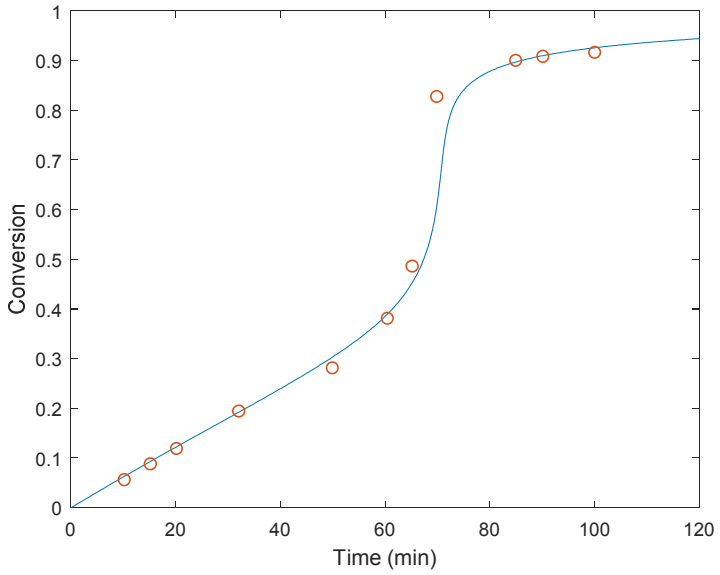


Figure 6: Conversion predicted by CCS model (I0=0.01548, T=70°C).

Another clear disadvantage of this model is the dependence of θ_t on initial initiator concentration. This dependence makes the model unfeasible for processes with intermediate or continuous addition of initiator to the reaction mixture, such as in semi-batch polymerization.

2.4.3. AK Model

One particular attempt to develop a model with meaningful physical parameters, which could be measured or calculated independently, are the works of Achilias and Kiparissides (1988; 1992), derived from the CCS model:

$$\frac{1}{k_t} = \frac{1}{k_{t0}} + \frac{(\zeta_t^2/3)\lambda_0 r_n^2}{V_r D_{p0}} \exp\left(\frac{E_{dp}}{RT}\right) \exp\left(\frac{\gamma(\varphi_m \hat{V}_m^* \rho_m + \xi \varphi_p \hat{V}_p^* \rho_p)}{\xi(\varphi_m \hat{V}_m^* V_{fm} \rho_m + \varphi_p \hat{V}_p^* V_{fp} \rho_p)}\right) \quad (5)$$

$$\frac{1}{k_p} = \frac{1}{k_{p0}} + \frac{(\zeta_p^2/3)\lambda_0}{V_r D_{m0}} \exp\left(\frac{E_{dm}}{RT}\right) \exp\left(\frac{\gamma(\varphi_m \hat{V}_m^* \rho_m + \xi \varphi_p \hat{V}_p^* \rho_p)}{(\varphi_m \hat{V}_m^* V_{fm} \rho_m + \varphi_p \hat{V}_p^* V_{fp} \rho_p)}\right) \quad (6)$$

In this model, the radii of interaction ζ_p and ζ_t (as in equations 1 and 2) are calculated from data available in the literature, and parameters of the process. λ_0 is the first moment of radicals. r_n is the number–average chain length at time t.

D_{p0} and D_{m0} are the intrinsic diffusion preexponential coefficients for polymer chains and monomer, and E_{dp} and E_{dm} are the activation energies for polymer and monomer, respectively, required for molecules to overcome attraction forces upon diffusion (Vrentas; Duda, 1977).

The second exponential applied to calculate diffusion can be interpreted as the ratio between the minimum of critical local hole free volume per gram of mixture and the average hole free volume per gram of mixture (Vrentas; Duda, 1977). The overlap factor γ is used to account for the fact that any free volume is available to more than one molecule in the reaction, and is reported to vary between ½ and 1 (Vrentas; Duda, 1977). ξ is the ratio of the critical molar volume of the solvent jumping unit to the critical molar volume of polymer jumping unit.

Hence the exponential components of equations (5) and (6) will account for the intramolecular forces counteracting the movement of the molecules, and the free volume available in the medium so that these molecules can diffuse.

This model was tested by the authors and good agreement with the experimental data could be found. The limitation of this model is the need to determine several parameters, which may not be readily available, but it sets a good approach by providing equations with physical meaning.

2.4.4. RSG Model

One step further on the use of the free volume approach in the modeling of MMA polymerization is the work of Ray, Saraf, and Gupta (1995). They developed a semiempirical model based on the CCS and AK models, which can be used to predict bulk and solution polymerization:

$$\frac{1}{k_t} = \frac{1}{k_{t0}} + \frac{\theta_t r_n^2 \lambda_0}{V_r \exp(-\psi + \psi_{ref})} \tag{7}$$

$$\frac{1}{k_p} = \frac{1}{k_{p0}} + \frac{\theta_p \lambda_0}{V_r \exp(\xi(-\psi + \psi_{ref}))} \tag{8}$$

The parameters θ_t and θ_p are obtained with the minimization technique of the normalized sum of square errors. These parameters are a function of temperature only, and one individual optimized parameter (IOP) is determined to each experiment. The parameters ψ and ψ_{ref} inside the exponentials are just the free-volume components of AK models, algebraically reorganized, and including a term for the solvent:

$$\psi = \frac{\gamma \left\{ \frac{\varphi_m \hat{V}_m^* \rho_m}{\xi_{13}} + \frac{\varphi_s \hat{V}_s^* \rho_s}{\xi_{23}} + \varphi_p \hat{V}_p^* \rho_p \right\}}{\varphi_m \hat{V}_m^* V_{fm} \rho_m + \varphi_s \hat{V}_s^* V_{fs} \rho_s + \varphi_p \hat{V}_p^* V_{fp} \rho_p} \tag{9}$$

$$\psi_{ref} = \frac{\gamma}{V_{fp}} \tag{10}$$

The subscripts in ξ refer to the species involved in the respective ratio (monomer=1, solvent=2, polymer=3). The authors also provided two polynomial equations, called best-fit correlations (BFC), which provide a continuous way of calculating the parameters. The error between both IOP and BFC are compared in the article.

θ_t and θ_p are calculated with polynomial functions provided by the authors:

$$\log_{10} \theta_t = C_1 \times 10^2 - C_2 \times 10^5 \left(\frac{1}{T}\right) + C_3 \times 10^7 \left(\frac{1}{T^2}\right) \tag{11}$$

$$\log_{10} \theta_p = C_4 \times 10^2 - C_5 \times 10^5 \left(\frac{1}{T}\right) + C_6 \times 10^7 \left(\frac{1}{T^2}\right) \tag{12}$$

The coefficients C_1 to C_6 are provided by the authors and are obtained with model fitting procedures such as the Box complex technique (Ray; Saraf; Gupta,

1995). This model also included a parameter f_{s0}, which is the initial volume fraction of solvent in the reaction mixture, in the calculation of f. This factor is an attempt to correct the efficiency factor *f* for the initiator.

The results of the RSG model were satisfactory for the set of MMA polymerization data, which is the same as of that from CCS. In addition to that dataset, a second experiment intended to evaluate the model in solution polymerization was provided. In this case, benzoyl peroxide was the initiator chosen for the experiments. However, this model will fail to predict experiments which are not inside the range of temperature and initiator concentration, such as the MMA polymerization with 0.1% AIBN at 50°C (Tefera; Weickert; Westerterp, 1997), as can be observed in Figure 7.

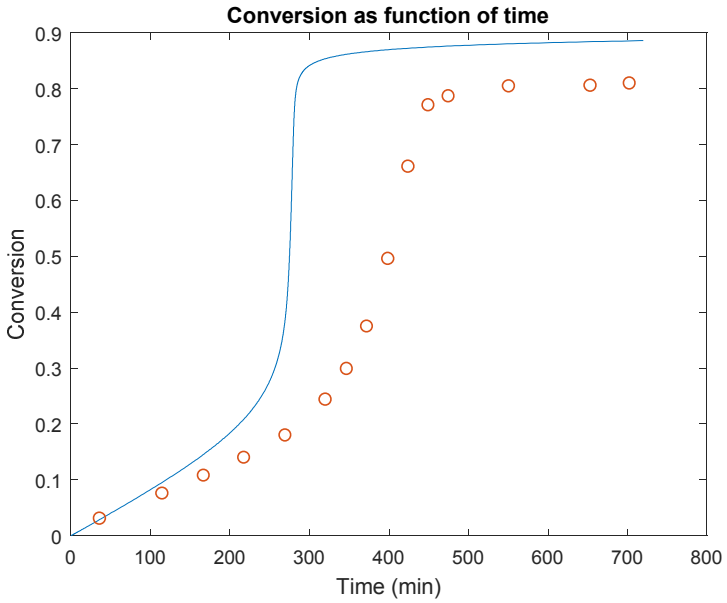


Figure 7: MMA polymerization with RSG model. Theoretical compared with experimental data with 0.1% AIBN and T=50°C.

This lack of fitness for experiments outside the range of the adjusted data in these diffusion models had been previously pointed out by Achilias (2007) and is still one of the main challenges for generalized models.

Another predictable limitation is the use of f_{s0} in the calculation of the efficiency factor f. This model does not account for the possibility of intermediate addition of solvents or monomers into the reaction mixture as in semi-batch polymerizations.

2.4.5. SG Model

Seth and Gupta (1995) published a new model, improving the previously mentioned RSG model with one additional new equation to calculate the initiator efficiency factor f. This equation resembles the equations for k_t (7) and k_p (8), providing a good model for this phenomena.

$$f = \frac{1}{f_0}\left[1 + \theta_f \frac{M}{V_r} \frac{1}{exp\{\xi_{13}(-\psi + \psi_{ref})\}}\right] \tag{13}$$

The parameter θ_f is the diffusion parameter for initiator radicals in the cage and is equivalent to the parameters θ_t and θ_p. f_0 is the intrinsic efficiency factor. The parameter θ_f is obtained with the adjustment of curves such as those for θ_t and θ_p from RSG. This polynomial is initiator-specific, meaning each initiator will have a different polynomial (reflecting the diffusion behavior of their radicals). For AIBN this polynomial equation is:

$$\log_{10} \theta_f = C_1 \times 10^2 - C_2 \times 10^5 \left(\frac{1}{T}\right) + C_3 \times 10^7 \left(\frac{1}{T^2}\right) \tag{14}$$

Coefficients C_1 to C_3 are provided by the authors in the article. The polynomial for BPO is also provided. Moreover, the article presents a new polynomial equation for θ_t, with parameters adjusted to take into consideration the diffusion computation for θ_f. The parameter θ_f remains unchanged.

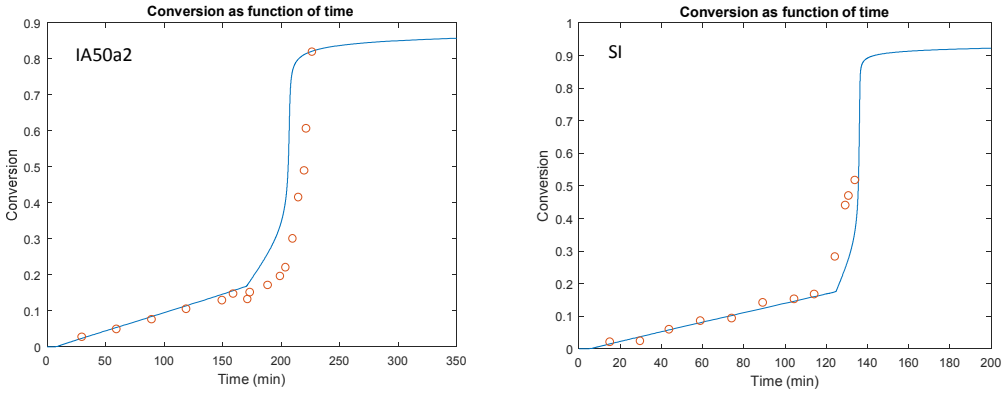


Figure 8: SG model of MMA polymerization with a) intermediate feed of monomer and initiator (IA50a2) and b) step increase of temperature from 50°C to 70°C (SI).

This model was tested on a series of conditions, including an intermediate feed of initiator and monomer (Figure 8). Also, step increase and decrease of

temperature were tested, with relatively good agreement between theoretical and experimental data.

2.4.6. MH Model

Marten and Hamielec (1982) developed a model based on free volume.

$$k_t = k_{t0}\left(\frac{Mw_{crit}}{Mw}\right)^{1.75}\exp\left[-C_2\left(\frac{1}{V_f}-\frac{1}{V_{f_{crit}}}\right)\right] \quad (15)$$

$$k_p = k_{p0}\exp\left[-C_1\left(\frac{1}{V_f}-\frac{1}{V_{f_{crit}}}\right)\right] \quad (16)$$

In these equations, Mw is the weight-average molecular weight, and Mw_{crit} is the critical weight-average molecular weight. C_1 and C_2 are parameters of the model. One of the critical aspects of this model is the need to determine Mw_{crit}, which depends on the experimental conditions.

This model also does not calculate corrections for the initiator efficiency factor f.

2.4.7. CB Model

Another approach establishing a model for k_p and k_t calculation is the work from Curteanu and Bulacovschi (1999). The CB model is a semi-empirical model, based on four parameters for each of the kinetic pseudo-constants:

$$k_t = k_{t0}\exp(A_1 + A_2X + A_3X^2 + A_4X^3) \quad (17)$$

$$k_p = k_{p0}\exp(B_1 + B_2X + B_3X^2 + B_4X^3) \quad (18)$$

The constants A_1 to A_4 and B_1 to B_4 were obtained by curve fitting with the experimental data. A good agreement could be obtained with this model, as it can be seen in Figure 9.

One limitation of this model is the requirement of parameters which are specific for this polymerization setting, and are dependent on initiator concentration and temperature. In this article, the parameters are supplied on a table and have to be selected according to the required experiment.

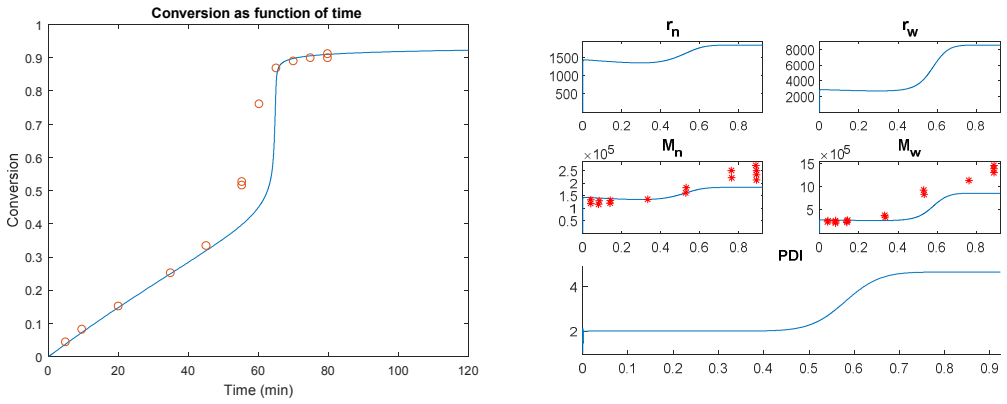


Figure 9: CB model tested with experimental data from Marten and Hamielec (1979)

Other possible limitation is the use of this model in other situations different than those used in the determination of the parameters, such as the use of chain transfer agents in the formulation. This is expected because the equations are functions of conversion only, being invariant to the presence of CTA's and the resulting changes in molecular weight. In those conditions, a new set of parameters would have to be determined. This model also does not model the initiator efficiency factor f, and it is treated as a constant.

2.4.8. SBGSG Model

Sangwai et al. (2005) provided a model where the constants of the CB model are automatically calculated as a function of temperature. Each of the parameters is calculated with a linear equation based on two other parameters:

$$A_i = A_{i1}(T - 273.15) + A_{i2} \quad (19)$$

$$B_i = B_{i1}(T - 273.15) + B_{i2} \quad (20)$$

Subscripts $i = 1, 2, .., 4$. Parameters A_i and B_i are then used in equations (17) and (18). These sum up to a total of 16 parameters required for the model. The model provides a good agreement between simulation and experimental data.

The limitations are the same as the CB model. The improvement in the continuous function to calculate the exponential parameters is offset by the doubling of the number of parameters required.

2.4.9. Inconsistency between SG and SBGSG models

Sangwai et al. (2005) pointed out an apparent inconsistency in one experimental result used in the development of the SG model. The experiment SD1, which stands for a step decrease in temperature, from 70°C to 50°C after 20 min, presents an unexpected behavior. According to the authors, one would expect this experiment to fit within the region delimited by the isothermal curves at 50°C and 70°C for the same initiator concentration (AIBN=0.0258 mol/L), but the data does not present such expected behavior, as shown in Figure 10.

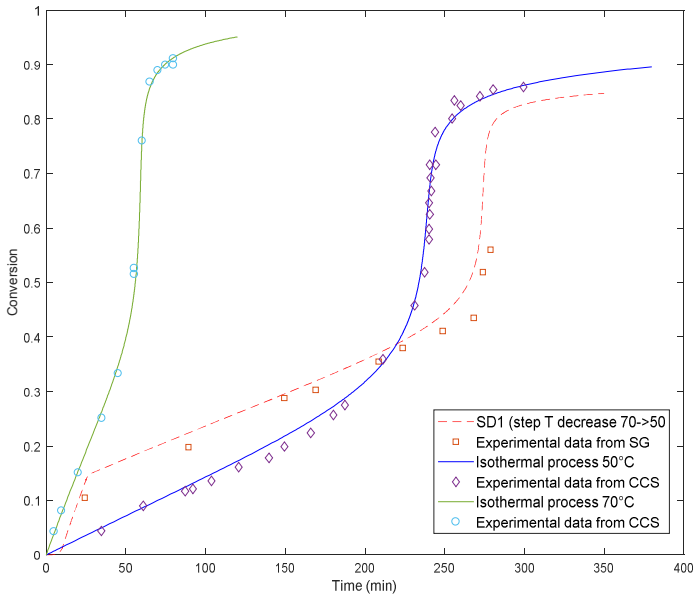


Figure 10: Experiment SD1 (step temperature decrease) as modeled by SG, compared with isothermal curves (AIBN=0.0258 mol/L) and with old experimental data (Seth; Gupta, 1995).

The argument of an intermediary curve itself is debatable because the polymerization at a higher temperature (70°C) would imply in a concentration of lower molecular weight polymer chains in the medium. On the other hand, the polymer fraction is higher than the isothermal process at 50°C, as more polymer was formed because of the higher temperature. In this manner, both effects would offset each other to a certain degree.

At the point where the experiments SD1 and isothermal at 50°C intersect, one would expect the same polymer fraction (same conversion), but an average Mn lower for SD1. This intersection point occurs at time = 220 min and conversion

= 38.85%. Considering the calculated molecular weight for this intersection point (Table 1), the delayed onset of the gel effect for the experiment SD1 seems to be reasonable.

Table 1: Molecular weight at t=220 min and X=38.85% of experiments of figure 10.

	Isothermal 50°C	SD1
Mn	5.43E+05	2.34E+05
Mw	1.18E+06	5.62E+05

In the same article (Sangwai et al., 2005), the authors present a new set of experimental data (Figure 11), suggesting that the previous experimental data, used for experiment SD1, could have been invalid. In this dissertation, we will consider the latest experimental data. If the results of this latter experiment are confirmed by future experiments, it must be kept in mind that the SG model might be imprecise in conditions with step changes in temperature.

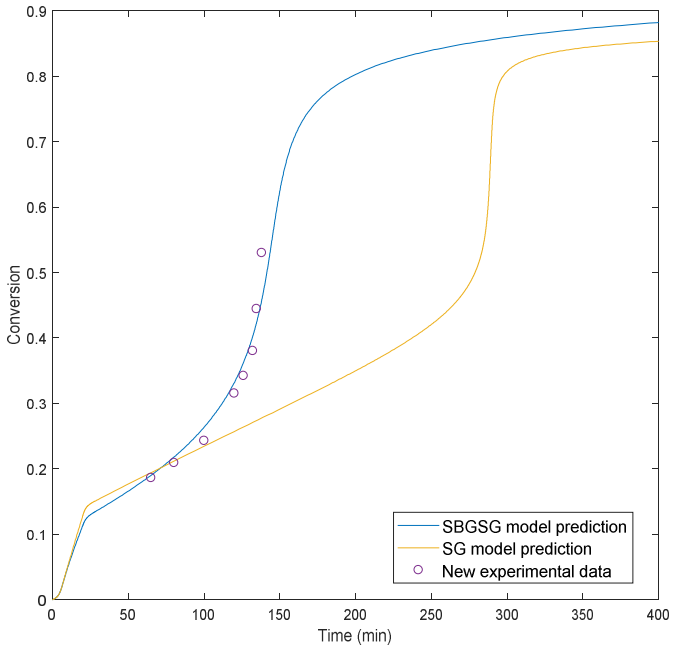


Figure 11: Step decrease temperature with new experimental data (Sangwai et al, 2005), as predicted by SBGSG (good fit) and SG (poor fit).

2.4.10. Summary of the reviewed models

The kinetic-diffusion models previously reviewed were evaluated by mathematical models consisting of a set of ODEs to determine:

- Reaction rate equations for each of the relevant reactions in the system (using k_p, k_t and f calculated with the previous models).
- Moments of free-radicals and dead polymer chains equations (required to calculate Mn and Mw), by using the method of moments, which has been recently reviewed by Mastan and Zhu (2015). As these moments are derived from the kinetic model, also the calculated k_p, k_t and f are used here.
- Mass balance equations (required for semi-batch polymerization and to account for phase changes, such as evaporation of the components)
- Energy balance equations (non-isothermal processes).

Some of the models, such as RSG, adopt simplifications, such as no termination for combination $(k_{tc} = 0)$ and no radical transfer to solvents $\left(k_{fm} = 0\right)$ in the case of MMA polymerization.

Table 2 summarizes the differences among the different models reviewed in this dissertation. This list of models is not exhaustive, but models are usually derived from one of the listed below.

Table 2: Comparison of the different models covered in section 2.4.

Model	Functions	f	ktc	kfm
CCS	Experiment-specific parameters	Constant	0	NI
AK	Continuous	Constant	Calculated	Calculated
RSG	Continuous	Modeled	0	0
SG	Continuous	Modeled	0	0
MH	Continuous	Constant	N/A	N/A
CB	Experiment-specific parameters	Constant	NI	NI
SBGSG	Continuous	Constant	0	Calculated

NI: not included in the equations.
N/A - not applicable (article based on styrene)

The models AK and MH were not further explored because of either complexity, lack of data, and/or limitations mentioned in the respective topics. Nonetheless, these models are not discarded for future analysis. Models CCS and CB are used for comparison, but these models are limited to a general application due to their experiment-specific character. Model SG is an evolution of the model RSG, but both are evaluated separately.

2.5. Other aspects of the MMA polymerization

This topic covers other essential aspects of the modeling of MMA polymerization, such as relevant reaction mechanisms, (e.g., thermal self-initiation, termination by combination, radical transfer to monomer and depolymerization), as well as other aspects of the polymerization process (e.g., heat of reaction).

2.5.1. Combination and disproportionation termination

Termination by combination occurs when two polymeric radicals react, creating an inert polymer chain which degree of polymerization is the sum of both radicals. Termination by disproportionation will occur by the abstraction of labile hydrogen from one polymer chain into the other, where the chain which suffered the hydrogen abstraction will stabilize itself by forming an internal double bond (Figure 12):

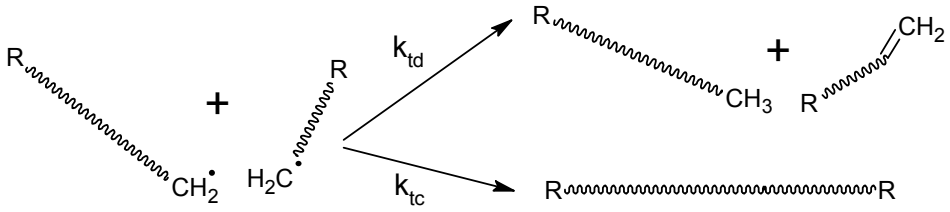


Figure 12: Disproportionation (k_{td}) and combination (k_{tc}) termination mechanisms.

The double bond formed at the end of one of the polymer chains could, in theory, react with a free radical, and restart the polymerization. This mechanism has been pointed out by Balke (1972) as very difficult to account for, and for that matter, usually, it is not considered in the models present in the literature.

The ratio of termination by disproportionation and by combination in MMA was first studied by Bevington and Melville (1954). This rate was found to be dependent on temperature.

The accounting of combination is important to determine the accurate molecular weight of the polymer. As Mn is also used in the kinetic models, the simplification $(k_{tc} = 0)$ could also affect the diffusional models. In order to consider the ratio of both termination reactions, equation (21) is used by some of the authors for MMA (Achilias; Kiparissides, 1988):

$$k_{tc}/_{k_{td}} = 3.956 * 10^{-4}\exp(4.09/(1.987\text{e-}3*\text{T})) \quad (21)$$

The models using the simplified $k_{tc} = 0$ showed reasonable agreement between theoretical and experimental data (Chiu; Carratt; Soong, 1983; Ray; Saraf; Gupta, 1995; Seth; Gupta, 1995), demonstrating that the simplification is not necessarily detrimental to the polymerization model of MMA in the temperature range studied by the authors. For those cases, k_t = k_{td}, since k_t = k_{tc} + k_{td}.

2.5.2. Radical transfer to monomer

The radical transfer (Farina, 1987) or chain transfer is another reaction which needs to be taken into account in polymerization systems because they decrease the overall molecular weight of the end polymer. Similar mechanisms may be required for other reaction components when applicable, such as solvents. The specific case of monomers is somewhat relevant, especially in bulk polymerization (high monomer concentration) and emulsion polymerization (Figure 13):

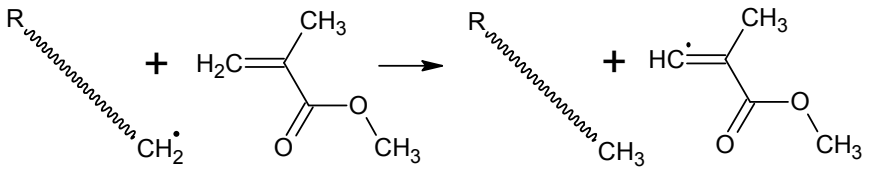


Figure 13: Radical transfer from growing polymer chain to monomer (MMA)

In the case of emulsion polymerization, transfer to monomer is said to be the most critical mechanism for radical exiting from the polymer particles. Such radical exiting mechanism is essential to explain why the average number of radicals per particle can be lower than the theoretical ½ due to compartmentalization (Gilbert, 1995). Thus, it is desirable to consider the radical transfer of monomers in the model so that it can be used in emulsion polymerization systems.

k_{fm} is the rate constant for chain transfer to monomer, and is modeled by equation (22) in the particular case of MMA (Achilias; Kiparissides, 1988):

$$k_{fm}/k_p = 9.48 * 10^3 \exp(\text{-}13.88/(1.987\text{e-}3*\text{T})) \quad (22)$$

Equation (22) establishes a direct dependence of k_{fm} with k_p. Such dependence allows the accurate modeling of radical transfer in glassy mediums, where even the diffusion of small radicals and molecules is limited. The limitation for small molecules (monomer and monomeric radical) is determined by k_p, as seen on previous diffusion models, and it is applied automatically to k_{fm} with the use of equation (22).

Even though the radical transfer to MMA occurs at temperatures as low as 50°C (Kukulj; Davis; Gilbert, 1995), the simplification adopted by some of the models $(k_{fm} = 0)$ did not influence the results to a significant extent. It is necessary to point out that the net effect of radical transfer to monomer is the decrease of Mn and Mw, so it can be important for predicting molecular weight but not for predicting the rate of polymerization. This effect is opposite to the effect of termination ratio k_{tc}/k_{td}, even though they may be different by orders of magnitude.

2.5.3. Monomer thermal self-initiation

One possible mechanism for polymerization is the thermal self-initiation of the monomer. This mechanism can be of major importance in processes at high temperature and for monomers prone to this mechanism, such as styrene (Pryor; Coco, 1970). MMA is a monomer with relatively low self-initiation (Walling; Briggs, 1946). Because of the negligible effect, it is common practice to not include this mechanism in models of MMA polymerization. Recent studies have explored the decomposition at high temperature, and the results suggest that this simplification is safe and not expected to affect models significantly (Nising et al., 2005; Nising, 2006).

2.5.4. Depropagation

It is crucial to consider the ceiling temperature (T_c) in processes carried out at high temperatures, which can be achieved in pressurized vessels (otherwise, the temperature will remain constant at the boiling point of the

monomer or solvents). At temperatures above the ceiling temperature, the reverse reaction, depolymerization, is favored, causing the breakdown of polymer chains previously formed. As the ceiling temperature for the polymerization of MMA is relatively high as compared with process temperatures commonly used, this reaction may be disregarded when $T<T_c$ (Louie; Carratt; Soong, 1985).

The depolymerization will have a net effect on k_p according to the following model (Matyjaszewski, 2002):

$$k_p^{eff} = k_p - \frac{k_{de}}{[M]} \tag{23}$$

The equilibrium constant is established by the monomer equilibrium concentration (Matyjaszewski, 2002):

$$\frac{k_p}{k_{de}} = \frac{1}{[M]_{eq}} \tag{24}$$

2.5.5. Heat of the polymerization reaction

Acrylic polymerizations are highly exothermic in nature (Figure 14), and if the temperature is not well controlled, a runaway reaction may take place in the reactor. Such a condition imposes a severe risk to the process and has to be closely monitored.

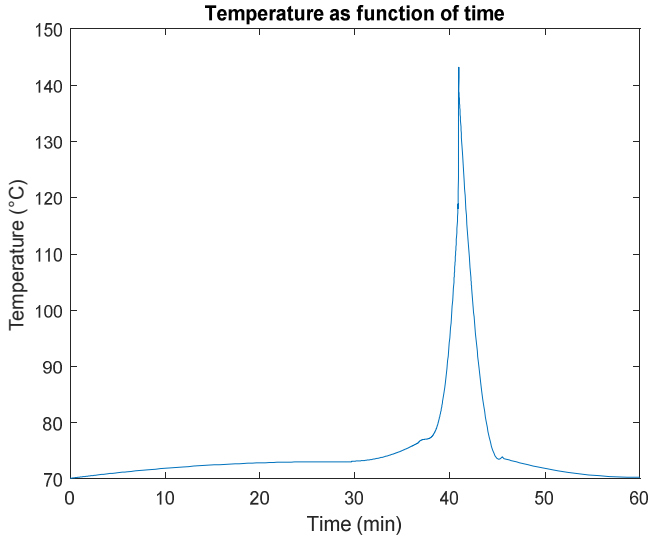


Figure 14: Temperature variation of a non-isothermal process of MMA polymerization due to the heat of reaction (Armitage et al., 1988)

Equation (25), the energy balance, provides a way to calculate temperature changes in a batch polymerization process (Louie; Carratt; Soong, 1985):

$$\rho C_p V_r \frac{dT}{dt} = -\Delta H_p k_p [M][\lambda_0] - UA_{ht}(T - T_s) \quad (25)$$

Most of the models in the literature for MMA are based on isothermal approach ($\frac{dT}{dt} = 0$). Nonetheless, some studies investigate the behavior of MMA polymerization under non-isothermal conditions. Louie, Carratt, and Soong (1985) evaluated a model for this condition, and their results achieved good agreement between experimental and theoretical data.

In order to make the reproducibility of results more manageable, some studies use the experimentally measured temperature-time history as an input for the model. Based on this input, the rate constants at each instant of the process is calculated. This approach avoids the solution of the energy balance simultaneously with the mass balances and the necessity of determining the heat transfer coefficient between the reactor content and the fluid flowing in the jacket.

Seth and Gupta (1995) provided this history for their experiments with step increase and decrease of temperature. Ghosh, Gupta, and Saraf (1998) provided a comprehensive list of experiments with their respective temperature history, including near isothermal processes. Such history equations for T(t) allow the reproduction of the results without the concern with the correct determination of heat exchange parameters.

CHAPTER III

3. MATHEMATICAL MODEL

This chapter presents the mathematical modeling used in the program *Reactormodel*, developed in this dissertation as the framework for model investigations. This modeling includes reaction mechanisms, kinetic models, and process parameters. The program *Reactormodel* is detailed on chapter IV.

3.1. Reaction Mechanisms

The starting point of the modeling of acrylic polymerizations is the determination of the relevant reactions to be considered in the mechanism. The reactions considered for this model are listed in Table 3:

Table 3: Reactions considered in the new model.

Reaction	Mechanism	
Initiator decomposition:	$I \overset{k_d}{\rightarrow} 2fR_I^*$	(26)
Initiation:	$R_I^* + M \overset{k_i}{\rightarrow} R_1^*$	(27)
Propagation:	$R_n^* + M \overset{k_p}{\rightarrow} R_{n+1}^*$	(28)
Depropagation:	$R_{n+1}^* \overset{k_{de}}{\rightarrow} R_n^* + M$	(29)
Radical transfer to CTA:	$R_n^* + CTA \overset{k_{fc}}{\rightarrow} D_n + R_{CTA}^*$	(30)
Radical transfer to monomer:	$R_n^* + M \overset{k_{fm}}{\rightarrow} D_n + R_M^*$	(31)
Radical transfer to solvent:	$R_n^* + S \overset{k_{fs}}{\rightarrow} D_n + R_s^*$	(32)
Radical transfer to polymer:	$R_n^* + D_m \overset{k_{fp}}{\rightarrow} D_n + R_m^*$	(33)
Termination by disproportionation:	$R_m^* + R_n^* \overset{k_{td}}{\rightarrow} D_m + D_n$	(34)
Termination by combination:	$R_m^* + R_n^* \overset{k_{tc}}{\rightarrow} D_{m+n}$	(35)

A few simplifications are considered in the implementation of this model:

- R_i^*=R_0^* (implicitly $k_i = k_p$), simplifying the initiation mechanism (not rate-limiting).
- R_{CTA}^*=R_0^* and R_M^*=R_1^*, simplifying the radical transfer mechanisms (both not rate-limiting).
- $k_{fs} = 0$ (no transfer to solvent), considering this mechanism negligible for the analyzed cases (but this mechanism may be relevant, depending on the solvent used in the system. Reaction (32) is currently not included in the algorithm.
- $k_{fp} = 0$ (no transfer to dead polymer chains), considering this mechanism negligible for the analyzed cases (but this mechanism may be relevant, depending on the polymer). Reaction (33) is currently not included in the algorithm.

In the list of reaction mechanisms displayed in Table 3, depropagation is the one that stands out as compared with the previous models analyzed in this dissertation. Because those models studied only ranges of temperature between 50°C and 70°C, such a mechanism is treated as negligible (which it is, on those conditions). In the case of MMA, the depropagation becomes relevant at higher temperatures, above Tg (see discussion on topic 2.5.4).

One particular mechanism not included in the model is the polymerization of double bonds present in chains formed by the mechanism of termination by disproportionation (Figure 12), because of the lack of experimental kinetic data.

3.2. Molecular weight

The reactions in Table 3 are the basis for setting the mass balances for each species. Due to the large and undetermined number of equations necessary to calculate the variation of radical and dead polymer chains of all sizes (subscripts m and n), the method of moments is adopted to calculate the first, second and third moments of radicals and dead polymer, following the procedure discussed elsewhere (Mastan; Zhu, 2015):

$$\lambda_i = \sum_{j=0}^{\infty} j^i R_j^* \tag{36}$$

$$\mu_i = \sum_{j=0}^{\infty} j^i D_j \tag{37}$$

The exponent $i = 0, 1, 2$ is used to calculate the first, second and third moments, respectively. These moments are then computed in the ODE set with the inclusion of only six additional equations. The demonstration of the calculation performed for this model is provided in Appendix A.

It is interesting to note that, from these moments, one is able to calculate Mn, Mw and the polydispersity index (PDI):

$$Mn = \frac{\mu_1 + \lambda_1}{\mu_0 + \lambda_0} \mathrm{MM}_{mon} \cong \frac{\mu_1}{\mu_0} \mathrm{MM}_{mon} \tag{38}$$

$$Mw = \frac{\mu_2 + \lambda_2}{\mu_1 + \lambda_1} \mathrm{MM}_{mon} \cong \frac{\mu_2}{\mu_1} \mathrm{MM}_{mon} \tag{39}$$

$$PDI = \frac{Mw}{Mn} = \frac{\mu_2 \mu_0}{{\mu_1}^2} \tag{40}$$

where MM_{mon} is the molar mass of the monomeric unit, and $\mu_i + \lambda_i \cong \mu_i$.

3.3. Model equations for batch and semi-batch processes

The set of coupled ODEs for this model was developed considering the reaction mechanisms described in Table 3 and the simplifications described in the previous topic. These equations are applicable for batch and semi-batch reactors, for both bulk and solution polymerization:

$$\frac{dM}{dt} = \frac{-(k_p - \frac{k_{de}}{M} + k_{fm}) M \lambda_0}{V_r} + F_M H(M_{FT}) \tag{41}$$

$$\frac{dI}{dt} = -k_d I + F_I H(I_{FT}) \tag{42}$$

$$\frac{dS}{dt} = F_s H(S_{FT}) \tag{43}$$

$$\frac{dCTA}{dt} = \frac{-f_{CTA} k_{fc} CTA}{V_r} + F_{CTA} H(CTA_{FT}) \tag{44}$$

$$\frac{d\lambda_0}{dt} = 2fk_dI - \frac{k_t{\lambda_0}^2}{V_r} \tag{45}$$

$$\frac{d\lambda_1}{dt} = \frac{\left(k_p - \frac{k_{de}}{M}\right)M\lambda_0 - k_t\lambda_0\lambda_1 + \left(k_{fm}M + f_{CTA}k_{fc}CTA\right)(\lambda_0 - \lambda_1)}{V_r} \tag{46}$$

$$\frac{d\lambda_2}{dt} = \frac{\left(k_p - \frac{k_{de}}{M}\right)M(2\lambda_1 + \lambda_0) - k_t\lambda_0\lambda_2 + \left(k_{fm}M + f_{CTA}k_{fc}CTA\right)(\lambda_0 - \lambda_2)}{V_r} \tag{47}$$

$$\frac{d\mu_0}{dt} = \frac{\left(k_{td} + \frac{k_{tc}}{2}\right){\lambda_0}^2 + (k_{fm}M + f_{CTA}k_{fc}CTA)\lambda_0}{V_r} \tag{48}$$

$$\frac{d\mu_1}{dt} = \frac{k_t\lambda_0\lambda_1 + (k_{fm}M + f_{CTA}k_{fc}CTA)\lambda_1}{V_r} \tag{49}$$

$$\frac{d\mu_2}{dt} = \frac{k_{tc}\left(\lambda_0\lambda_2 + {\lambda_1}^2\right) + k_{td}\lambda_0\lambda_2 + (k_{fm}M + f_{CTA}k_{fc}CTA)\lambda_2}{V_r} \tag{50}$$

All species are modeled with the absolute number of mols in the reactor and in the feed tanks (as opposed to models based on concentration). The operator H() is Heaviside's function, which computes 1 for positive values and 0 for negative and zero values (the molar feed is only applicable when the number of moles of the referred species is a positive value). A set of auxiliary ODEs to compute the cumulative number of mols fed to the reactor in semi-continuous processes is provided:

$$\frac{dM_{fr}}{dt} = F_M H(M_{FT}) \tag{51}$$

$$\frac{dI_{fr}}{dt} = F_I H(I_{FT}) \tag{52}$$

$$\frac{dS_{fr}}{dt} = F_S H(S_{FT}) \tag{53}$$

$$\frac{dCTA_{fr}}{dt} = F_{CTA} H(CTA_{FT}) \tag{54}$$

When the quantities of initiator and CTA are much lower than the monomer and solvent (situation applicable in most cases), equations (52) and (54) can be skipped.

The following equations model the variation of materials quantity in the feed tanks:

$$\frac{dM_{FT}}{dt} = -\frac{dM_{fr}}{dt} \tag{55}$$

$$\frac{dI_{FT}}{dt} = -\frac{dI_{fr}}{dt} \tag{56}$$

$$\frac{dS_{FT}}{dt} = -\frac{dS_{fr}}{dt} \tag{57}$$

$$\frac{dCTA_{FT}}{dt} = -\frac{dM_{fr}}{dt} \tag{58}$$

For non-isothermal reactors, the temperature variation due to the heat of the propagation reaction can be included, including the modification for depropagation:

$$\rho C_p V_r \frac{dT}{dt} = -\Delta H_p \left(k_p - \frac{k_{de}}{M}\right) M \frac{\lambda_0}{V_r} - UA_{ht}(T - T_s) + \sum_{K=M,I,S,CTA} F_K H(K_{FT}) \tilde{C}_{P,K}(T_K - T) \tag{59}$$

The model developed here considers a single-phase reaction system. In special cases, such as the polymerization at temperatures above the boiling point of MMA, it is considered that the reactor is pressurized. Nonetheless, reactors with two phases (liquid and vapor) can be modeled by including the relevant set of thermodynamic and mass balance equations.

3.4. Output Parameters

The output typically evaluated in the modeling of MMA polymerization are Conversion (X_m), Mn and Mw. Conversion is calculated in terms of the mass fraction of the polymer in the reactor:

$$X_m = \frac{m_{pol}}{m_{pol} + m_{mon} + m_{sol} + m_{ini} + m_{CTA}} \tag{60}$$

The masses of initiator m_{ini} and CTA m_{CTA} may be neglected when their concentrations are much lower than the masses of polymer, monomer, and solvent. Note that the conversion X_m, calculated with (60), is similar to the conversion calculated on a molar base in bulk polymerization (61).

$$X_{molar} = 1 - \frac{M}{M_{fr}} \tag{61}$$

In fact, equation (61) is a particular case of (60). Equation (60) is used in this model, because of its generality, required for solution polymerization.

Mn and Mw are directly calculated from equations (38) and (39). The instantaneous degree of polymerization (P_n), commonly used in the literature (O'Brien; Gornick, 1955), is calculated as follows, including the correction for depropagation:

$$P_n = \frac{\left(k_p - \frac{k_{de}}{M}\right) M\lambda_0}{(k_{tc}{\lambda_0}^2 + 2k_{td}{\lambda_0}^2 + k_{fm}M\lambda_0 + k_{fc}CTA\lambda_0)} \tag{62}$$

3.5. Kinetic and diffusion parameters

The equations developed in the previous topic are applicable to all models included in *Reactormodel*. For some of these models, a few parameters may be considered as zero, such as k_{tc}. The simplifications adopted by the authors were previously described in Table 2.

The difference in the implementation of the different models is the calculation of the kinetic pseudo-constants (performed in constants_calc.m in *Reactormodel*), which are then used by the ODE set developed previously.

These calculations follow the models discussed on topic 2.4, with a few modifications to generalize them to the present reactor model. Equation (13) is modified (instead of monomer fraction, making it a function of polymer fraction) so it can be used in solution polymerization on the newly proposed models:

$$f = \frac{1}{f_0}\left[1 + \theta_f(1 - \varphi_p)\frac{1}{exp\{\xi_{13}(-\psi + \psi_{ref})\}}\right] \tag{63}$$

Another set of equations worth of mention is that from (17) and (18), which was originally used with conversion based on molar fractions, but it can also be directly used with polymer mass fraction:

$$k_t = k_{t0}\exp(A_1 + A_2 X_m + A_3 {X_m}^2 + A_4 {X_m}^3) \quad (64)$$

$$k_p = k_{p0}\exp(B_1 + B_2 X_m + B_3 {X_m}^2 + B_4 {X_m}^3) \quad (65)$$

This substitution allows its direct use in solution polymerization processes. All the remaining equations are used as they are presented on topic 2.4.

CHAPTER IV

4. METHODOLOGY

This chapter discloses the settings used in this study, such as the selected scope of the experiments, software, and experimental data sources.

4.1. Scope of the experiments

As previously mentioned, methyl methacrylate (MMA) has been chosen as the monomer for this polymerization study. The decision relies on the widespread availability of studies and experimental data. MMA has been extensively studied because of its remarkable gel and glass effects when polymerized below its transition temperature (Tg).

The experimental data used in the present study were obtained from the literature. These experiments use azo-bis-isobutyronitrile (AIBN) or benzoyl peroxide (BPO) as initiators. AIBN has been chosen as the initiator for several academic studies of diffusion and is the initiator used in most of the experiments that are evaluated in this dissertation. Experiments with BPO were included in the work of Ray, Saraf, and Gupta (1995).

In this dissertation, the use of chain transfer agents (CTA) is also considered. The use of this mechanism in MMA has been reviewed elsewhere (O'Brien; Gornick, 1955; Destruel; Taufer, Granel, 1991; Madruga; San Román; Benedi, 1990).

This dissertation focuses on bulk and solution polymerization, which are commonly used by the adhesives and coatings industry. Nonetheless, the results are expected to be useful for emulsion polymerization, especially the calculation of particle-exiting free radicals resulting from radical transfer to monomers, and suspension polymerization.

Batch and semi-batch reactors are both considered in the models, as previously discussed. It is assumed that all processes are carried out under pressure, so there is a single phase even at temperatures above the boiling point of the reactants.

4.2. Modeling Software

Initial studies were carried out with Polymath Professional version 6.1 (Figure 15). Polymath kindly provided a free upgrade from the educational version. The professional version can handle a larger number of ODEs and auxiliary equations. This program was very useful to analyze simple models such as CCS. Polymath also generates an algorithm code, which can be used in Matlab. The structure of this algorithm from Polymath for ODE solution was adopted in this study, and it is underlying the final code. One example of the CCS model in Polymath is provided in Appendix B.

Later on, because of the increased complexity involved in the models (e.g., plot capability, Boolean operators, logical loops), Matlab version R2017 was used as the primary platform for the new models. Matlab is an interpreted programing language specialized in scientific and engineering applications. It has built-in functions for optimization and integration of the ODE systems and more flexibility in data analysis (e.g., plots, output files generation).

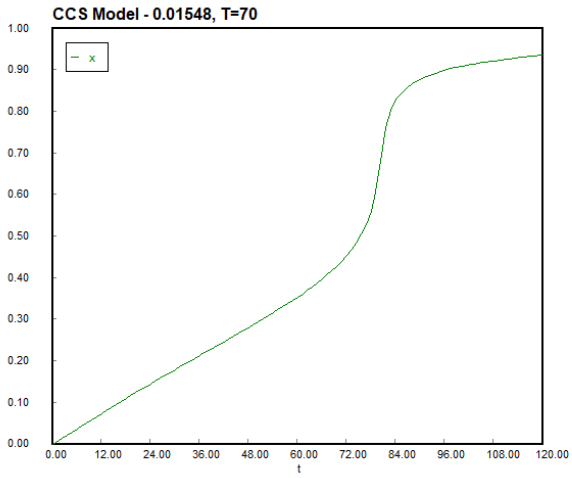


Figure 15: Conversion of MMA with CCS model in Polymath

All models were run on a desktop personal computer (PC) based on Windows 10 64 bits, with a processor Intel i7 7740 overclocked to 5GHz and 32 GB of RAM. The integration models are solved within seconds. Optimization routines, used to adjust parameters, could take a few hours to be solved depending on the complexity of the model and the number of parameters to be optimized.

4.3. The program *Reactormodel*

A structured program, with a simplified user interface (command prompt), was developed in Matlab. This code was named *Reactormodel*, and its main routine file is reactormodel.m (Figure 16).

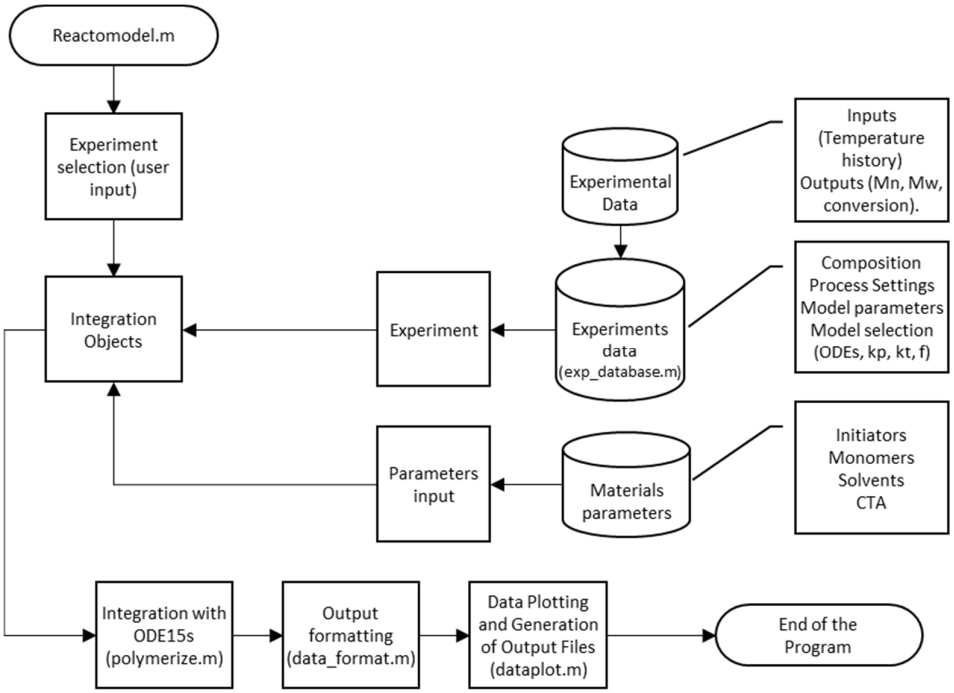


Figure 16: Structure of reactormodel.m

This program was created with the intent to provide a flexible platform for model selection and testing. The model is supported by input files, which contain the composition, and process parameters (temperature, including the possibility to use functions to reproduce the temperature curves used in the literature).

The model setup (ODE equation set and the diffusion and kinetic models to be applied in that experiment) and other relevant parameters (e.g., CCS parameters, individual optimized parameters for RSG model) are also included. These parameters are combined in integration objects, which are used as input for polymerize.m, the integration function of the model.

4.3.1. Integration function - Polymerize.m

The integration will proceed with the calculation of the kinetic and diffusion pseudo-constants, based on the selected model provided via the integration

object. Next step is the integration itself, which is carried out with the ODE set of equations selected to that experiment, as demonstrated in Figure 17.

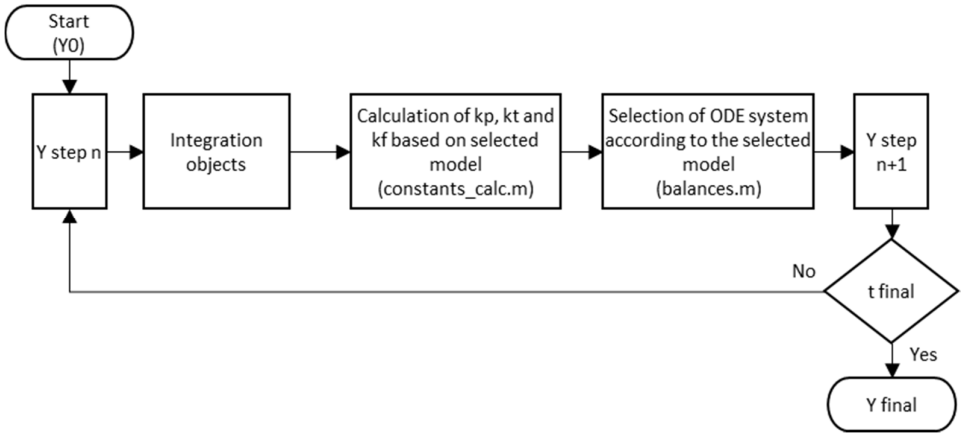


Figure 17: Functions underlying the integration with ODE15s

ODE15s is a solver provided by Matlab, which is suitable for stiff ODE systems, such as the polymerization models.

A relative tolerance (RelTol) of 1e-9 has been adopted, which resulted in an excellent overall performance.

4.3.2. Plots and output files

Reactormodel will generate graphical and text outputs at the end of the integration, as illustrated in Figures 18-22 (examples with the polymerization of MMA with AIBN, I_0=0.01548 mol/L, and T=70°C, from Marten and Hamielec, 1979). The plots are divided in:

- Integration variables as a function of conversion (species concentration, radical and dead polymer moments, temperature, volume).
- The molecular weight as a function of conversion (Mn, Mw, polydispersity index, average number, and average weight molecular weight).
- (Pseudo) constants as a function of conversion (k_p, k_t, k_{tc}, k_{td}, k_{fm}, k_d), f, and degree of polymerization (P_n):
- Free volume parameters as a function of time (fraction volumes of species, free volume, ψ).
- Conversion as a function of time.

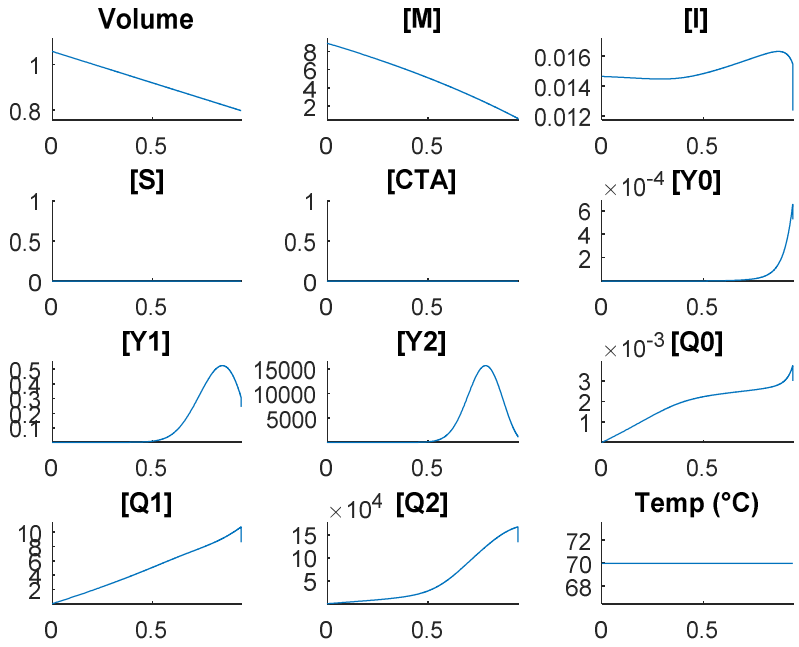


Figure 18: Integration variables in *Reactormodel*.

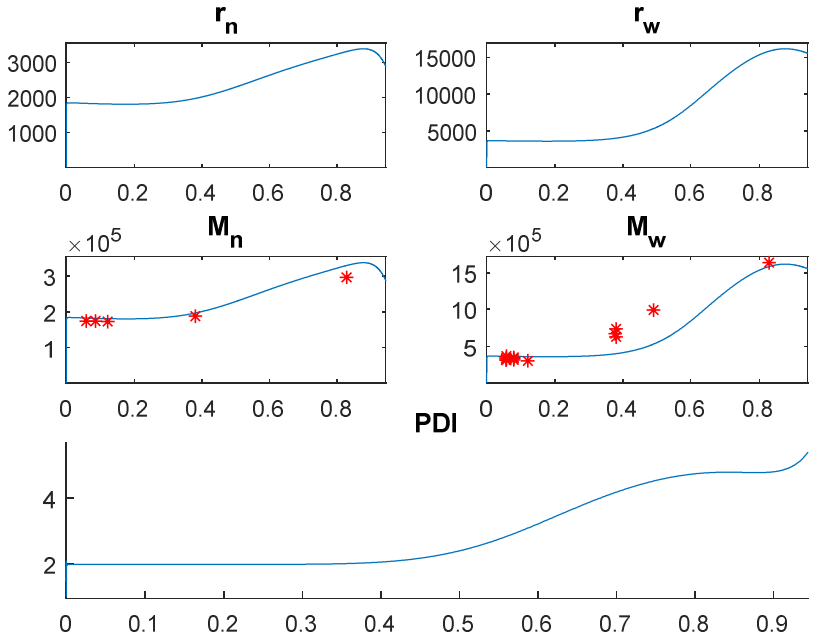


Figure 19: Molecular weight results parameters in *Reactormodel*.

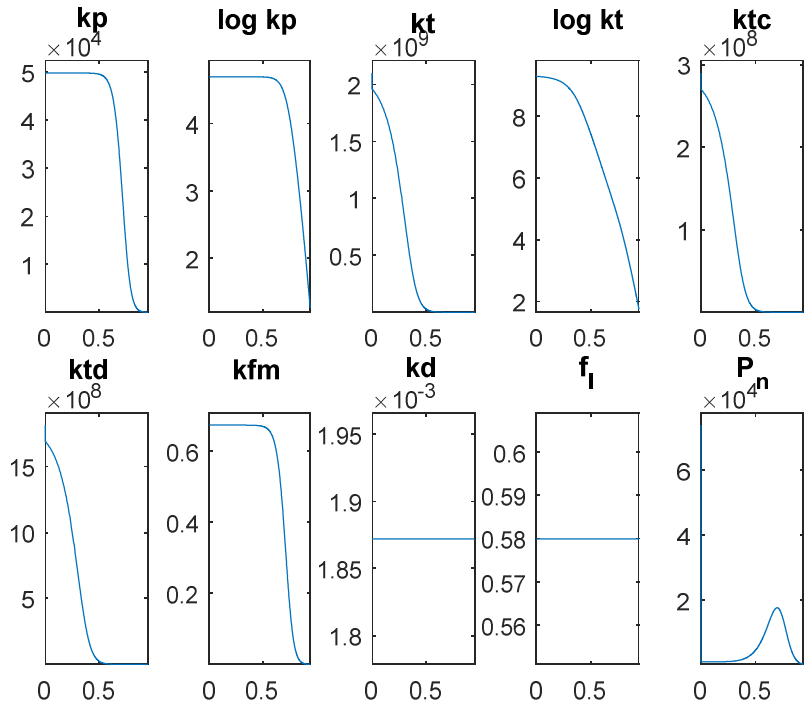


Figure 20: Kinetic and diffusion parameters in *Reactormodel*.

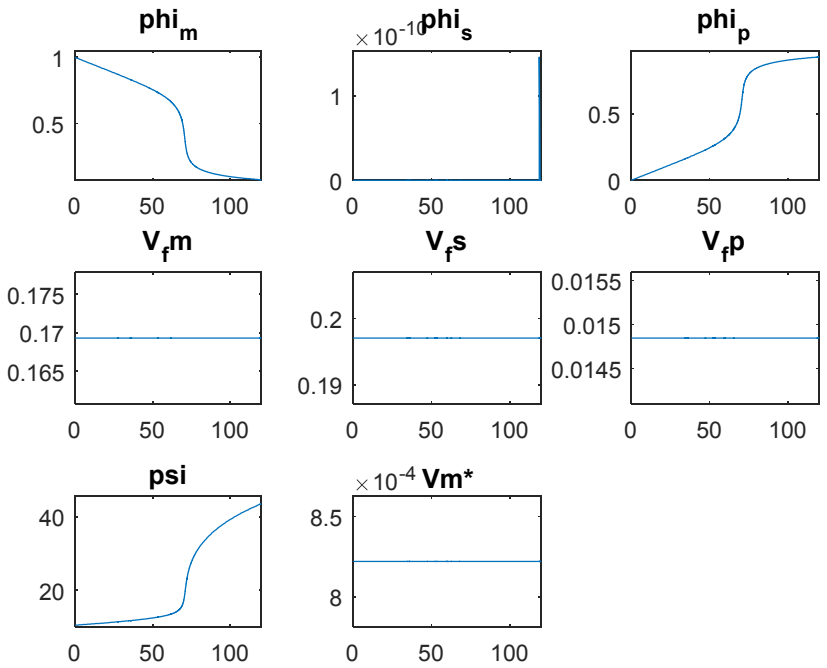


Figure 21: Free volume parameters in *Reactormodel*.

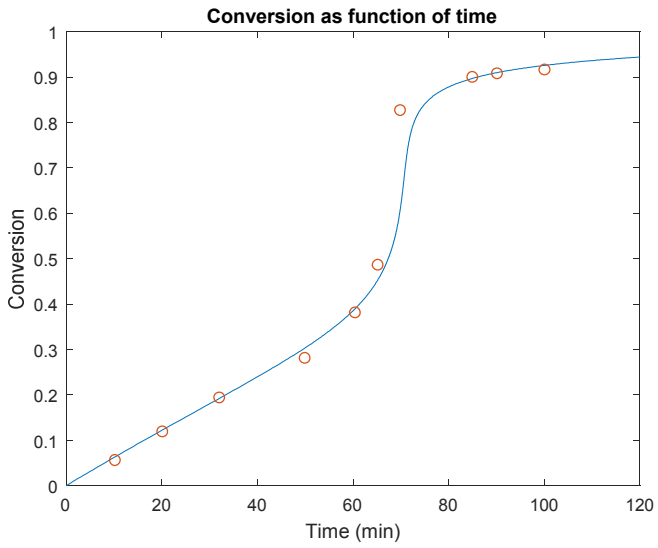


Figure 22: Conversion results in *Reactormodel*.

Conversion, Mn and Mw plots will also display the experimental data whenever they are available, so the agreement between model results and experimental data can be visually inspected.

Reactormodel will also provide a set of files in comma-separated values (.csv) with the numerical results of the integration, which allow the manipulation of data for any additional assessment.

4.3.3. Additional functionalities

A small set of tools was developed in order to automatize operations in *Reactormodel*. These tools are:

- Multi-testing (Option “I”): overrides the models selected in experiments with the options entered by the user, and test a vector of experiments simultaneously. Error data is generated in the “error” folder (files Error_summary_*.csv).
- Multi-plotting (option “G”): plots a vector of experiments in the same graph. The experiment must have been run at least one time (CSV files are generated at first run) before this tool is used.
- Capability to run experiments with different models (option “J”), allowing tests with variations in the experiments stored in the database.

- DoE (option “H”): design of experiments tool, allowing the batch calculation of DoE’s (e.g., using Minitab, a statistical analysis software).

4.4. Experimental data

All models evaluated in this dissertation were tested with experimental data from the literature (see Table 4). The data was extracted with the help of the web-based digital tool WebPlotDigitizer, available at https://automeris.io/WebPlotDigitizer. Each data point was carefully extracted, and a database with this experimental data (experimental_data.m) was created in *Reactormodel*. This data can be selected in experiments, so both can be used in plots and the optimization function.

Table 4: Summary of the experiment sources used in this dissertation.

Exp	Process	Source
2	Isothermal	Marten; Hamielec (1979)
3	Isothermal	Marten; Hamielec (1979)
4	Isothermal	Marten; Hamielec (1979)
5	Isothermal	Marten; Hamielec (1979)
6	Isothermal	Marten; Hamielec (1979)
7	Isothermal	Marten; Hamielec (1979)
25	Isothermal	Tefera; Weickert; Westerterp (1997)
26.1	Isothermal	Tefera; Weickert; Westerterp (1997)
16	Intermediate initiator feed	Ray; Saraf; Gupta (1995)
18	Temperature step decrease	Seth; Gupta (1995)
19	Temperature step decrease	Seth; Gupta (1995)
41	Temperature step decrease	Sangwai et al (2005)
41.2	Temperature step decrease	Sangwai et al (2005)
20	Temperature step increase	Seth; Gupta (1995)
21	Intermediate initiator and monomer feed	Seth; Gupta (1995)
22	Intermediate initiator and monomer feed	Seth; Gupta (1995)
29	Isothermal	Ray; Saraf; Gupta (1995)
30	Isothermal	Ray; Saraf; Gupta (1995)
31	Isothermal	Ray; Saraf; Gupta (1995)
34	Isothermal	Madruga; San Román; Benedi (1990)
35	Isothermal	Madruga; San Román; Benedi (1990)
36	Isothermal	Madruga; San Román; Benedi (1990)
37	Isothermal	Madruga; San Román; Benedi (1990)
32	Isothermal	Destruel; Taufer; Granel (1991)
33	Non-isothermal	(Armitage et al., 1988)

The experiments settings for each of these experimental data were also included in the experiments database (exp_database.m) in *Reactormodel*. A summary of the experiments selected for all analysis in this dissertation is provided with Table 5. The numbers in column “exp”, in Tables 4 and 5, refers to the number identifier of the experiment in *Reactormodel*, and these are used from here on to identify those in the discussion of the results.

Table 5: Experiments settings (stored in exp_database.m in *Reactormodel*).

Exp	T(°C)	Initiator	I0 (mol/L)	CTA	CTA(0) (mol/L)	Solvent	Fraction
2	50	AIBN	0.01548				
3	70	AIBN	0.01548				
4	90	AIBN	0.01548				
5	50	AIBN	0.02580				
6	70	AIBN	0.02580				
7	90	AIBN	0.02580				
25	50	AIBN	0.00061				
26.1	70	AIBN	0.00183				
16	70	AIBN	0.01548				
18	70->50	AIBN	0.02580				
19	70->50	AIBN	0.02580				
41	70->50	AIBN	0.01548				
41.2	70->50	AIBN	0.02580				
20	50->70	AIBN	0.02580				
21	50	AIBN	0.01548				
22	70	AIBN	0.01548				
29	70	BPO	0.04130				
30	70	BPO	0.04130			Benzene	0.2
31	70	BPO	0.04130			Benzene	0.4
34	70	AIBN	0.00304				
35	70	AIBN	0.00304	DDM	0.056		
36	70	AIBN	0.00304	DDM	0.108		
37	70	AIBN	0.00304	DDM	0.156		
32	60	AIBN	0.00188	GDMA	0.0159		
33	70, (>140)	AIBN	0.05700				

4.5. Model fitting

The polynomial equations to calculate the parameters θ_t, θ_p and θ_f were adjusted with Matlab’s function fminsearch, a nonlinear programming solver based on the simplex search method of Lagarias et al. (1998). The overlap factor γ was also included in the optimization. The objective function used in the optimization (minimization) is the normalized sum of squared errors:

$$Min\ E \equiv \sum_{i=1}^{N} \left(\frac{x_i^{exp} - x_i^{calc}}{x_i^{exp}} \right) \tag{66}$$

The procedure is similar to the one used by Ray, Saraf, and Gupta (1995), with the difference that, in this model, the functions were minimized only for conversion ($x_i = X_m$). Mn and Mw were found to be of minor importance for data fitting, and thus those were not included in the minimization objective function.

The experiments selected for the model adjustments are the original experiments from Marten and Hamielec (1979), with identifiers 2 to 7 in *Reactormodel*, and two experiments from Tefera, Weickert, and Westerterp (1997), with identifiers 25 and 26.1.

4.6. Criteria for models comparison

In order to compare the different models, which have a variable number of experimental data points, the relative normalized root mean square (RMS) method, which measures the average error between experimental and calculated values, was adopted in this dissertation:

$$RMS_{normalized} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \left(\frac{x_n^{exp} - x_n^{calc}}{x_n^{exp}} \right)^2} \tag{67}$$

4.7. Parameters for models AI01 and AI02

Tables 6, 7, and 8 provide a list of the relevant parameters to models AI01 and AI02. Parameters for models from the literature were used as provided in the original texts.

Table 6: Parameters used by AI01 and AI02

Parameter	Value / equation (*Reactormodel*)	Units	Source
ρ_{mon}	`(966.5 - 1.1*(T))/1000`	g.cm^{-3}	(1)
ρ_{pol}	`1.2`	g.cm^{-3}	(1)
ρ_{benzene}	`(844.18-1.07165*(T+273.15-323.1))/1000`	g.cm^{-3}	(1)
k_{p0}	`2.95e7*exp(-4353/(1.987*(T+273.15)))`	L.mol^{-1}.min^{-1}	(1)
k_{t0}	`5.88e9*exp(-701/(1.987*(T+273.15)))`	L.mol^{-1}.min^{-1}	(1)
k_{de}	`3.888e13*exp(-1.12*18253/(1.987*(T+273.15)))`	L. min^{-1}	(2)
$k_{d\ (\text{AIBN})}$	`6.32e16*exp(-15.43e3/(T+273.15))`	min^{-1}	(1)
$k_{d\ (\text{BPO})}$	`1.0140e+16*exp(-125.4/(8.314e-3*(T+273.15)))`	min^{-1}	(1)
k_{tc}/k_{td}	`3.956e-4*exp(4.09/(1.9858775e-3*(T+273.15)))`		(2)
k_{fm}/k_p	`9.48e3*exp(-13.88/(1.9858775e-3*(T+273.15)))`		(2)
$k_{fc(\text{GDMA})}/k_p$	`1.26`		(3)
$k_{fc(\text{DDM})}/k_p$	`7.5e-5*exp(25289/(8.314*(T+273.15)))`		(4)
$f_{cta(\text{GDMA})}$	`0.961`		(3)
$f_{cta(\text{DDM})}$	`1`		
$f_{0(\text{AIBN})}$	`0.58`		(1)
$f_{0(\text{BPO})}$	`0.5414-0.178974*(fs0)`		(1)

(1) (Ray; Saraf; Gupta, 1995); (2) (Louie; Carratt; Soong, 1985), (3) (Destruel; Taufer; Granel, 1991); (4) (Jahanzad et al, 1993)

Table 7: Specific parameters for AI01

Parameter	Value / equation (*Reactormodel*)	Units	Source
θ_p	`10^(8.2054e1-7.7342e4*(1/T) +1.8149e7* (1/(T^2)))`		Fitted
θ_t	`10^(1.2432e2-1.0282e5*(1/T)+2.2589e7* (1/(T^2)))`		Fitted
$\theta_{f(\text{AIBN})}$	`10^(1.9908e2-1.4883e5*(1/T)+2.6903e7* (1/(T^2)))`		Fitted
$\theta_{f(\text{BPO})}$	`10^(-3.763e1+1.686e4*(1/T))`		(2)
$\hat{V}^*_{mon}$	`8.22e-4`	m^3.kg^{-1}	(1)
$\hat{V}^*_{benzene}$	`9.01e-4`	m^3.kg^{-1}	(1)
$\hat{V}^*_{pol}$	`7.70e-4`	m^3.kg^{-1}	(1)
M_{jp}	`0.18781`	kg.mol^{-1}	(1)
V_{fmon}	`0.149+2.9e-4*T`		(1)
V_{fpol}	`T<105: 0.0194+1.3e-4*(T-105);` `T>=105: 0.0194+3e-4*(T-105);`		(1)
$V_{fbenzene}$	`0.025+1e-3*(T+273.15-171.1)`		(1)
γ	`0.9956`		Fitted

(1) (Achilias; Kiparissides, 1988); (2) (Ray; Saraf; Gupta, 1995)

Table 8: Specific parameters for AI02

Parameter	Value (*Reactormodel*)		Units	Source
k_t coefficients	`A11=-2.3085e-2`	`A12=8.0622e-1`		Fitted
	`A21=-2.3350e-1`	`A22=9.0358`		
	`A31=1.4003e-1`	`A32=-3.5822e1`		
	`A41=-1.7502e-3`	`A42=2.2250`		
k_p coefficients	`B11=-4.020e-3`	`B12=1.246e-1`		Fitted
	`B21=-3.054e-1`	`B22=9.734`		
	`B31=2.706e-1`	`B32=8.278`		
	`B41=2.493e-1`	`B42=-4.459e1`		

CHAPTER V

5. RESULTS AND DISCUSSION

In this chapter, the selected models (RSG, SG and SBGSG) and two new models, AI01, derived from RSG-SG, and AI02, derived from SBGSG, were analyzed. Models RSG, SG, and SBGSG were tested with the parameters as presented by the authors. All models were evaluated with the set of experiments described in Table 5. The summary of differences among models is shown in Table 9:

Table 9: Comparison of polymerization features for the tested models.

Polymerization features	CCS	RSG	SG	CB	SBGSG	AI01	AI02
Bulk polymerization	Y	Y	Y	Y	Y	Y	Y
Solution polymerization	N	Y	Y	Y	Y	Y	Y
Termination by combination	N	N	N	N	N	Y	Y
Radical transfer to monomer	N	N	N	N	Y	Y	Y
Efficiency factor f calculated	N	N	Y	N	N	Y*	Y
Depropagation at high temperature	N	N	N	N	N	Y	Y

Y= Yes, N=No

* See discussion of results on topic 5.1.

The evaluation of these models was performed comparing the model results with experimental data, taken from the literature, for different processes conditions: isothermal bulk polymerization, step feed of initiator, step changes in temperature, use of CTAs and non-isothermal polymerization.

The outputs analyzed in this comparison were Conversion, Mn and Mw, as discussed in Chapter 4. The comparison was performed in terms of the normalized root mean square error between experimental data and simulation results, calculated for each model according to equation (67).

5.1. Models AI01 and AI02

The new models AI01 and AI02 were adjusted (parameters fitting) with the function fminsearch from Matlab.

In the case of AI01, the optimized parameters were the coefficients of the diffusion parameters θ_t, θ_p and θ_f from the original model from Ray, Saraf, and Gupta (1995) and Seth and Gupta (1995), as in equations (11), (12) and (14).

The parameter γ was allowed to vary during this optimization, and the best fit was achieved with $\gamma = 0.9956$ (this parameter is found to be extremely sensitive).

One important remark in model AI01 is that the polynomial coefficients to calculate θ_t, as optimized by fminsearch, make this parameter diverge to a very high value, tending to infinite, even at high conversions. The net effect on equation (63) is $f = f_0$, which converts the SG-based model into an RSG-based model (constant f). It is not discarded that this can be due to a local minimum found during optimization, but the results evaluated here will be based on this setting. The model AI02 also was optimized with fminsearch by varying the coefficients from equations (64) and (65).

5.2. Conversion

The option "Multi-testing" of *Reactormodel* was used to run the experiments with the applicable models (CCS, RSG, SG, CB, SBGSG, AI01, and AI02) and the relevant set of ODE's (options with k_{fm}, k_{tc} and depropagation). The error results for conversion are summarized in Table 10:

Table 10: Average error results (relative root mean square) for conversion as calculated by the different models.

Exp	CCS	RSG	SG	AI01	CB	SBGSG	AI02	Experiment
2	7.00E-02	9.13E-02	1.44E-01	1.59E-01	5.38E-01	6.86E-02	1.09E-01	I0=0.01548 mol/L AIBN, T=50°C (1)
3	1.02E-01	1.75E-01	1.64E-01	1.38E-01	2.61E-01	1.07E-01	1.17E-01	I0=0.01548 mol/L AIBN, T=70°C (1)
4	9.92E-02	1.34E-01	1.06E-01	1.83E-01		1.67E-01	2.66E-01	I0=0.01548 mol/L AIBN, T=90°C (1)
5	7.27E-02	2.60E-01	2.22E-01	2.14E-01	2.32E-01	1.83E-01	1.17E-01	I0=0.0258 mol/L AIBN, T=50°C (1)
6	4.57E-02	1.52E-01	1.29E-01	9.15E-02	7.48E-02	2.03E-01	1.46E-01	I0=0.0258 mol/L AIBN, T=70°C (1)
7	8.20E-02	1.25E-01	8.21E-02	1.08E-01		1.46E-01	1.69E-01	I0=0.0258 mol/L AIBN, T=90°C (1)
25		9.76E-01	9.26E-01	9.32E-01		2.09E-01	2.57E-01	I0=0.1% AIBN, T=50°C (2)
26.1		1.28E-01	1.18E-01	7.03E-02		1.40E-01	6.12E-02	I0=0.3% AIBN, T=70°C (2)
16		2.95E-01	3.71E-01	5.37E-01		4.42E-01	3.88E-01	I0=0.01548 AIBN, T=70, IA(initiator) (3)
18		7.87E-02	1.91E-01	2.10E-01		9.56E-01	8.68E-01	SD1 (4)
19		1.93E-01	1.68E-01	1.62E-01		5.47E-01	5.10E-01	SD2 (4)
41		2.18E-01	2.04E-01	2.03E-01		1.02E-01	1.28E-01	SD70(20)50 - 0.01548 mol/L AIBN(5)
41.2		2.80E-01	2.61E-01	2.59E-01		7.13E-02	1.21E-01	SD70(20)50 - 0.0258 mol/L AIBN (5)
20		3.00E-01	2.96E-01	3.00E-01		3.64E-01	3.72E-01	SI (4)
21		6.30E-01	5.95E-01	6.41E-01		1.66E-01	2.04E-01	IA50a2 (4)
22		2.93E-01	6.83E-01	6.80E-01		6.60E-01	5.78E-01	IA70a2 (4)
29		3.87E-01	1.96E-01	1.66E-01		1.73E-01	8.52E-02	fs0=0, I0=0.0413 mol/L BPO, T=70°C (3)
30		1.93E-01	2.49E-01	2.58E-01		2.65E-01	1.90E-01	fs0=0.2, I0=0.0413 mol/L BPO, T=70°C (3)
31		8.99E-02	1.17E-01	1.49E-01		2.40E-01	1.98E-01	fs0=0.4, I0=0.0413 mol/L BPO, T=70°C (3)
34		3.33E-01	3.09E-01	2.92E-01		2.26E-01	2.71E-01	0.5% AIBN, T=70°C (6)
35		2.74E-01	2.55E-01	2.55E-01		3.55E-01	2.75E-01	DDM=5.6e-2 mol/L, 0.5% AIBN, T=70°C (6)
36		2.29E-01	2.29E-01	2.29E-01		5.09E-01	4.66E-01	DDM=10.8e-2 mol/L, 0.5% AIBN, T=70°C (6)
37		2.15E-01	2.37E-01	2.37E-01		5.84E-01	5.53E-01	DDM=15.6e-2 mol/L, 0.5% AIBN, T=70°C (6)
32		1.69E-01	1.64E-01	1.64E-01		9.93E-01	8.82E-01	GDMA=0.0159 mol/L, AIBN=0.00188 mol/L, T=60°C (7)
33		2.81E-01	3.24E-01	3.29E-01		5.38E-01	4.26E-01	Non-Isothermal (8)

Experiment sources: 1- Marten and Hamielec (1979); 2- Tefera; Weickert; Westerterp (1997); 3- Ray; Saraf; Gupta (1995); 4-Seth; Gupta (1995); 5-Sangwai et al (2005); 6- Madruga; San Román; Benedi (1990); 7- Destruel; Taufer; Granel (1991); 8- Armitage et al. (1988).

The models CCS and CB were only tested with a limited set of experiments because the parameters required for those different conditions are not available, and since the objective of this study is to develop a generalized model with continuous functions, no additional efforts were taken to calculate those missing parameters.

Models RSG and SG were tested with the best-fit correlation (BFC) curves provided by the respective authors. The parameters used in these models are in line with the ones used by the authors, including the simplifications pointed out in Table 2.

The results in Table 10 confirm the very good fit of the original CCS model. Such degree of agreement between experimental and calculated data could not be achieved by the models CB, RSG, SG, SBGSG, AI01, and AI02, which, except for the first, are generalized models with continuous functions.

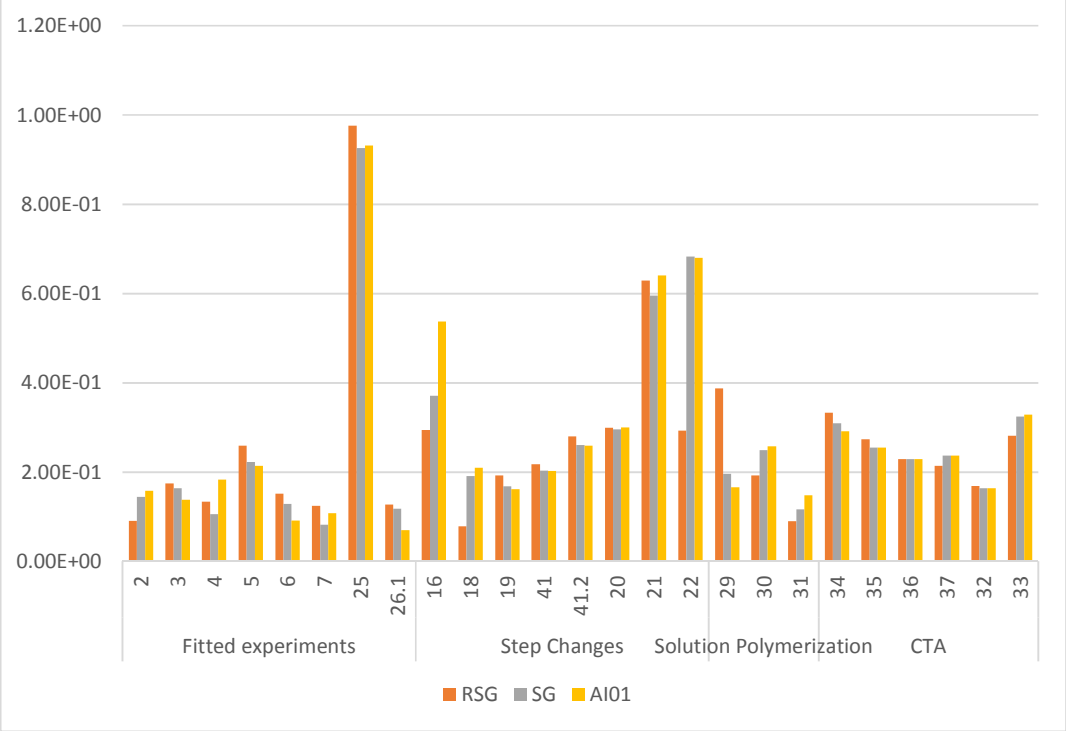


Figure 23: Comparison of the normalized root mean square errors for conversion – RSG, SG and AI01.

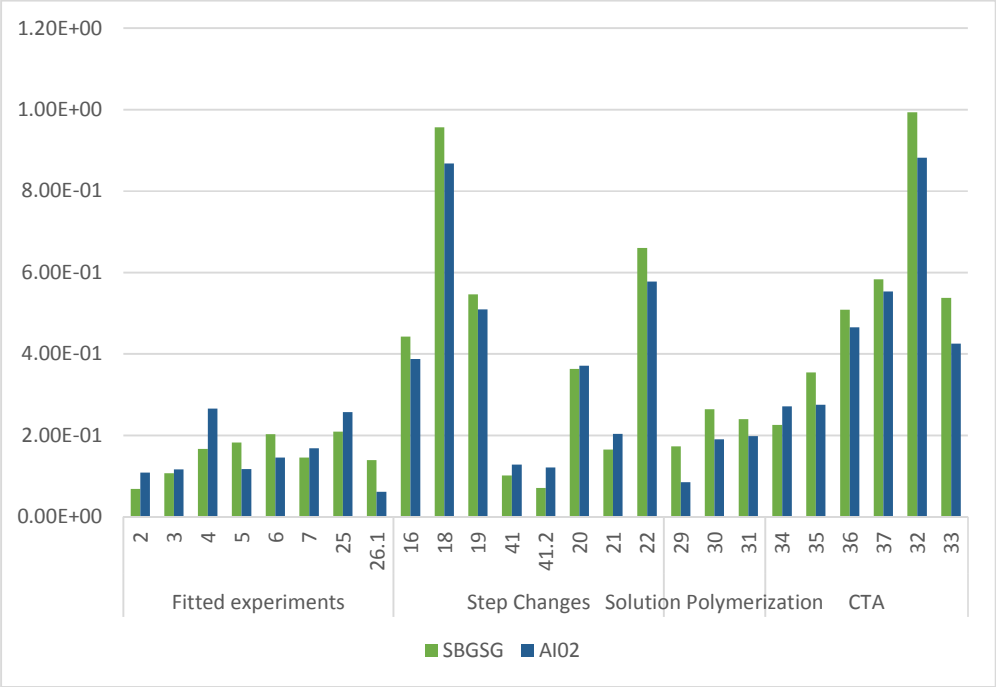


Figure 24: Comparison of the normalized root mean square errors for conversion – SBGSG and AI02.

The experiments 18 (SD1) and 19 (SD2), discussed on topic 2.4.9, are included in Table 10 and Figures 23 and 24, but it has been purged from Figure 25 as to compute the correct cumulative errors (all based on validated data).

The experiment AI01 presented slightly larger errors as compared with RSG and SG. In the case of AI02, it showed better overall results than SBGSG model. Hence, we can conclude that both experiments deliver similar results as compared with the existing models while delivering additional features, namely the inclusion of termination by combination, depropagation, and radical transfer to monomer mechanisms.

5.2.1. Model-fitting experiments

The models AI01 and AI02 were tested with the experiments used to fit the parameters (experiments 2, 3, 4, 5, 6, 7, 25 and 26.1). Models SG and SBGSG are selected here as the references for the comparison plots:

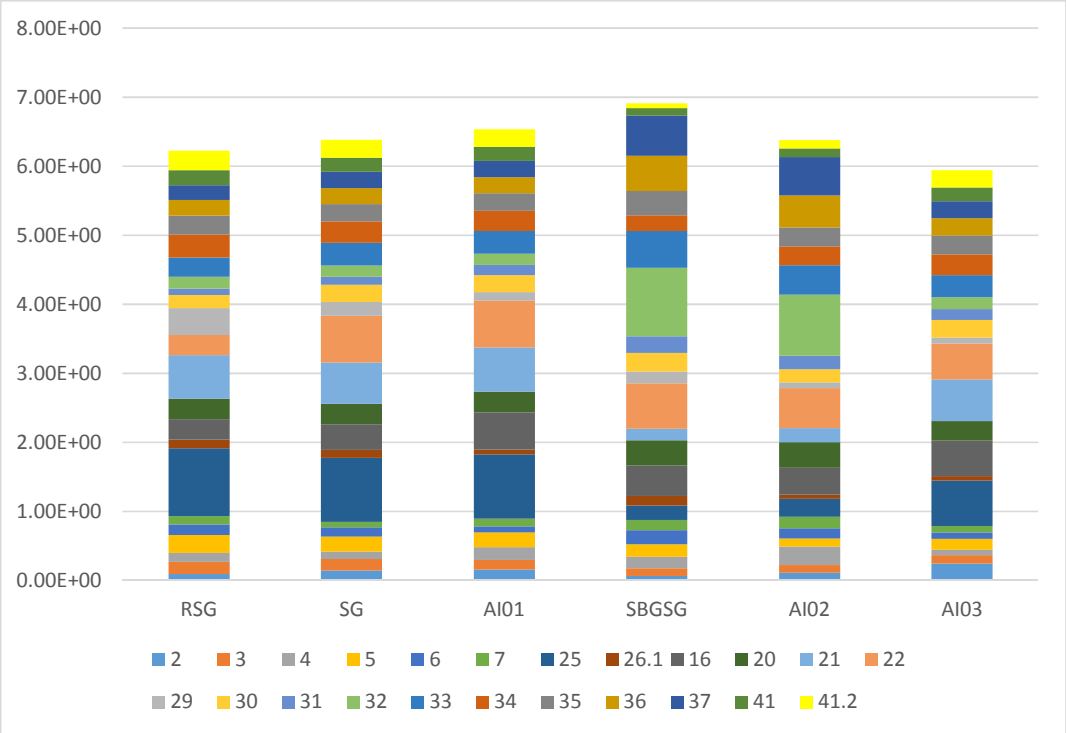


Figure 25: Cumulative normalized root mean square errors of conversion, considering all experiments except experiment 18 (SD1) and 19 (SD2).

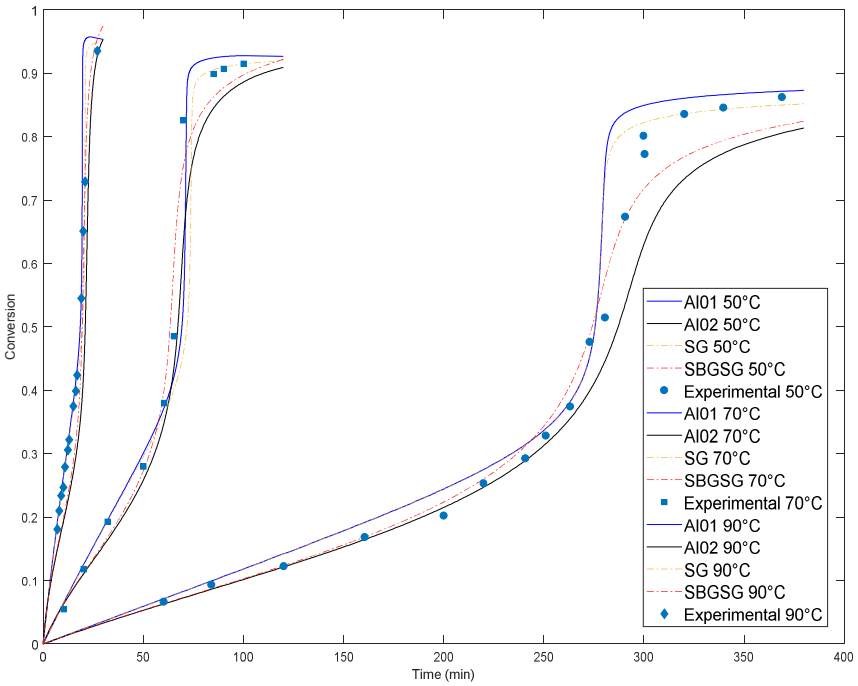


Figure 26: Models prediction - AIBN=0.01548 mol/L, isothermal process.

The error magnitude reported on Table 10 and Figures 23 and 24 can be visually inspected in Figure 26. In this particular set of experiments, the models AI01 and SG calculated higher values for the final conversion, as compared with the models AI02 and SBGSG. In some cases, the model AI02 outperformed the model SBGSG regarding predicting the gel effect (Figure 27):

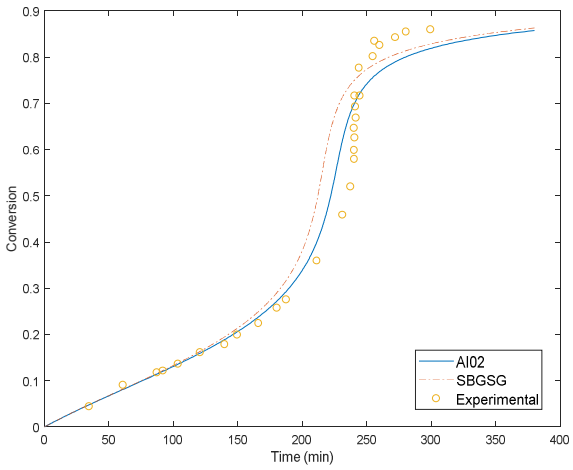


Figure 27: Experiment 5 (AIBN=0.0258 mol/L, T=50°C, isothermal process).

A particular experiment, 25, could not be well predicted from any of the models (Figure 28). It is not clear whether this is due to an issue with the models not being able to capture the behavior of gel effect at very low concentrations of initiator, or if an experimental error could be present in the data.

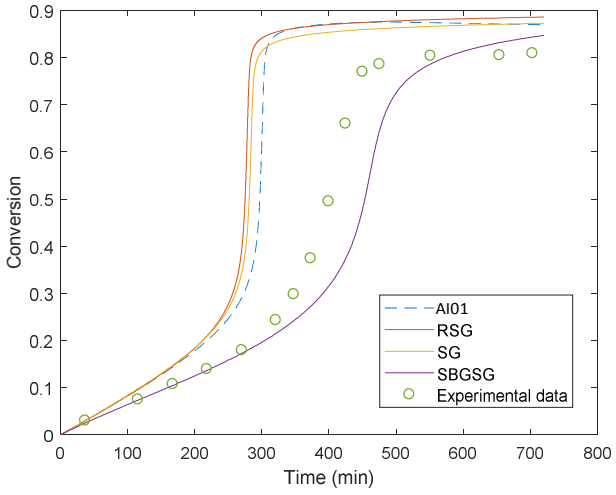
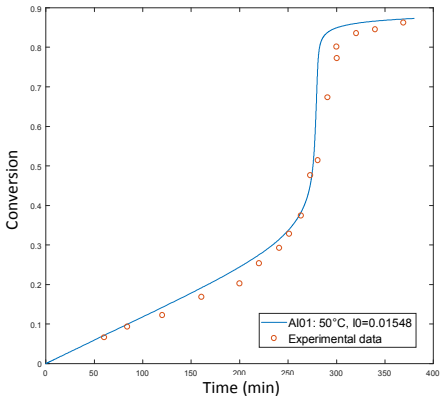
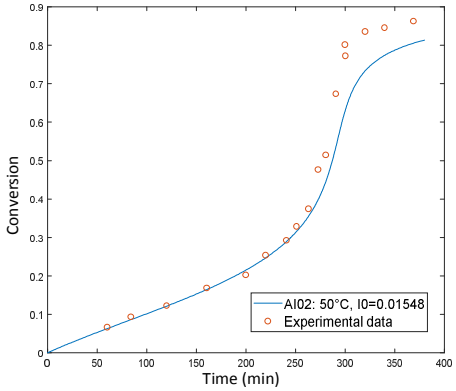
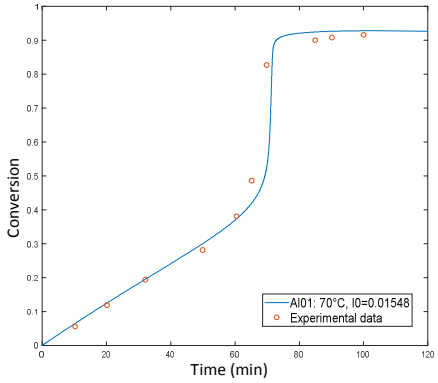
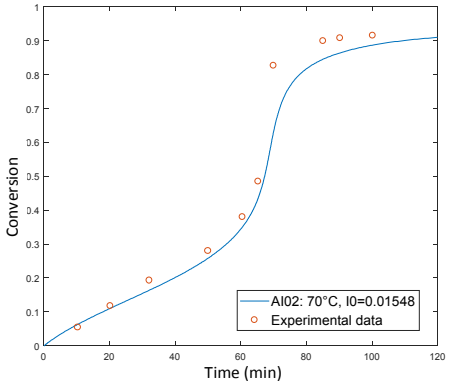
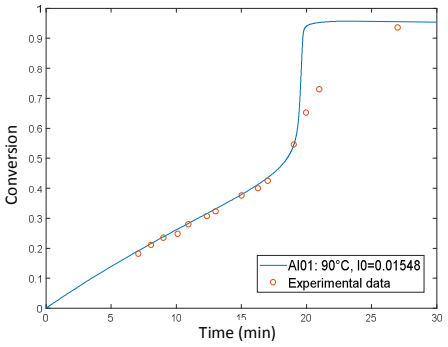
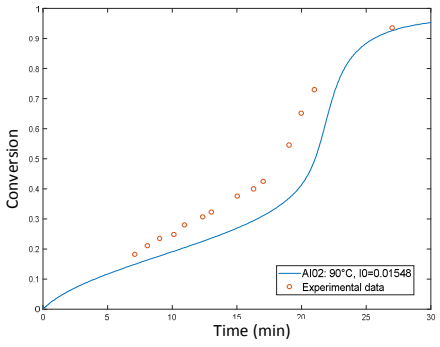
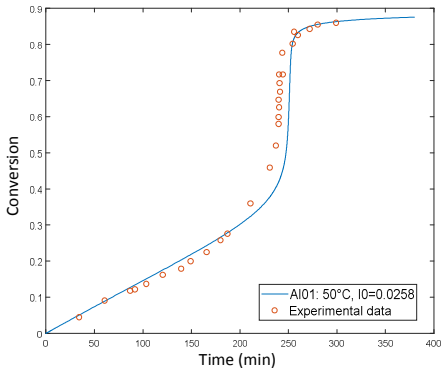
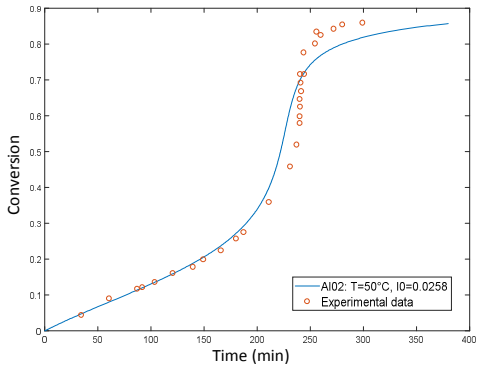


Figure 28: Experiment 25 - bulk polymerization of MMA at 50°C with 0.1% AIBN (Tefera; Weickert; Westerterp, 1997), tested with different models.

The individual prediction of models AI01 and AI02 are shown in Figure 29:

	AI01	AI02
Experiment 2		
Experiment 3		
Experiment 4		
Experiment 5		

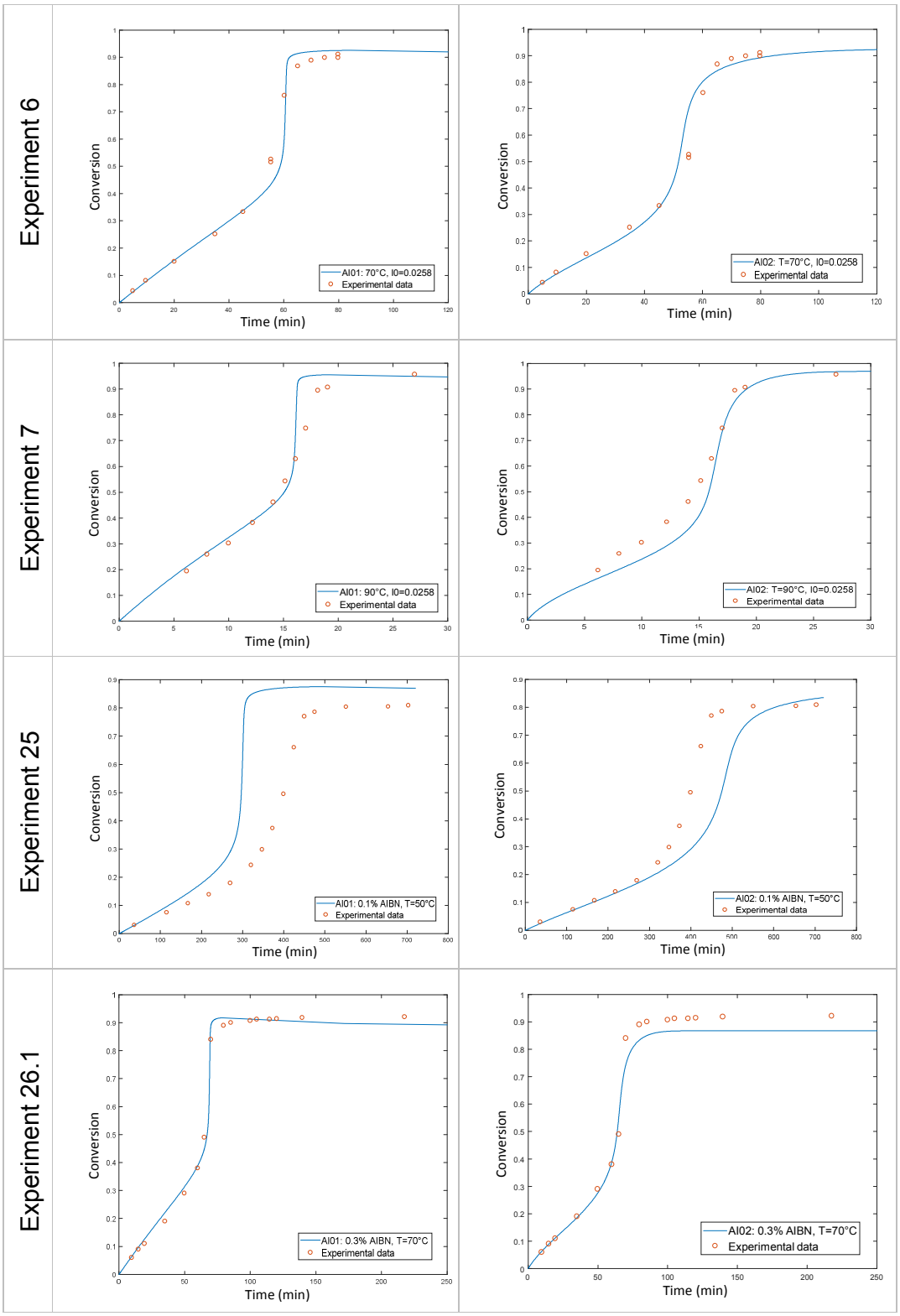


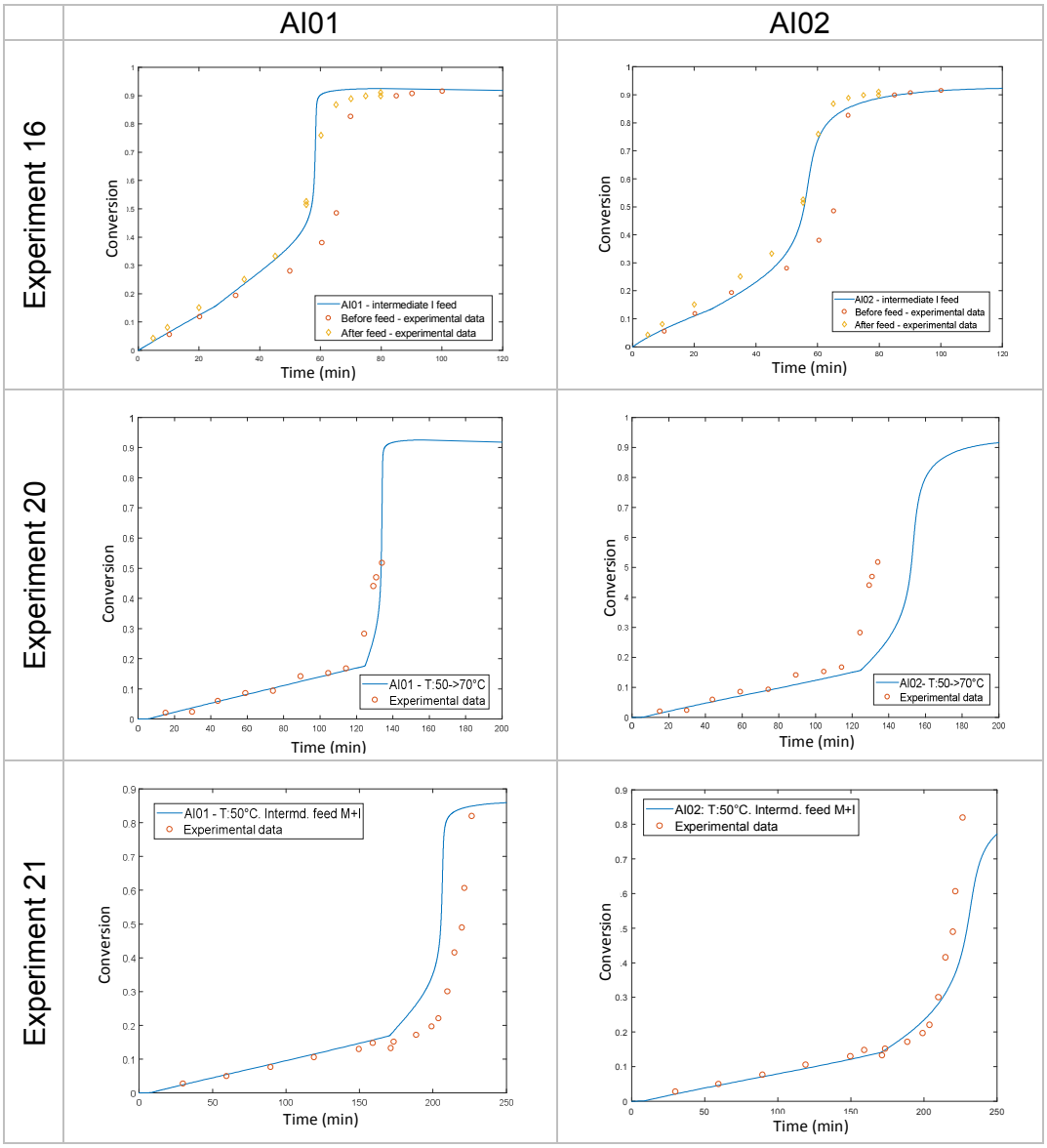
Figure 29: Prediction of conversion for experiments used in data fitting, models AI01 and AI02.

Overall, the results of models AI01 and AI02 showed good agreement with the experimental data. The exception is the experiment 25, bulk polymerization

of MMA at 50°C with 0.1% AIBN (Tefera; Weickert; Westerterp, 1997), as previously discussed. There is a trend of model AI01 having its onset for gel effect at an earlier time as compared with its counterpart AI02 for the same experiment.

5.2.2. Step changes in process

The experiments with identifiers 16, 18, 19, 20, 21, 22, 41, and 41.2 are related to step changes in the process, testing the behavior of non-steady state conditions in the reactor. Experiments 18 and 19, as discussed previously on topic 2.4.9, were removed from the tests (replaced by experiments 41 and 41.2). Results are shown in Figure 30:



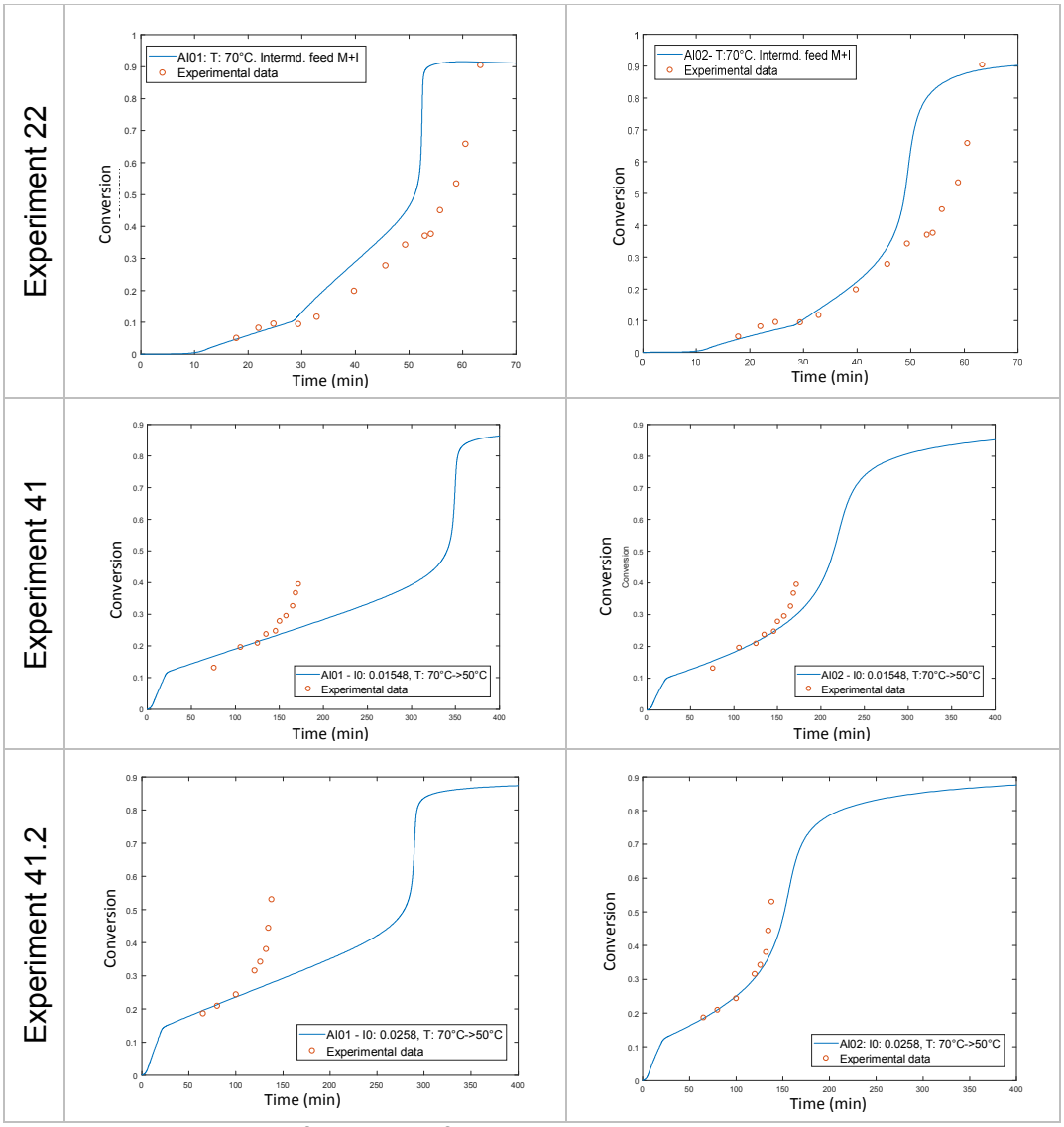


Figure 30: Prediction of conversion for step changes in process, models AI01 and AI02.

With the exception of the experiments 41 and 41.2 with AI01, simulating the step decrease from 70°C to 50°C (discussed on topic 2.4.9), the results provided reasonable adjustment with the experimental data. The trend of early gel effect for AI01, as compared with AI02, is also confirmed in these experiments.

5.2.3. Solution polymerization

The experiments in this section compare the bulk polymerization (f_{s0}=0, experiment 29) with two solution polymerizations (f_{s0}=0.2 and f_{s0}=0.4, experiments 30 and 31 respectively). Results are shown in Figure 31:

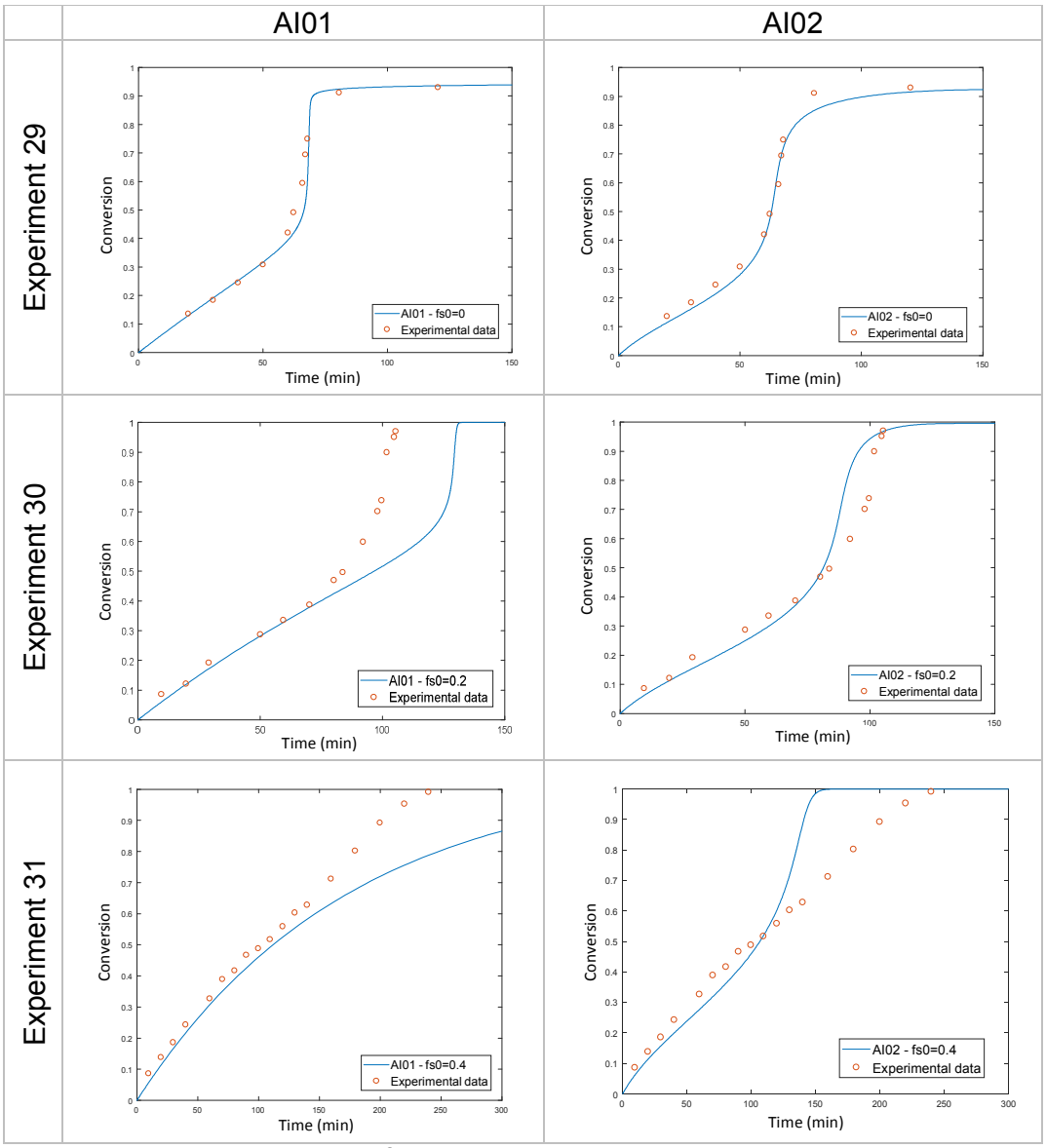


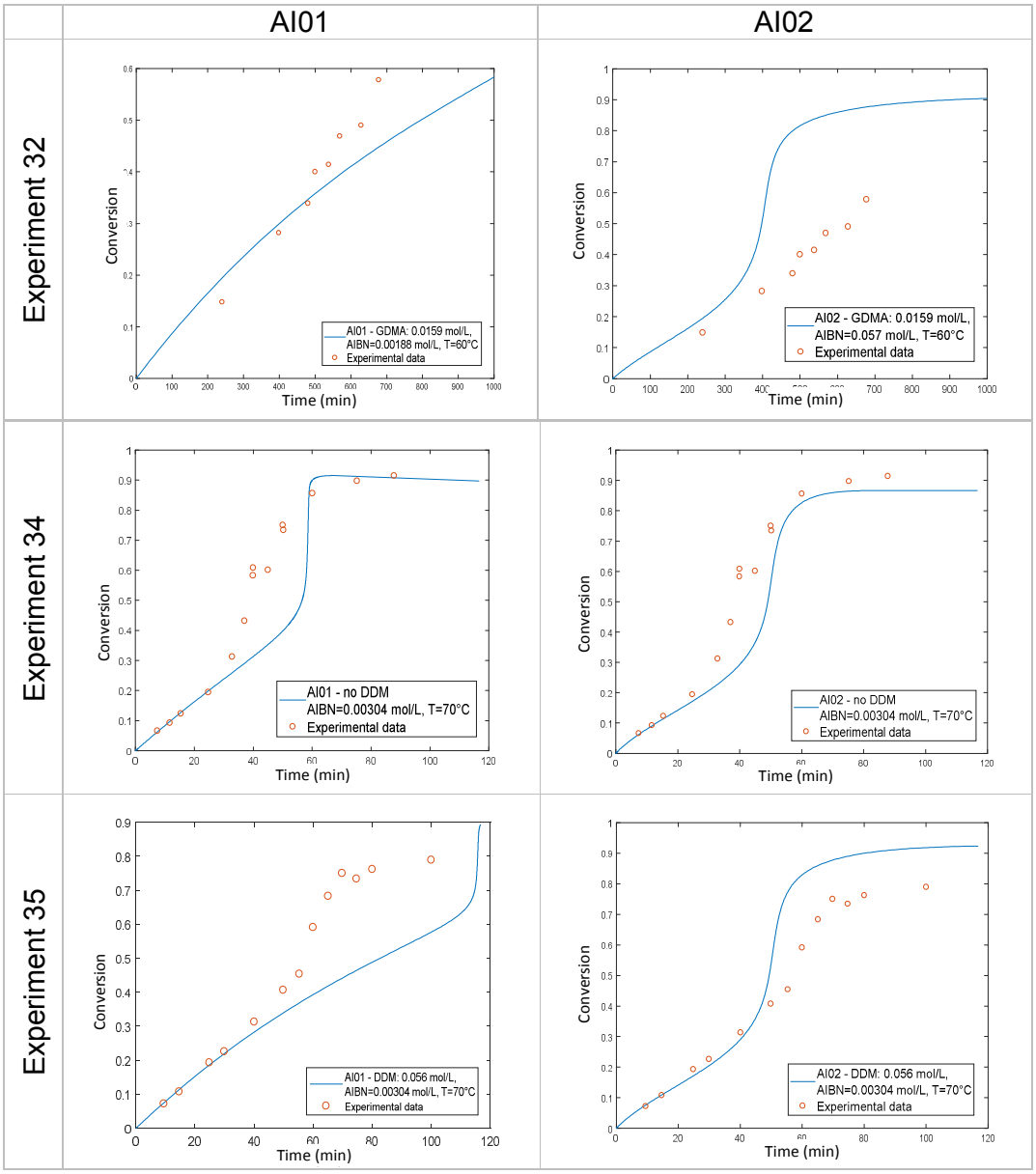
Figure 31: Prediction of solution polymerization, models AI01 and AI02.

The results for solution polymerization showed some significant divergence, when comparing the models AI01 and AI02. The errors observed in model AI01 are similar to the same cases of the original SG model (Seth; Gupta, (1995). This might indicate that this model requires improvement for solution polymerization, but it can be used to predict polymerizations at low solvent fractions (fs0<0.4).

The model AI02 showed better results when tested with fs0=0.2, but started to diverge to shorter polymerization times at fs0=0.4. Since the original parameters were not adjusted considering this polymerization scenario, it remains as a possibility to do so.

5.2.4. Use of CTA's

The fit of models AI01 and AI02 with experiments using CTAs in the polymerization of MMA are analyzed in Figure 32. These experiments are related to the use of glycol dimercapto acetate (GDMA), experiment 32, and dodecyl mercaptan (DDM), experiments from 34 to 37:



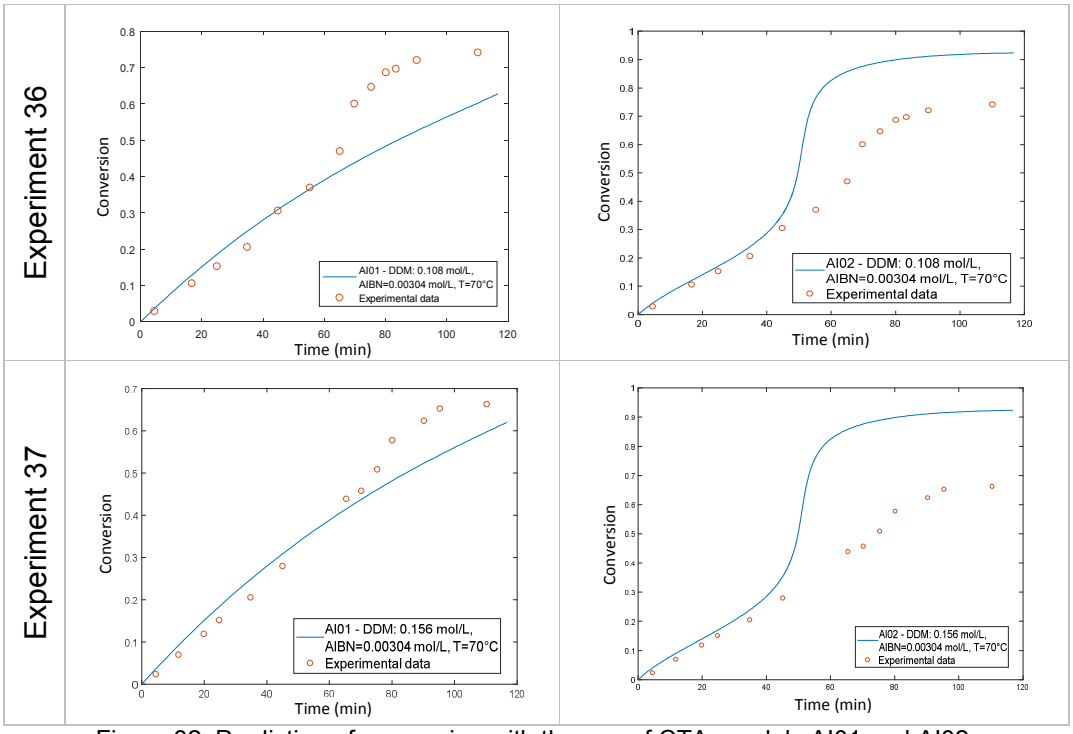


Figure 32: Prediction of conversion with the use of CTA, models AI01 and AI02.

Although the average errors are low, in comparison with other experiments, the visual inspection of the plots reveals a poor prediction of the onset of the gel effect, suggested by the comparison with the experimental data (Figure 33). The errors for SG and SBGSG are about the same as AI01 and AI02, respectively, since the latter set of models are derived from the previous ones.

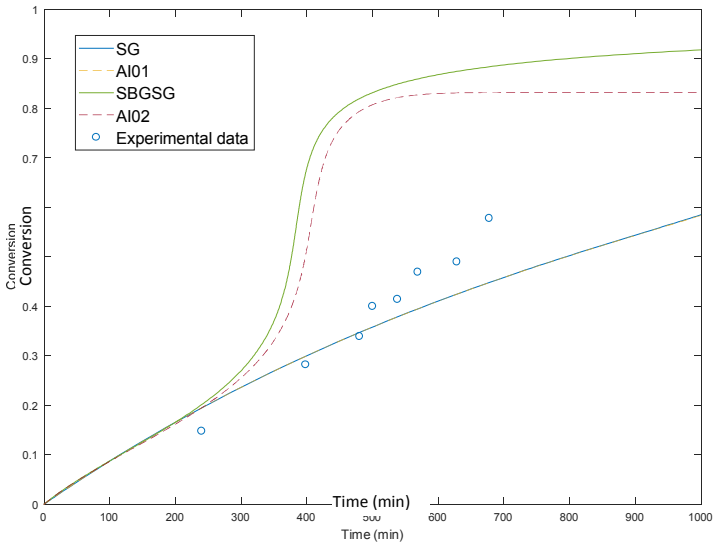


Figure 33: Experiment 32 - bulk polymerization of MMA at 60°C with AIBN 0.0258 mol/L and 0.00188 mol/L GDMA (CTA), tested with different models.

Both models failed to capture the behavior of CTAs in the conversion. The only models which can provide a weak qualitative indication are SG and AI01 because both have k_t calculated as a function of molecular weight (Mn), as shown in Figure 34. Even though the onset for gel effect is underestimated, with the models SG and AI01 predicting a longer time for this onset, the model SBGSG, and its derived AI02, are both only functions of conversion. Because of this limitation, these models are unable to capture the nuances due to the presence of CTAs decreasing the molecular weight.

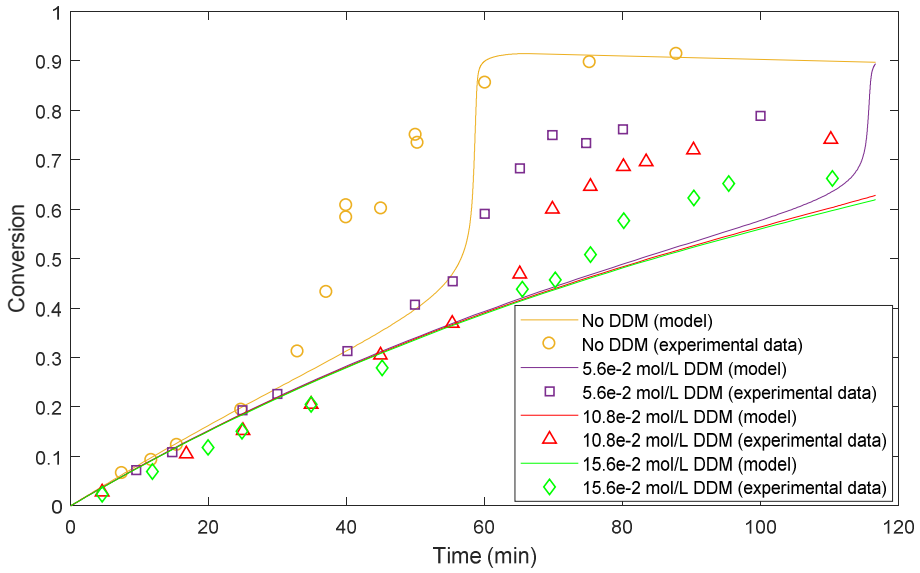


Figure 34: Model AI01 reproducing tests with DDM as CTA (Madruga; San Román; Benedi, 1990).

5.2.5. Non-isothermal process

The last evaluation is the conversion as a function of temperature. Experiment 33 was reproduced by using a function to model the temperature as a function of time (Figure 35), as reported by Armitage et al. (1988).

The result seen in Figure 36 demonstrates that the previous models (SG, RSG, and SBGSG) overestimate the final conversion, as the depropagation reaction is not included in the models. Models AI01 and AI02 present good results. It is pointed out that, in this dissertation, the activation energy for propagation was assumed with a correction factor of +12% in order to achieve

the best fit with the data. When the correction is not applied, the final conversion is slightly below the experimental values.

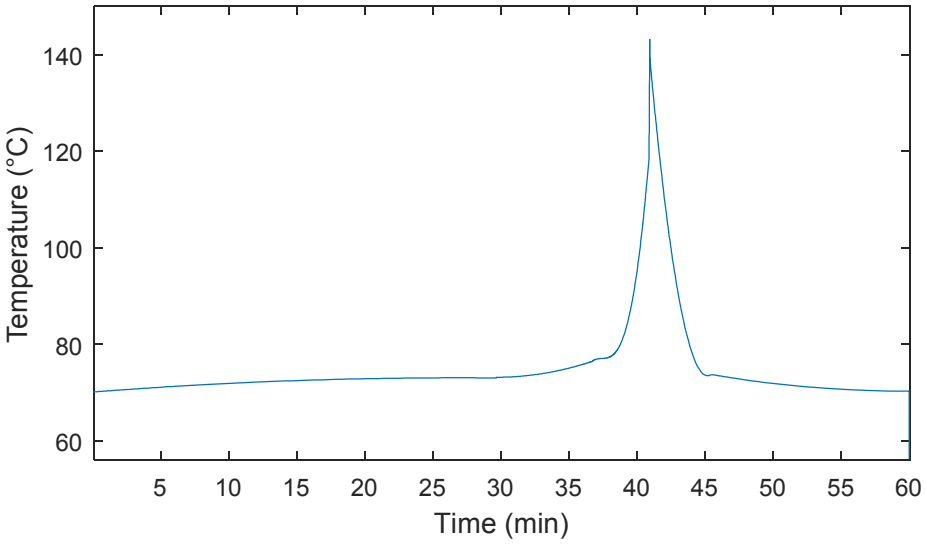


Figure 35: Temperature profile for experiment 33 (Armitage et al., 1988).

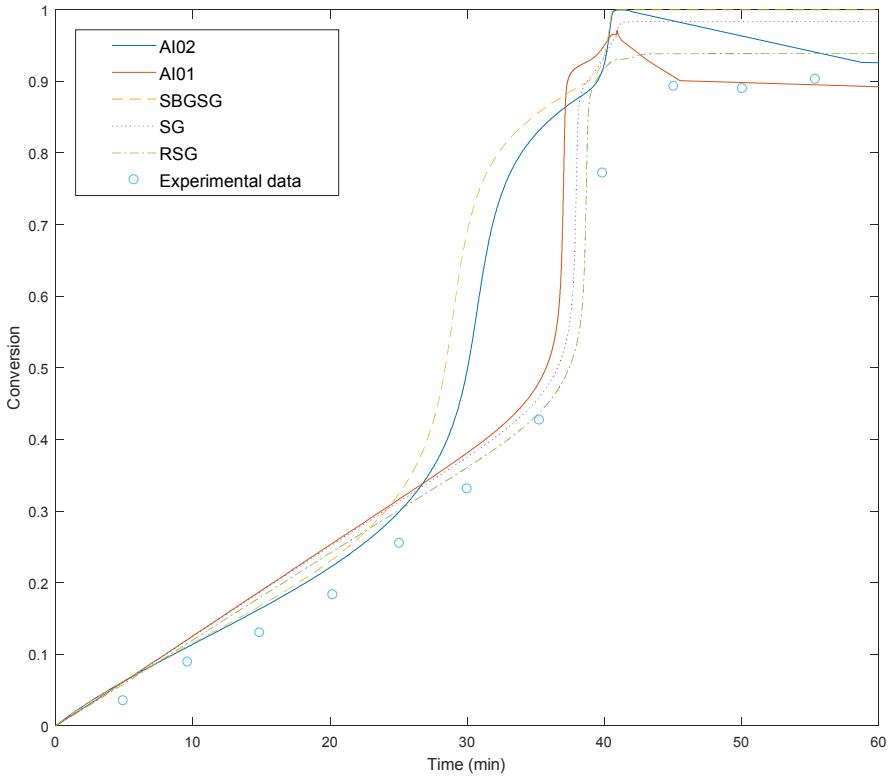


Figure 36: Experiment 33 (Armitage et al., 1988) tested with different models.

The behavior of conversion decreasing after reaching the maximum is due to the depropagation reaction. This decrease will stabilize at the equilibrium concentration of the monomer $[M]_e$ at the reaction mixture temperature. The peak of conversion, with subsequent retraction to a lower equilibrium value, could not be confirmed with the present data, because it occurs in between two of the experimental values. This phenomenon could be the object of investigation in future studies.

5.2.6. Error trend in AI01 and AI02:

One underlying conclusion that can be taken from the pictures shown in previous topics, which could not be observed on the error data (the module of the error), is the fact that models AI01 and AI02 are, in most cases, orbiting the experimental data in opposite sides. For example, when AI01 has a positive error, AI02 may have a negative error. This trend can be further explored in future models.

5.3. Molecular weight (Mn, Mw)

The molecular weight is determined by Mn and Mw, as well as the polydispersity index. The number of experimental data for these properties is much more limited as compared with the availability of conversion data. Nonetheless, a set of representative molecular weight data is evaluated here. Tables 11 and 12 display the results for Mn and Mw, respectively.

Table 11: Average error results (normalized sum of squares) of Mn in different models.

	CCS	RSG	SG	AI01	CB	SBGSG	AI02	Experiment
3	3.20E-02	7.78E-02	8.61E-02	5.42E-02	1.93E-01	1.00E-01	7.77E-02	CCS I0=0.01548, T=70°C (1)
4	9.62E-02	1.27E-01	1.69E-01	8.60E-02		2.93E-01	2.85E-01	CCS I0=0.01548, T=90°C (1)
6	8.78E-02	1.78E-01	1.93E-01	1.30E-01	2.41E-01	1.59E-01	1.59E-01	CCS I0=0.0258, T=70°C (1)
7	1.24E-01	1.63E-01	1.56E-01	1.43E-01		2.14E-01	2.09E-01	CCS I0=0.0258, T=90°C (1)
26.1		2.46E-01	2.22E-01	2.27E-01		2.23E-01	2.32E-01	TFF I0=0.0172 T=70°C (2)

Table 12: Average error results (normalized sum of squares) of Mw in different models.

	CCS	RSG	SG	AI01	CB	SBGSG	AI02	Experiment
3	2.56E-01	2.77E-01	2.86E-01	2.72E-01	3.54E-01	2.46E-01	2.55E-01	CCS I0=0.01548, T=70°C (1)
4	2.75E-01	2.25E-01	2.56E-01	2.88E-01		4.29E-01	4.25E-01	CCS I0=0.01548, T=90°C (1)
6	2.31E-01	2.90E-01	3.06E-01	2.37E-01	3.49E-01	2.99E-01	3.04E-01	CCS I0=0.0258, T=70°C (1)
7	3.01E-01	2.88E-01	3.04E-01	3.09E-01		3.43E-01	3.41E-01	CCS I0=0.0258, T=90°C (1)
26.1		2.06E-01	2.26E-01	1.64E-01		2.94E-01	2.49E-01	TFF I0=0.0172 T=70°C (2)

It can be observed that the errors for molecular weight were found to be lower in the new models AI01 and AI02 when compared with their respective parent models. These results demonstrate that the inclusion of k_{tc} can, in fact, improve the prediction of molecular weight properties of the polymer. Figure 37 shows the comparative differences for each model.

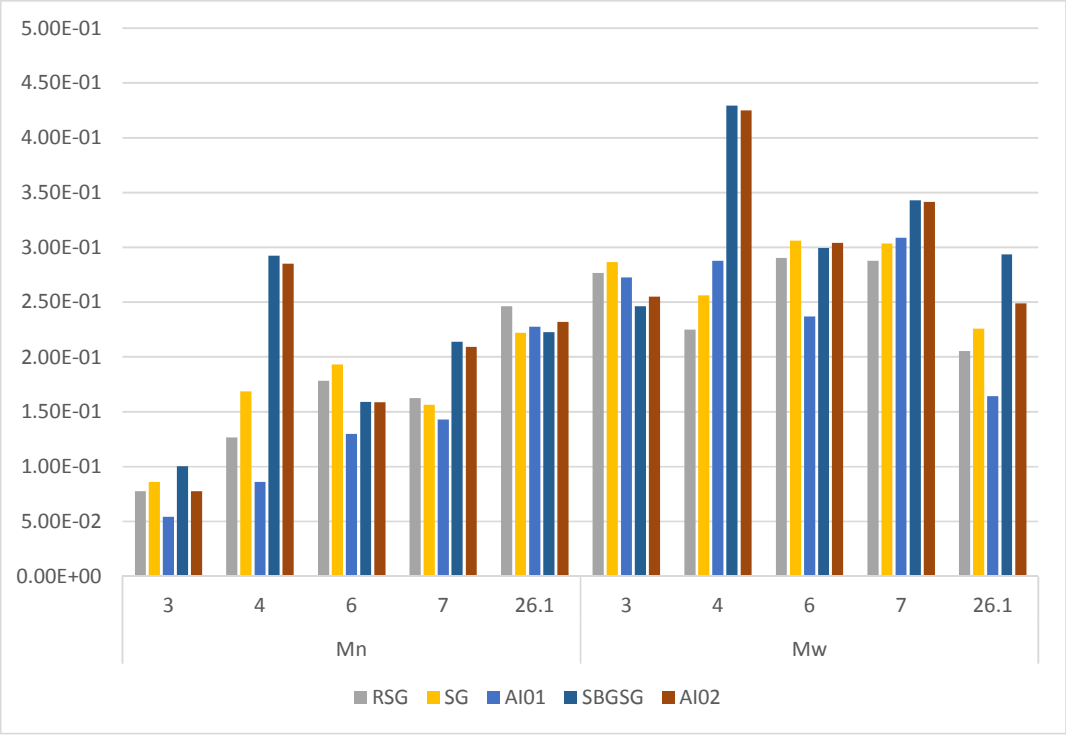


Figure 37: Comparison of the normalized root mean square errors for Mn and Mw, calculated with the different models.

The results obtained with the RSG-based models (RSG, SG, and AI02) presented lower errors as compared with the SBGSG-based models (SBGSG and AI02). As the data in the literature was not organized in a way where the pairs of Mn and Mw could be undoubtedly defined, it was not possible, in this study, to determine the fit of the models with the experimental polydispersity index.

5.4. Exploring the results – hybrid model

As indicated previously, a trend of opposite errors was observed with the models AI01 and AI02 (Figure 38). This trend indicates that there is a possibility

that the polymerization process can be better modeled if a hybrid model, considering the best prediction regions (early and later stages of polymerization) for each of the models.

Such hybridization of the calculation of k_p and k_t can be modeled with a simple weighing equation:

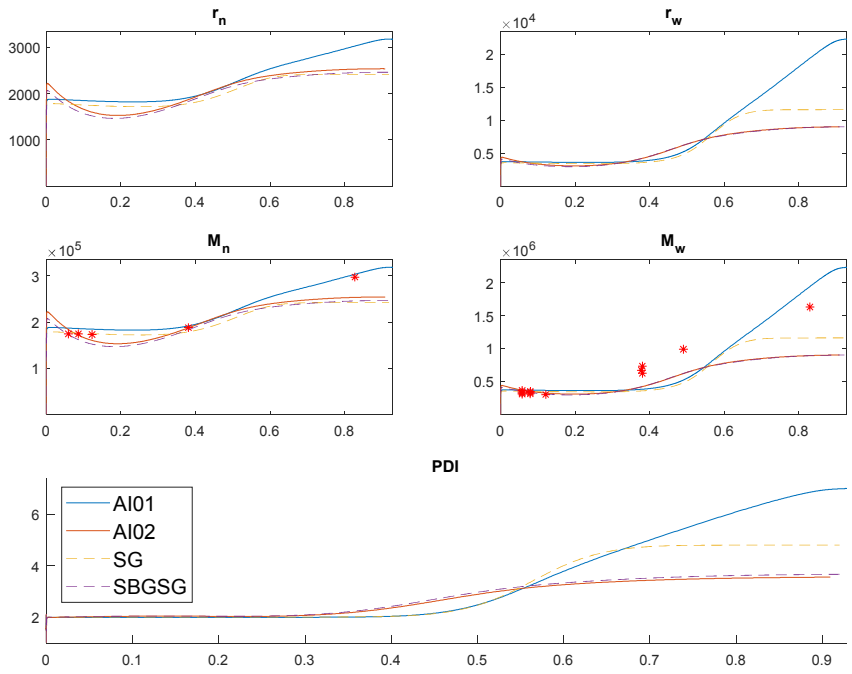


Figure 38: Mn and Mw prediction with the different models, experiment 3.

$$k_t = k_t(AI01) \times \left(1 - {X_m}^{\omega 1}\right) + k_t(AI02) \times {X_m}^{\omega 1} \quad (68)$$

$$k_p = k_p(AI01) \times \left(1 - {X_m}^{\omega 2}\right) + k_p(AI02) \times {X_m}^{\omega 2} \quad (69)$$

In these equations, $k_{p,t}(AI01, AI02)$, are the kinetic pseudo-constants as calculated in the models AI01 and AI02, as previously discussed, and $\omega 1$ and $\omega 2$ are the hybridization parameters (which determine the fading degree of the first in favor of the second). For the current experiments, AI01 and AI02, applied to bulk polymerization only, the optimal hybridization parameters, determined with Matlab's fminsearch fitting with experiments 2 to 7, 25 and 26.1, are $\omega 1 = 0.9348$ and $\omega 2 = 1.0707$. This kinetic model, named AI03, was used with the same

equation system as in AI01 and AI02, and the improvement can be seen in Figure 39:

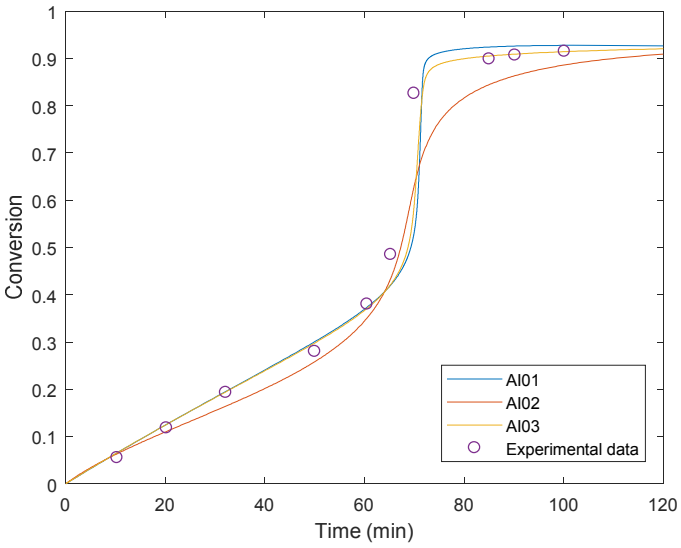


Figure 39: Experiment 03 – improvement with hybrid model AI03.

One of the benefits of the hybridization is the reduction of the stiffness of the gel effect as calculated by the RSG-derived models. Since the use of a hybrid model involves the evaluation of the multiples scenarios, as studied here for the simple models, and some of the parameters need to be better understood (for example, dependencies of ω factors with temperature), this model is not explored here (but this can be an idea for future investigations).

5.5. Process Sensitivity Analysis

The sensitivities of the models AI01 and AI02 were analyzed with Minitab version 16.1.1., with the contour plot functionality, using the interpolation method of distance with power 2. The data was generated with *Reactormodel*, considering the levels described in Table 13:

Table 13: Levels of experiments used to generate the contour plots for sensitivity analysis.

Variable	Levels
T (°C)	50, 100, 110, 140
I_0 (mol/L)	0.01, 0.03
f_{s0}	0, 0.4
Conversion	0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8

The results are as discussed in the next sessions.

5.5.1. Bulk polymerization

As expected, k_p remains constant over the course of the reaction until high conversion levels are achieved (Figure 40). The behavior is pronounced in temperatures below Tg (105°C). The trace of a similar behavior in temperatures above Tg may be explained by the fact that the efficiency factor of the initiator was lumped together with k_p when the model converged to $\theta_f \rightarrow \infty$.

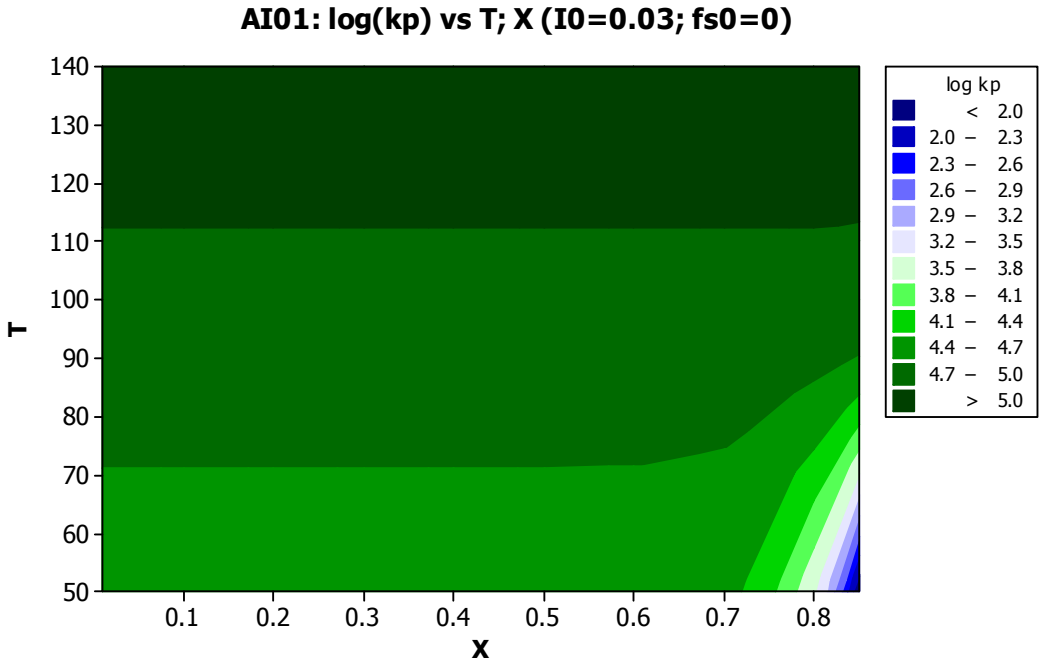


Figure 40: log(kp) as a function of temperature and conversion (AI01).

As k_p is calculated in AI02 with equation (65), which does not model the transition of glassy to non-glassy regions, the k_p will diverge to low values (Figure 41). As a result, the conversion is limited to approximately 25% when the temperature of polymerization is set to 140°C (I_0=0.03, f_{s0}=0). This result demonstrates that the model AI01 (and its parent model, SBGSG) cannot be used in processes with the temperature ranges above Tg.

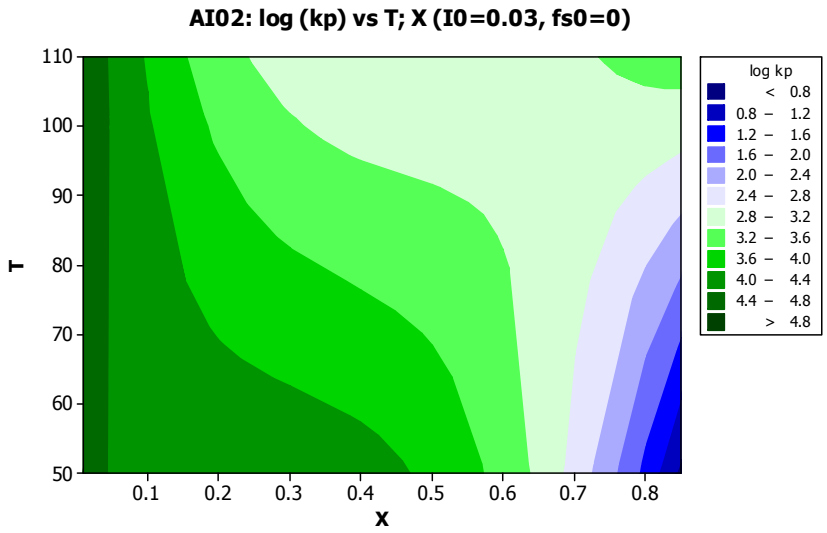


Figure 41: log(kp) as a function of temperature and conversion (AI02).

Figure 42 also shows, for AI01, the k_t behavior in the two different zones – above and below Tg. Termination is favored in the low-temperature range and when at low Tg reaction medium, which favors the increase of average molecular weight, and thus, decreases the diffusion of polymer chains.

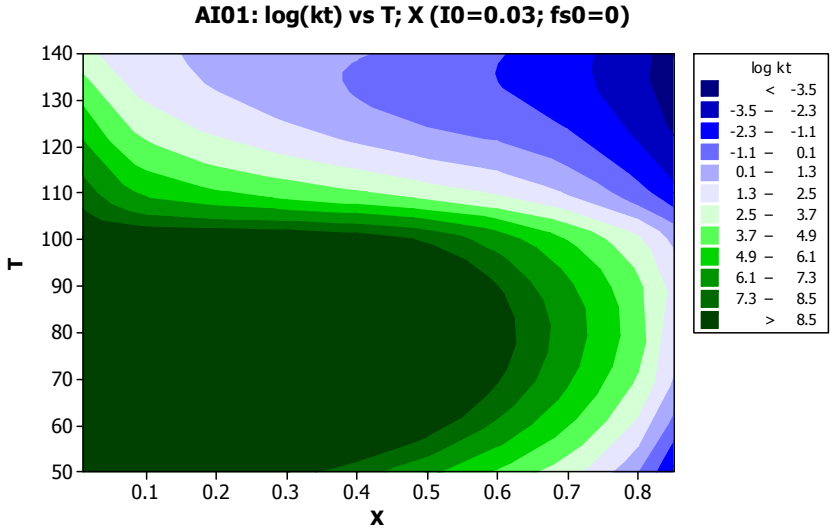


Figure 42: log(kt) as a function of temperature and conversion (AI01).

A linear behavior is observed for k_t with AI02 model (Figure 43):

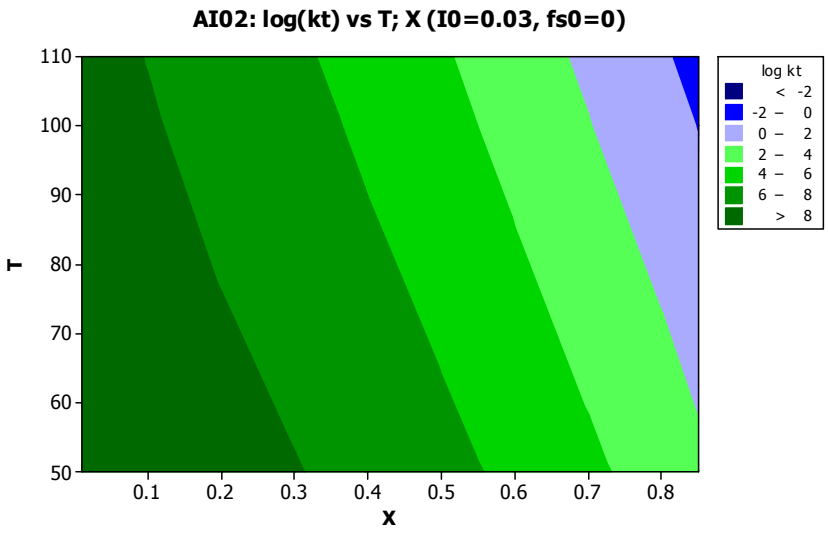


Figure 43: log(kt) as function of temperature and conversion (AI02).

In Figure 44, one can observe the behavior of the depropagation reaction. In the low-temperature range, at high conversion, it is observable that the ratio k_{de}/k_p increases considerably. This behavior in this region is due to the decreasing k_p because of the glassy medium. Such a condition does not impose a restriction to the opposite reaction, which is depropagation.

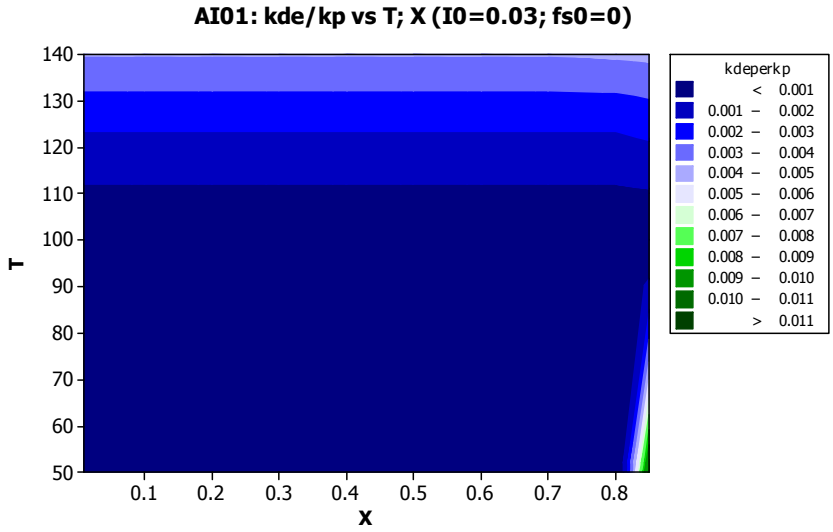


Figure 44: Ratio kde/kp as a function of temperature and conversion (AI01).

Despite this small region of the glassy state with a high ratio k_{de}/k_p, it can be noted that the ratio remains constant over the entire region. In the region

above Tg, the ratio increases with increasing temperature, as it would be expected.

This result confirms the non-significant effect of depropagation in reactions at lower temperature ranges as evaluated by the experiments in the literature (commonly, below Tg).

Figure 45 displays the dynamics of polymerization in the two conditions (below and above Tg). The degree of polymerization increases by orders of magnitude at high conversion when below Tg. The rate of reaction remains virtually stable above Tg.

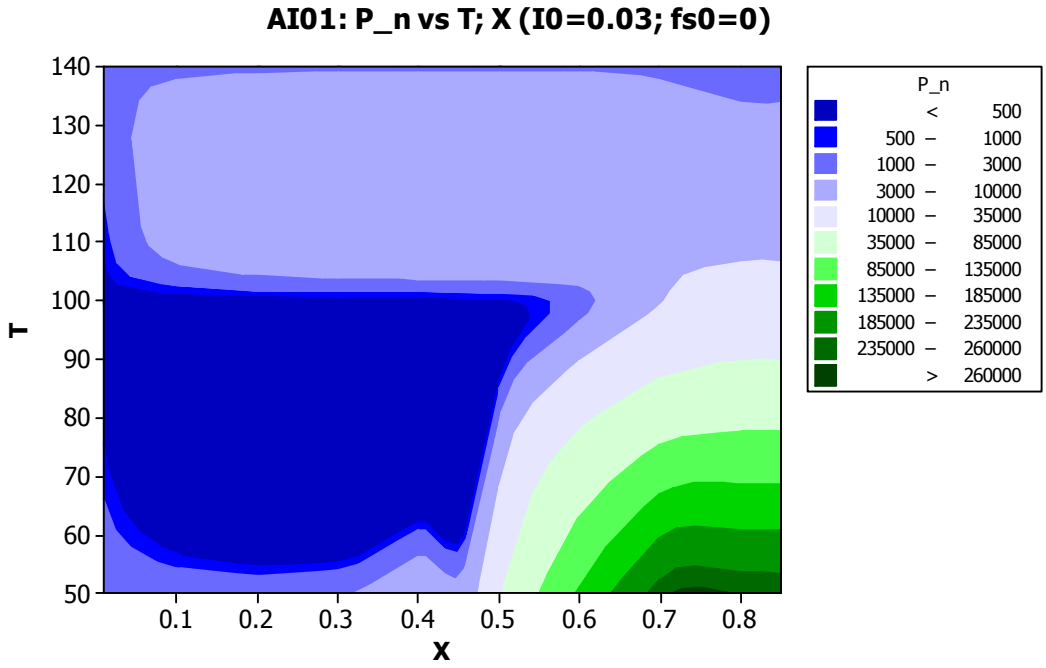


Figure 45: Pn as a function of temperature and conversion (AI01).

In Figure 46, one can observe the demonstration of the low degree of polymerization in temperatures above Tg, with model AI02, as a net result of low k_p calculated with equation 65. Because of these results, model AI02 can only be used at temperatures below Tg.

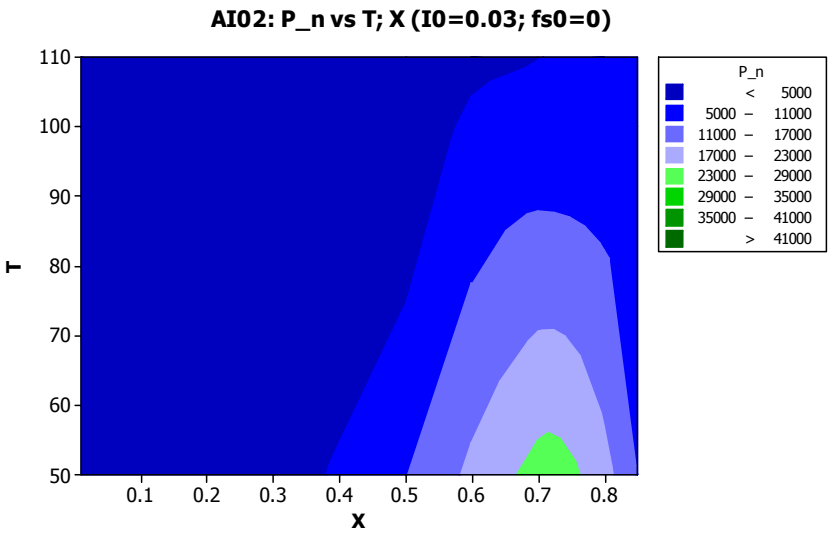


Figure 46: Pn as a function of temperature and conversion (AI02).

5.5.2. Solution polymerization

The following plots show the variation of k_p and k_t as a function of conversion and solvent fraction. Both models, AI01 (Figure 47) and AI02 (Figure 48), behave similarly. The temperature tested here, 50°C, is below Tg. Extrapolations for AI02 at temperatures above Tg will fall on the same issues as discussed in bulk polymerization; hence those conditions should not be modeled with AI02.

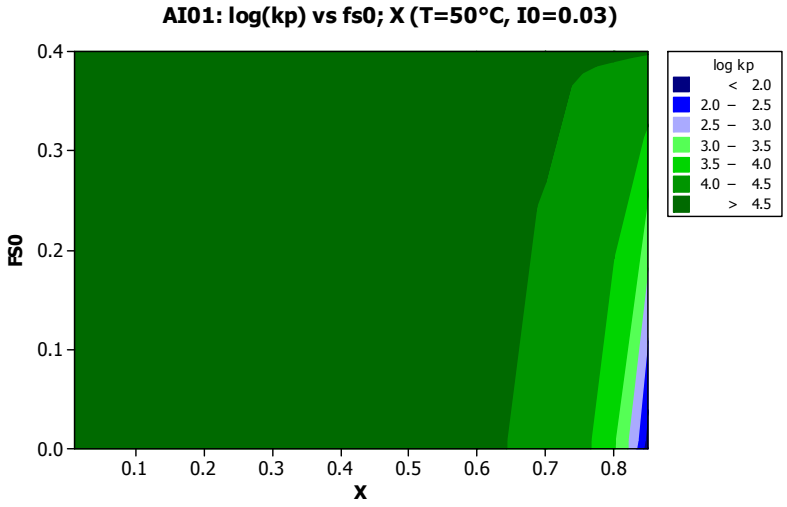


Figure 47: kp as function of solvent fraction and conversion (AI01).

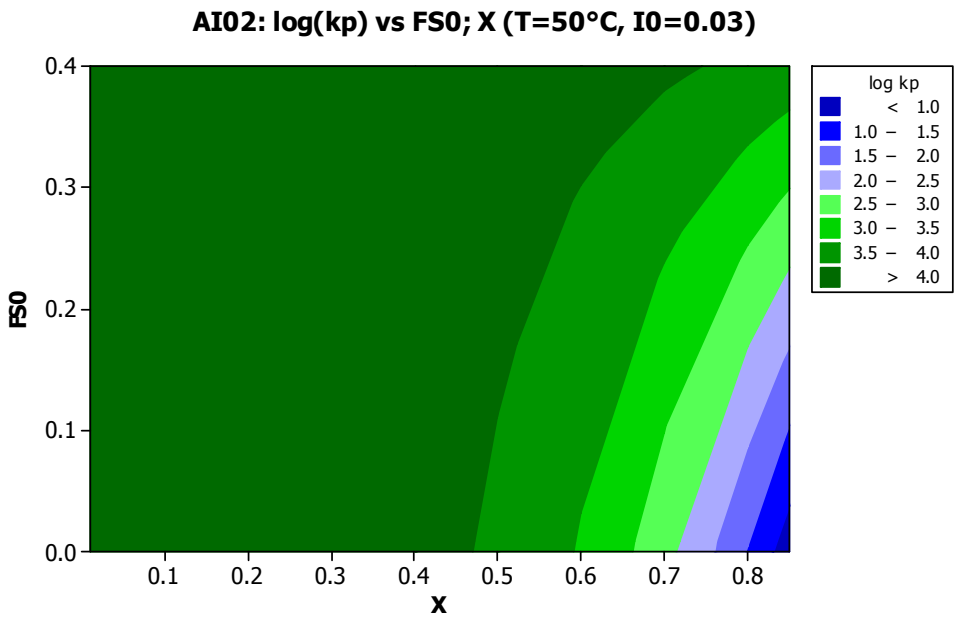


Figure 48: kp as a function of solvent fraction and conversion (AI02).

Figures 49 and 50 show the same evaluation of k_t for both AI01 and AI02 as function of the solvent fraction and conversion:

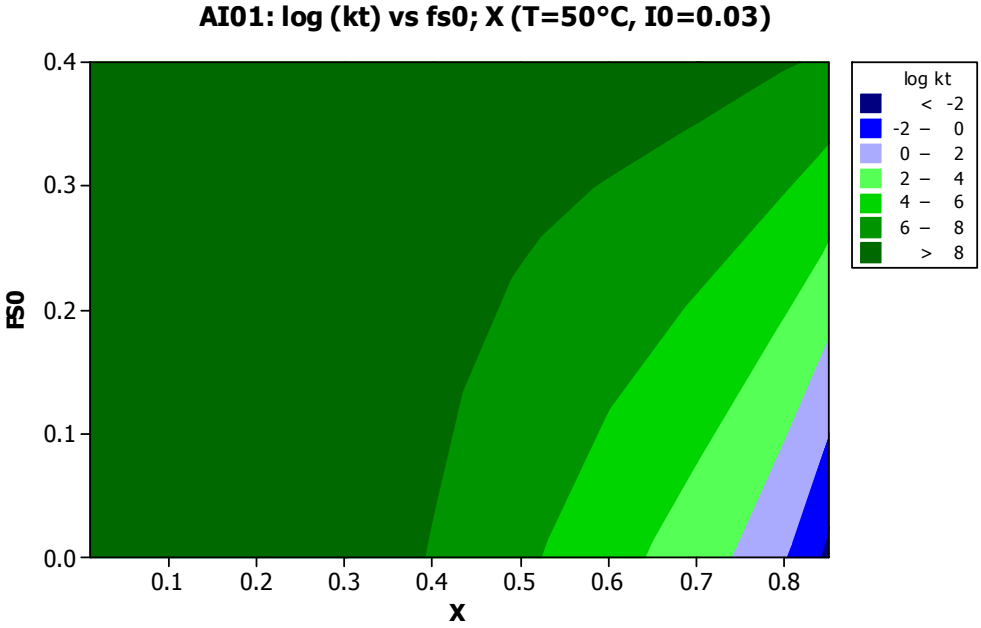


Figure 49: kt as a function of solvent fraction and conversion (AI01).

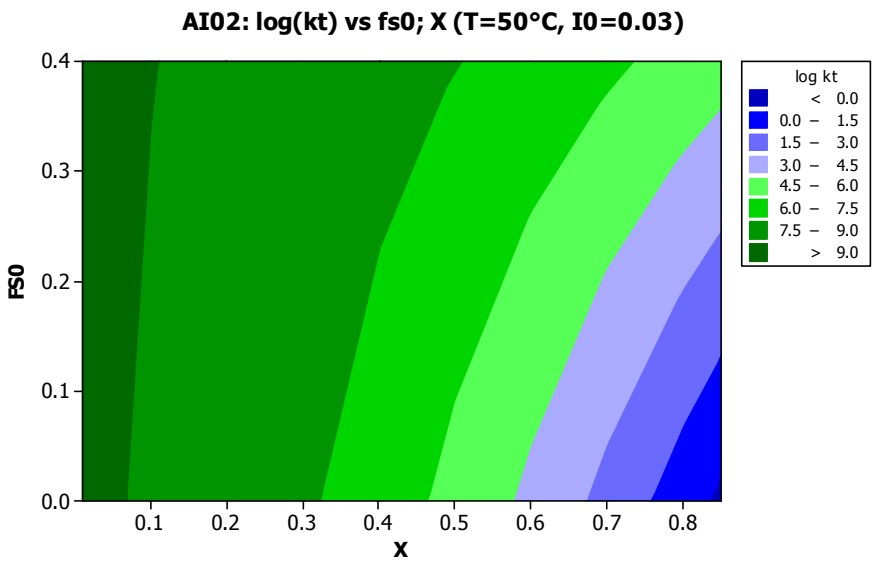


Figure 50: kt as a function of solvent fraction and conversion (AI02).

5.5.3. Cumulative molecular weight and polydispersity

The influence of initiator concentration and conversion in cumulative molecular weight as modeled by AI01 and AI02 is shown in Figures 51 and 52. This result confirms the experience of the inverse relationship between initiator concentration and molecular weight:

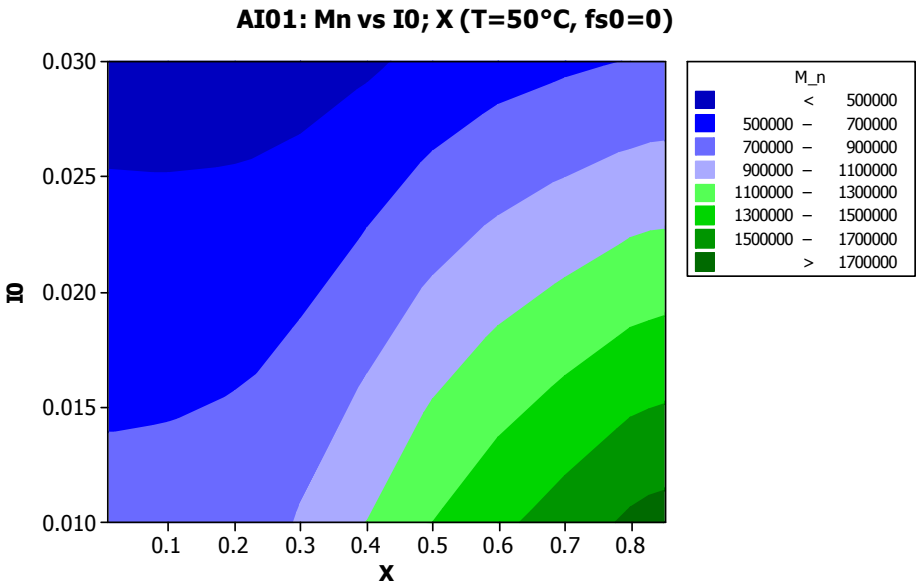


Figure 51: Cumulative Mn as function of initial initiator concentration (I_0) and conversion (AI01).

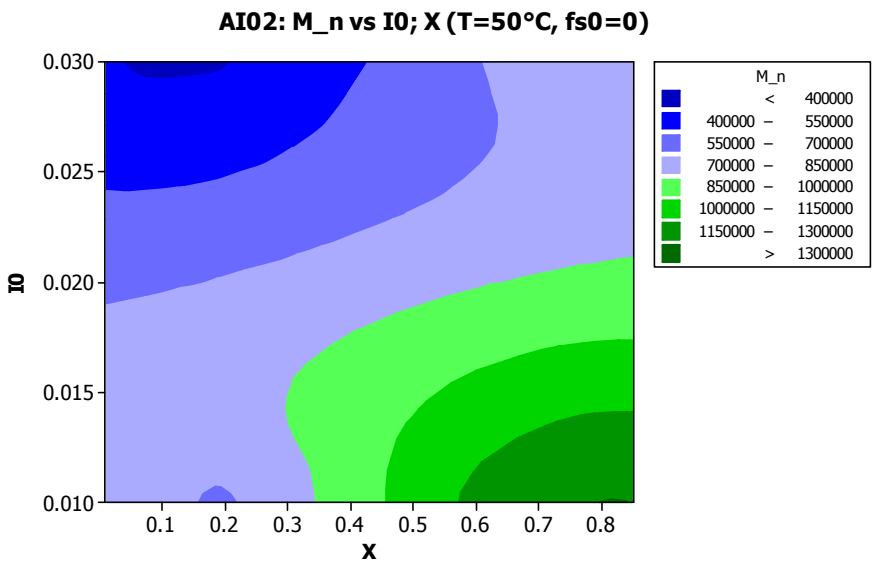


Figure 52: Cumulative Mn as a function of initial initiator concentration (I_0) and conversion (AI02).

The development of the polydispersity index (PDI) is shown in Figures 53 and 54. This demonstrates that the gel effect will increase PDI, as polymer chain size increases considerably with an increase in k_p.

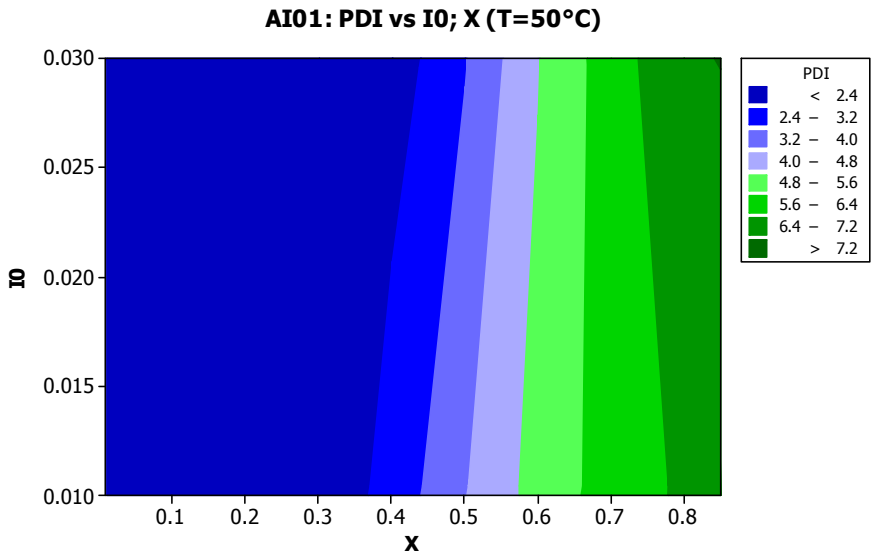


Figure 53: PDI as a function of initiator concentration and conversion (AI01).

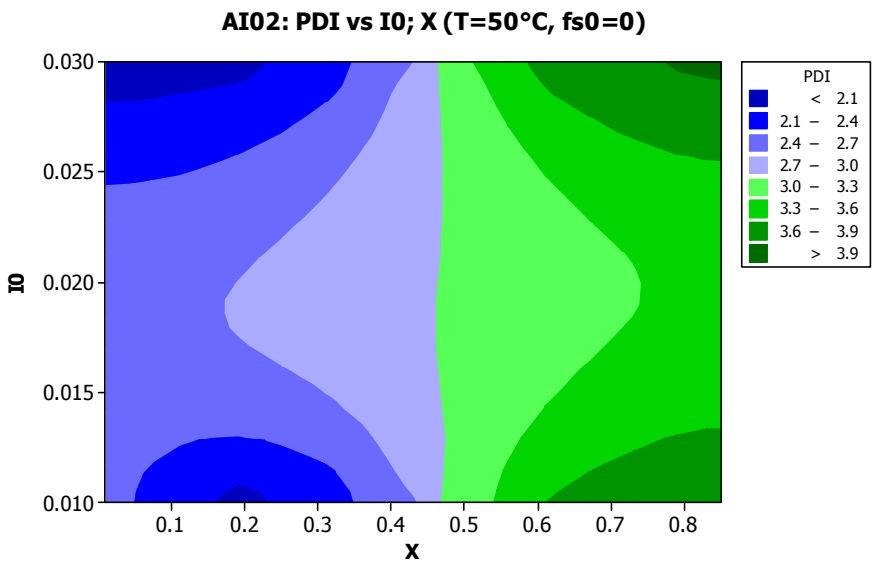


Figure 54: PDI as a function of initiator concentration and conversion (AI02).

The polydispersity index, as modeled by AI02, is considerably lower than AI01 for high conversion rates.

The effect of temperature in model AI01, the only one suitable for temperatures above Tg, is shown in Figures 55 and 56. It is remarkable the difference of behavior between regions below and above Tg, as a direct consequence of the glassy behavior in temperatures close to Tg.

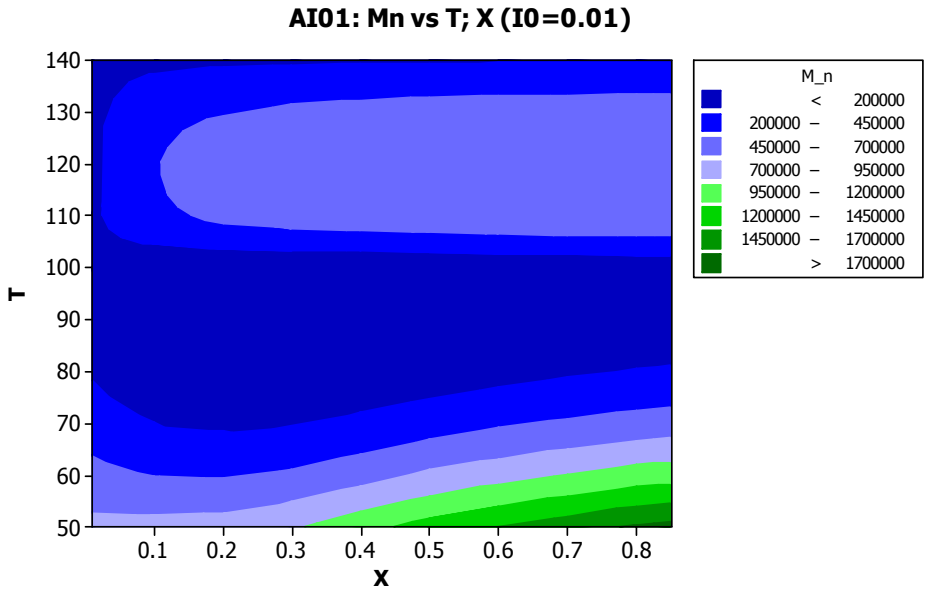


Figure 55: Cumulative Mn as a function of temperature and conversion (AI01).

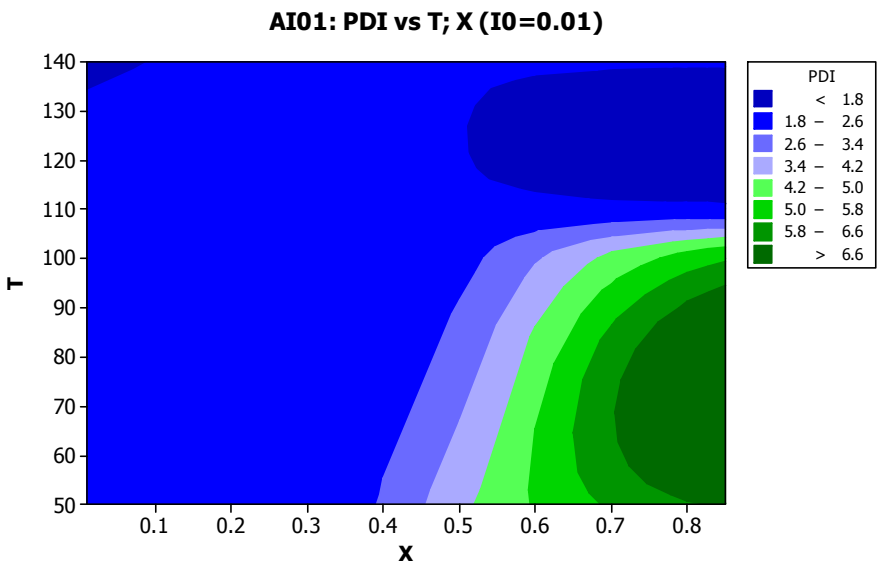


Figure 56: PDI as a function of temperature and conversion (AI01).

5.5.4. Summary of the sensitivity analysis

The results of the sensitivity analysis with two proposed models, AI01 and AI02, which were previously tested with a selection of experiments from the literature, were tested in conditions commonly present in semi-batch reactors. The particular case of temperatures above Tg was explored, as the experiments tested previously were mainly concentrated in the temperature zone below the Tg of the polymer.

The results clearly show the inadequacy of AI02 for such temperature range. It is important to point out that the parent model SBGSG was originally developed and tested in the T<Tg region, so its prediction capability above Tg was not expected to be accurate.

The experiment AI01 showed reasonable overall results, with behavior as expected by common sense. Nonetheless, these results could not be verified with actual experiments in that region, and this remains as a requirement for full validation of this model.

CHAPTER VI

6. CONCLUSION AND RECOMMENDATIONS

This chapter summarizes the conclusions and recommendations based on the results presented in this dissertation.

6.1. Conclusion

A selection of the current models for MMA polymerization was explored with experimental data from the literature, and results were compared. Selected models were the RSG/SG and SBGSG models. These models were expanded with two new models, AI01 and AI02, which incorporate radical transfer to monomer, termination by combination and depolymerization.

Both AI01 and AI02 models improved the prediction of molecular weight and, in the case of AI02, also a noticeable improvement in the prediction of conversion is noticed in the temperature range below Tg (all comparisons were done by analyzing the relative root mean square errors between experimental and calculated results).

AI01 stands out as presenting the best results for molecular weight determination in bulk polymerization and presented the best results for polymerization above Tg. Limitations are present in the step change processes, as discussed before. The model AI02 underestimates k_p and k_t by orders of magnitude in this temperature region, and thus this model is not recommended for high-temperature polymerizations.

AI02 provides a good model for solution polymerization at temperature ranges below Tg and non-steady states (temperature step changes), when considering the revision of experimental data that apparently was flawed in the SG model. The prediction of molecular

The new models are viable alternatives for bulk, solution, and non-isothermal MMA polymerization processes analysis, as long as the user keeps in mind the limitation of each of these models. Depending on the specific requirements of the user, the models can be easily switched, as to make the use of the best each has to offer (see Table 14 for a comparison of both models):

Table 14: Comparison of models AI01 and AI02 – prediction of conversion, Mn, and Mw.

Process	AI01	AI02
Isothermal, T<Tg	+++	+++
Isothermal, T>Tg	++	---
Step temperature changes	--	++
Intermediate feed (M, I)	++	++
Solution polymerization (fs_0<0.4)	+	+
Use of CTAs	-	---
Non-isothermal process	+++	+

+ better, - worse

A program that allows the simulation of several different scenarios with great flexibility, named *Reactormodel*, was coded with Matlab. The current program can easily support future studies of this type of processes because its modular nature allows the inclusion of new models, parameters, and experiments effortlessly.

6.2. Recommendations

The following recommendations may provide insights for future studies in the area of polymerization of vinylic monomers.

6.2.1. Error calculation and data fitting

It has been noticed that the root mean square methods used to fit models, such as the box complex technique, used in the present work and elsewhere (Ray; Saraf; Gupta, 1995) may cause the algorithms to diverge from optimal results because those methods only consider the error in the y-axis.

It is recommended that future studies use methods that compute the error orthogonal to the fitted curve (considering the error in x-axis as well). The models based on SRG model are particularly prone to this error, as the gel effect as modeled creates very stiff conversion curves, as seen in Figure 57. A method based on orthogonal errors could also improve issues with local minima found during optimization of some of the models.

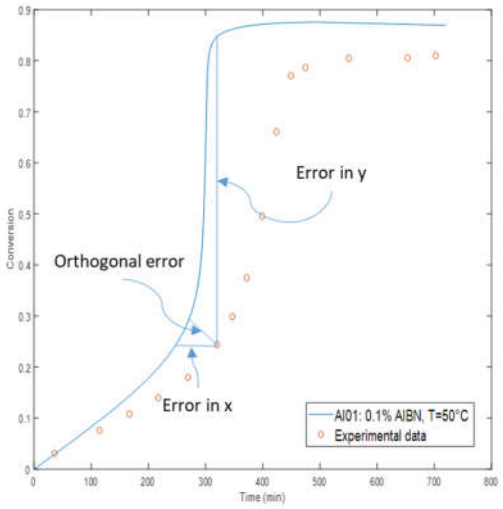


Figure 57: Error in y versus orthogonal error in SG model.

6.2.2. Model Al02 at high-temperature ranges – new parameters

The fit of the model Al02 at high-temperature ranges could be further explored by adjusting the coefficients and including new ones or including conditional equations based on free-volume to include the gel effect in an effective and accurate model.

6.2.3. Generalization of models Al01 and Al02

The generality of the current model can be investigated by testing it with different monomers, such as styrene, acrylates, methacrylates, acrylonitrile, acrylamide, and other vinylic monomers (specific parameters fitting/determination required). The extension of this model to copolymerization is of particular interest, as most of the polymers in the adhesives and coatings industry are synthesized out of two or more monomers.

6.2.4. Emulsion polymerization

The inclusion of transfer to monomer mechanism provides a valuable tool to calculate the exiting monomer radicals in emulsion polymerization systems. Not only that – emulsion polymerization can also be benefited by the modeling of the gel effect, particularly in starved polymerizations (where the polymer fraction can be kept high during the course of the polymerization).

6.2.5. Chain transfer agents

There is a significant gap for models of polymerizations of MMA with CTAs when the ones derived from RSG and SBGSG are considered. Other relevant models, not explored here, could be easily adapted to test their fit in all conditions tested here (bulk and solution isothermal and non-isothermal semi-batch reactors). This remains as a good opportunity for improvements.

6.2.6. Opportunities for molecular weight prediction

The current gap in molecular weight prediction for methyl methacrylate may indicate the requirement of additional mechanisms. Examples of possible mechanisms are the polymerization of the terminal double bond present in half of the polymer chains terminated by the disproportionation mechanism (most relevant termination mechanism for MMA), creating branches which will increase Mw (Figure 58), and other mechanisms such as radical transfer to polymer.

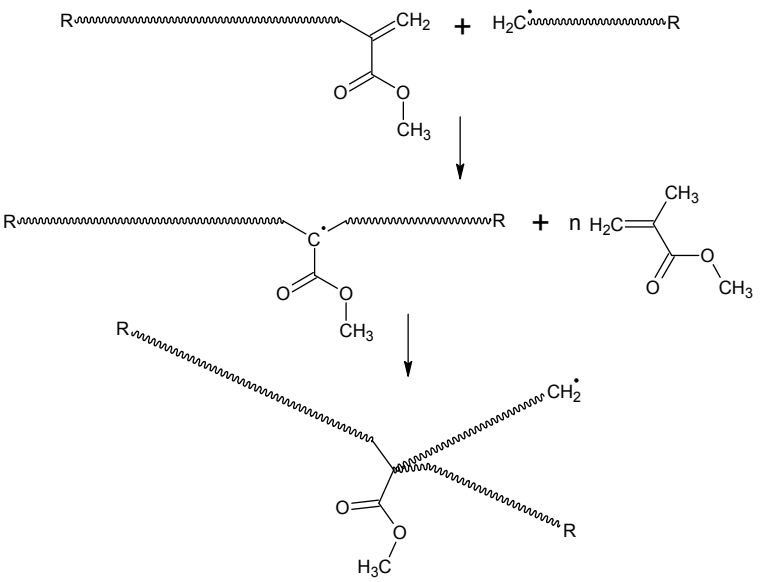


Figure 58: Molecular weight increase by polymerization of terminal double bonds resulted from termination by disproportionation.

6.2.7. Hybrid models

The hybridized model, briefly evaluated in section 5.4, could also be further explored. The concept of using two (or more) models to calculate the kinetic

constants at different stages of polymerization can be investigated. The equations (68) and (69) provide a basis to tests with any kinetic models, including others not evaluated here.

6.2.8. Solution polymerization

The present models were not validated, in the present work, for low solvent fractions. It is desirable that a more generic model is developed, as to allow higher fractions of solvent in the reactor. Most of the solution polymerizations will be carried out in low solvent fractions, below f_{s0}=0.4, but higher loads might be required for the polymerization of thermoplastic resins (which require very high molecular weight properties, and hence, high solvent fraction, possibly above the limit of f_{s0}=0.4).

The model AI01 adopted the definition of f_{s0} as suggested by the original model SG. However, this factor, which is based on the initial concentration of solvents, is not generic enough to allow non-steady processes with an intermediate feed of solvents. The instantaneous solvent fraction could be tested as a possible alternative in future studies (this hypothesis was not tested due to lack of data).

6.2.9. Depropagation

The limited data available for high-temperature polymerization of MMA restricted the test of the depropagation reaction, as to confirm the behaviors observed in the model. The first behavior is the peak of conversion observed in the non-isothermal process. The peak occurred in between two samples, which makes it difficult to confirm this behavior with the experimental data. It is desirable to confirm the activation energy with modern methods.

The variation of 12%, assumed in the experiment, could well be within the variation of the method used at the time of its measurement. If the current value is confirmed, the mechanism of depropagation would have to be reviewed in order to reflect the observation of experimental data.

REFERENCES

Achilias, D. (2007). A Review of Modeling of Diffusion Controlled Polymerization Reactions. *Macromolecular Theory and Simulations*, pp. 319-347.

Achilias, D., & Kiparissides, C. (1988). Modeling of Diffusion-Controlled Free-Radical Polymerization Reactions. *Journal of Applied Polymer Science*, pp. 35, 1303-1323.

Achilias, D., & Kiparissides, C. (1992). Development of a General Mathematical Framework for Modeling Diffusion-Controlled Free-Radical Polymerization Reactions. *Macromolecules*, pp. 25, 3739-3750.

Armitage, P., Hill, S., Johnson, A., Mykytiuk, J., & Turner, J. (1988). Bulk polymerization of methyl methacrylate: Part I: some kinetic and modelling considerations for isothermal reactions. *Polymer*, pp. 29(12) 2221-2228.

Balke, S. T. (1972). The Free Radical Polymerization of Methyl Methacrylate to High Conversions. *PhD Thesis*.

Bevington, J., & Melville, H. (1954). The Termination Reaction in Radical Polymerizations. II. Polymerizations of Styrene at 60° and of Methyl Methacrylate at 0 and 60°, and the Copolymerization of these Monomers at 60°. *Journal of Polymer Science*, pp. 463-476.

Chiu, W. Y., Carratt, G. M., & Soong, D. (1983). A Computer Model for the Gel Effect in Free-Radical Polymerization. *Macromol., 16*, pp. 348-357.

Chiu, W. Y., Carratt, G. M., & Soong, D. S. (1983). A Computer Model for the Gell Effect in Free-Radical Polymerization. *Macromolecules*, pp. 16, 348-357.

Curteanu, S., & Bulacovschi, V. (1999). Free Radical Polymerization of Methyl Methacrylate: Modeling and Simulation under Semibatch and Noninsothermal Reactor Conditions. *Journal of Applied Polymer Science*, pp. 74, 2561-2570.

Destruel, P., Taufer, M., & Granel, C. (1991). Modeling the Bulk Polymerization of Methyl Methacrylate with Glycol Dimercapto Acetate as Chain Transfer Agent. *Journal of Applied Polymer Science*, pp. 42, 93-97.

Farina, M. (1987). Chemistry and Kinetics of the Chain Transfer Reaction. . *Makromolekulare Chemie. Macromolecular Symposia.*, pp. 10-11(1), 255-272.

Fazenda, J. M. (2009). *Tintas Ciência e Tecnologia.* São Paulo: Blucher.

Flory, P. J. (1953). *Principles of Polymer Chemistry.* Cornell.

Fujita, H., Kishimoto, A., & Kinya, M. (1960). Concentration and temperature dependence of diffusion coefficients for systems polymethyl acrylate and n-alkyl acetates. *Trans. Faraday Soc.*, pp. 56, 424-437.

Gao, J., & Penlidis, A. (1996). A Comprehensive Simulator/Database Package for Reviewing Free-Radical Homopolymerizations. *Rev. Macromol. Chem. Phys.*, pp. C36(2), 199-404.

George, F. S. (1940). *US Patent No. 2225471A.*

Ghosh, P., Gupta, S., & Saraf, D. (1998). An Experimental Study on Bulk and Solution Polymerization of Methyl Methacrylate with Responses to Step Changes in Temperature. *Chemical Engineering Journal*, pp. 25-35.

Gilbert, R. (1995). *Emulsion Polymerization, a Mechanistic Approach.* London: Academic Press Limited.

Jahanzad, F., Kazemi, M., Sajjadi, S., & Taromi, F. (1993). n-Dodecyl mercaptan transfer constant in polymerization of methyl. *Polymer*, pp. 34(16), 3542-3544.

Kukulj, D., Davis, T., & Gilbert, R. (1995). Chain Transfer to Monomer in the Free-Radical Polymerizations of Methyl Methacrylate, Styrene, and α-Methylstyrene. *Macromolecules*, pp. 31(4), 994-999.

Lagarias, J., Reeds, J., Wright, M., & Wright, P. (1998). Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions. *SIAM Journal of Optimization*, pp. 9(1), 112-147.

Louie, B., Carratt, G., & Soong, D. (1985). Modeling the Free Radical Solution and Bulk. *Polymerization of Methyl Methacrylate*, pp. 3985-4012.

Madruga, E., San Román, J., & Benedi, P. (1990). High Conversion Polymerization of Methyl Methacrylate in the Presence of n-Dodecylmercaptan. *Journal of Applied Polymer Science*, pp. 41, 1133-1140.

Marten, F., & Hamielec, A. (1979). *ACS Symp. Ser.*, pp. No. 104, 43.

Marten, F., & Hamielec, A. (1982). High-conversion diffusion-controlled polymerization of styrene. *Journal of Applied Polymer Science*, pp. 27(2), 489-505.

Mastan, E., & Zhu, S. (2015). Method of Moments: A Versatile Tool for Deterministic Modeling of Polymerization Kinetics. *European Polymer Journal*, pp. 68, 139-160.

Matyjaszewski, K. (2002). *Handbook of Radical Polymerization.* Wiley-Interscience.

Mishra, M. K., & Yagci, Y. (1998). *Handbook of Vinyl Polymers.* CRC Press.

Nising, P. (2006). High Temperature Radical Polymerization of Methyl Methacrylate in a Continuous Pilot Scale Process. *PHd Thesis*.

Nising, P., Meyer, T., Carloff, R., & Wicker, M. (2005). Thermal Initiation of MMA in High Temperature Radical Polymerizations. *Macromolecular Materials and Engineering*, pp. 311-318.

O'Brien, J., & Gornick, F. (1955). Chain Transfer in the Polymerization of Methyl Methacrylate. I. Transfer with Monomer and Thiols. The Mechanism of the Termination Reaction at 60°C. *Journal of the American Chemical Society*, pp. 4757-4763.

Pryor, W. A. (1967). Introduction to Free Radical Chemistry. In W. A. Pryor. Edgad Blucher (edição traduzida).

Pryor, W., & Coco, J. (1970). Computer Simulation of the Polymerization of Styrene. The Mechanism of Thermal Initiation and the Importance of Primary Radical Termination. *Macromolecules*, pp. 3(5), 500-508.

Ray, A., Saraf, D., & Gupta, S. (1995). Free Radical Polymerizations Associated with the Trommsdorff Effect Under Semibatch Reactor Conditions. I: Modeling. *Polymer Engineering and Science*, pp. 35(16), 1290-1299.

Sangwai, J., Bhat, S., Gupta, S., Saraf, D., & Gupta, S. (2005). Bulk Free Radical Polymerizations of Methyl Methacrylate Under Non-Isothermal Conditions and With Intermediate Addition of Initiator: Experiments and Modeling. *Polymer*, pp. 11451-11462.

Schulz, V., & Harborth, G. (1947). Über den mechanismus des explosiven polymerisationsverlaufes des methacrylsäuremethylesters. *Macromolecular Chemistry and Physics*, pp. 106-139.

Seth, V., & Gupta, S. (1995). Free Radical Polymerizations Associated with the Trommsdorff Effect Under Semibatch Reactor Conditions: An Improved Model. *Journal of Polymer Engineering*, pp. 283-326.

Soh, S., & Sundberg, D. (1982). Diffusion-controlled Vinyl Polymerization. IV. Soh, S. K., & Sundberg, D. C. (1982). Diffusion-controlled vinyl polymerization. IV. Comparison of theory and experiment. *Journal of Polymer Science: Polymer Chemistry Edition*, pp. 20(5), 1345–1371.

Tefera, N., Weickert, G., & Westerterp, K. (1997). Modeling of Free Radical Polymerization up to High Conversion. I. A Method for the Selection of Models by Simultaneous Parameter Estimation. *J. Appl. Polym. Sci.*, pp. 63: 1649-1661.

Trommsdorff, V., Köhle, H., & Lagally, P. (1948). Zur Polymerisation des Methacrylsäuremethylesters. *Macromolecular Chemistry and Physics*, pp. 169-198.

Vivaldo-Lima, E., Hamielec, A., & Wood, P. (1994). Auto-Acceleration Effect in Free Radical Polymerization. A Comparison of the CCS and MH models. *Polymer Reaction Engineering*, pp. 2(1-2), 17-85.

Vrentas, J., & Duda, J. (1977). Diffusion in polymer-solvent systems. I. Reexamination of the free-volume theory. *Journal of Polymer Science: Polymer Physics Edition*, pp. 15(3), 403-416.

Walling, C., & Briggs, E. (1946, July 19). The Thermal Polymerization of Methyl Methacrylate. *Jornal of the American Chemical Society*, pp. 1141-1145.

APPENDIX

A. Demonstration of the method of moments applied to present models

The method of moments can be used as a tool to simplify the equation system of polymerization reactions, by reducing an infinite set of equations into only six equations, which can capture important statistical information of the reaction system. This method has been reviewed elsewhere (Mastan; Zhu, 2015), and this appendix demonstrates the development of the substitute equations.

The starting point is the definition of the nth moments for radical polymer chains and dead polymer chains, with equations (36) and (37):

$$\lambda_i = \sum_{j=0}^{\infty} j^i R_j^* \tag{36}$$

$$\mu_i = \sum_{j=0}^{\infty} j^i D_j \tag{37}$$

In order to compute these moments, one has to sum all reactions displayed in Table 3 and replacing the respective sum of radical terms by λ_0 (0^{th} moment of radicals):

$$\frac{dR_0^*}{dt} = 2fk_dI - k_p^{eff}\frac{M}{V_r}R_0^* - k_{fm}\frac{M}{V_r}R_0^* + f_{cta}k_{fc}\frac{CTA}{V_r}\lambda_0 - k_t\frac{\lambda_0}{V_r}R_0^* \tag{70}$$

$$\frac{dR_1^*}{dt} = k_p^{eff}\frac{M}{V_r}R_0^* - k_p^{eff}\frac{M}{V_r}R_1^* + k_{fm}\frac{M}{V_r}\lambda_0 - f_{cta}k_{fc}\frac{CTA}{V_r}R_1^* - k_t\frac{\lambda_0}{V_r}R_1^* \tag{71}$$

$$\vdots$$

$$\frac{dR_n^*}{dt} = k_p^{eff}\frac{M}{V_r}R_{n-1}^* - k_p^{eff}\frac{M}{V_r}R_n^* - k_{fm}\frac{M}{V_r}R_n^* - f_{cta}k_{fc}\frac{CTA}{V_r}R_n^* - k_t\frac{\lambda_0}{V_r}R_n^* \tag{72}$$

The moments of radical and dead polymer chains are obtained by summing up equations (70) to (72), with the help of the definitions from equations (36) and (37). For the first moment of radical chains, λ_0, the development is as follows:

$$\frac{d\lambda_0}{dt} = 2fk_dI + \sum_{n=0}^{\infty} n^0 \left(-k_p^{eff}\frac{M}{V_r}R_n^* + k_p^{eff}\frac{M}{V_r}R_{n-1}^* - k_{fm}\frac{M}{V_r}R_n^* + k_{fm}\frac{M}{V_r}\lambda_0 - f_{cta}k_{fc}\frac{CTA}{V_r}R_n^* + f_{cta}k_{fc}\frac{CTA}{V_r}\lambda_0 - k_t\frac{\lambda_0 R_n^*}{V_r}\right)$$

$$\frac{d\lambda_0}{dt} = 2fk_dI - k_p^{eff}\frac{M}{V_r}\lambda_0 + k_p^{eff}\frac{M}{V_r}\lambda_0 - k_{fm}\frac{M}{V_r}\lambda_0 + k_{fm}\frac{M}{V_r}\lambda_0 - f_{cta}k_{fc}\frac{CTA}{V_r}\lambda_0 + f_{cta}k_{fc}\frac{CTA}{V_r}\lambda_0 - k_t\frac{\lambda_0\lambda_0}{V_r}$$

$$\frac{d\lambda_0}{dt} = 2fk_dI - k_t\frac{{\lambda_0}^2}{V_r} \tag{44}$$

The first moment of polymer radicals follow a similar procedure:

$$\frac{d\lambda_1}{dt} = \sum_{n=0}^{\infty} n^1 \left(-k_p^{eff}\frac{M}{V_r}R_n^* + k_p^{eff}\frac{M}{V_r}R_{n-1}^* - k_{fm}\frac{M}{V_r}R_n^* + k_{fm}\frac{M}{V_r}\lambda_0 - f_{cta}k_{fc}\frac{CTA}{V_r}R_n^* + f_{cta}k_{fc}\frac{CTA}{V_r}\lambda_0 - k_t\frac{\lambda_0 R_n^*}{V_r}\right)$$

$$\frac{d\lambda_1}{dt} = -k_p^{eff}\frac{M}{V_r}\sum_{n=0}^{\infty} nR_n^* + k_p^{eff}\frac{M}{V_r}\sum_{n=0}^{\infty}[(n-1)+1]R_{n-1}^* - k_{fm}\frac{M}{V_r}\sum_{n=0}^{\infty} nR_n^* + k_{fm}\frac{M}{V_r}\lambda_0 - f_{cta}k_{fc}\frac{CTA}{V_r}\sum_{n=0}^{\infty} nR_n^* + f_{cta}k_{fc}\frac{CTA}{V_r}\lambda_0 - k_t\frac{\lambda_0}{V_r}\sum_{n=0}^{\infty} nR_n^*$$

$$\frac{d\lambda_1}{dt} = -k_p^{eff}\frac{M}{V_r}\sum_{n=0}^{\infty} nR_n^* + k_p^{eff}\frac{M}{V_r}\sum_{n=0}^{\infty}(n-1)R_{n-1}^* + k_p^{eff}\frac{M}{V_r}\sum_{n=0}^{\infty} R_{n-1}^* - k_{fm}\frac{M}{V_r}\lambda_1 + k_{fm}\frac{M}{V_r}\lambda_0 - f_{cta}k_{fc}\frac{CTA}{V_r}\lambda_1 + f_{cta}k_{fc}\frac{CTA}{V_r}\lambda_0 - k_t\frac{\lambda_0}{V_r}\lambda_1$$

$$\frac{d\lambda_1}{dt} = k_p^{eff}\frac{M}{V_r}\sum_{n=0}^{\infty} R_{n-1}^* - k_{fm}\frac{M}{V_r}\lambda_1 + k_{fm}\frac{M}{V_r}\lambda_0 - f_{cta}k_{fc}\frac{CTA}{V_r}\lambda_1 + f_{cta}k_{fc}\frac{CTA}{V_r}\lambda_0 - k_t\frac{\lambda_0}{V_r}\lambda_1$$

$$\frac{d\lambda_1}{dt} = k_p^{eff}\frac{M}{V_r}\lambda_0 + \left(k_{fm}M + f_{cta}k_{fc}CTA\right)\frac{(\lambda_0-\lambda_1)}{V_r} - k_t\frac{\lambda_0\lambda_1}{V_r} \tag{45}$$

By invoking equation (23), k_p^{eff} can be directly calculated from k_p and k_{de} to include the depropagation reaction:

$$\frac{d\lambda_1}{dt} = (k_p - \frac{k_{de}}{M})\frac{M}{V_r}\lambda_0 + \left(k_{fm}M + f_{cta}k_{fc}CTA\right)\frac{(\lambda_0 - \lambda_1)}{V_r} - k_t\frac{\lambda_0\lambda_1}{V_r} \quad (46)$$

The second moment of radical chains follow the same procedure, and the result is equation (47):

$$\frac{d\lambda_2}{dt} = (k_p - \frac{k_{de}}{M})\frac{M}{V_r}(2\lambda_1 + \lambda_0) + \left(k_{fm}M + f_{cta}k_{fc}CTA\right)\frac{(\lambda_0 - \lambda_2)}{V_r} - k_t\frac{\lambda_0\lambda_2}{V_r} \quad (47)$$

The 0th moment of dead polymer chains can be determined from the differential equation (73):

$$\frac{dD_n}{dt} = \frac{k_{tc}}{2V_r}\sum_{m=0}^{n} R_m^* R_{n-m}^* + \left(k_{fm}M + f_{cta}k_{fc}CTA\right)\frac{R_n^*}{V_r} + k_{td}\frac{\lambda_0 R_n^*}{V_r} \quad (73)$$

The same procedure adopted for the moments of radical chains is adopted here, using the definitions of equations (36) and (37):

$$\frac{d\mu_0}{dt} = \sum_{n=0}^{\infty} n^0 \left(\frac{k_{tc}}{2V_r}\sum_{m=0}^{n} R_m^* R_{n-m}^* + k_{td}\frac{\lambda_0 R_n^*}{V_r} + \left(k_{fm}M + f_{cta}k_{fc}CTA\right)\frac{R_n^*}{V_r}\right)$$

$$\frac{d\mu_0}{dt} = \frac{k_{tc}}{2V_r}\sum_{n=0}^{\infty}\sum_{m=0}^{n} R_m^* R_{n-m}^* + k_{td}\frac{{\lambda_0}^2}{V_r} + \left(k_{fm}M + f_{cta}k_{fc}CTA\right)\frac{\lambda_0}{V_r} \quad (74)$$

The following development can solve the summations of the first term in equation 74:

	m=0	m=1	m=2	m=3	⋯	$\sum \rightarrow$
n=0	$R_0^* R_0^*$				⋯	
n=1	$R_0^* R_1^*$	$R_1^* R_0^*$			⋯	
n=2	$R_0^* R_2^*$	$R_1^* R_1^*$	$R_2^* R_0^*$		⋯	
n=3	$R_0^* R_3^*$	$R_1^* R_2^*$	$R_2^* R_1^*$	$R_3^* R_0^*$	⋯	
⋮	⋮	⋮	⋮	⋮	⋮	
$\sum \downarrow$	$R_0^*\lambda_0$	$R_1^*\lambda_0$	$R_2^*\lambda_0$	$R_3^*\lambda_0$	⋯	$\lambda_0\lambda_0$

Thus, the 0th moment of dead polymer chains is demonstrated to be:

$$\frac{d\mu_0}{dt} = \left(\frac{k_{tc}}{2} + k_{td}\right)\frac{{\lambda_0}^2}{V_r} + \left(k_{fm}M + f_{cta}k_{fc}CTA\right)\frac{\lambda_0}{V_r} \quad (48)$$

A similar procedure is used to obtain the first and second moments of dead chains:

$$\frac{d\mu_1}{dt} = (k_{tc} + k_{td})\frac{\lambda_0\lambda_1}{V_r} + \left(k_{fm}M + f_{cta}k_{fc}CTA\right)\frac{\lambda_1}{V_r} \quad (49)$$

$$\frac{d\mu_2}{dt} = k_{tc}\left(\lambda_0\lambda_2 + {\lambda_1}^2\right) + k_{td}\lambda_0\lambda_2 + \left(k_{fm}M + f_{cta}k_{fc}CTA\right)\frac{\lambda_2}{V_r} \quad (50)$$

B. CCS model in Polymath

```
# Model CCS - Chiu, Carrat, Soong, Macromolecules, 1983
# Batch, isothermal reactor
# MMA

# Process parameters:
T=70                                                    # [°C]
f= 0.58                                                 # Initiator efficiency factor
MM_MMA=100.121                                          # [g/gmol]
Tg_p=114                                                # [°C]
ro_p=1.2                                                # [g/cm^3]
ro_M=0.973-1.164*(10^-3)*T                              # [g/cm^3]
t(0) = 0                                                # [min] Initial time
t(f) = 120                                              # [min] Final time

# kinetic constants:
kd=6.32*(10^16)*exp(-15.43*(10^3)/(T+273.15))           # [L/min]
kt0=5.88*(10^9)*exp(-701/(1.987*(T+273.15)))            # [L/(min*mol)]
kp=1/(1/kp0+theta_p*P_/( exp(2.3*phi_m_/(A+B*phi_m_))))  # [L/(min*mol)]
kp0=2.95*(10^7)*exp(-4353/(1.987*(T+273.15)))           # [L/(min*mol)]
ktc=0                                                   # [L/(min*mol)]
ktd=1/(1/kt0+theta_t*P_/( exp(2.3*phi_m_/(A+B*phi_m_))))  # [L/(min*mol)]
kt=ktc+ktd                                              # [L/(min*mol)]

# CCS Parameters:
A= 0.152
B=0.03
theta_t=4.9*10
theta_p=2.5*(10^2)

# ODEs:
d(I_) / d(t) = -kd*I_-eps*I_*Y0_*(1-x)*kp/(1+eps*x)
d(x) / d(t) = kp*(1-x)*Y0_
d(Y0_) / d(t) = -eps*(Y0_^2)*(1-x)*kp/(1+eps*x)+2*f*kd*I_-kt*(Y0_^2)
d(Y1_) / d(t) = -eps*Y1_*Y0_*(1-x)*kp/(1+eps*x)+2*f*kd*I_-kt*Y0_*Y1_+kp*Y0_*M0_*(1-x)/(1+eps*x)
d(Y2_) / d(t) = -eps*Y2_*Y0_*(1-x)*kp/(1+eps*x)+2*f*kd*I_-kt*Y0_*Y2_+kp*M0_*((1-
x)/(1+eps*x))*(2*Y1_+Y0_)
d(Q0_) / d(t) = -eps*Q0_*Y0_*(1-x)*kp/(1+eps*x)+ktd*(Y0_^2)+0.5*ktc*(Y0_^2)
d(Q1_) / d(t) = -eps*Q1_*Y0_*(1-x)*kp/(1+eps*x)+ktd*Y0_*Y1_+ktc*Y0_*Y1_
d(Q2_) / d(t) = -eps*Q2_*Y0_*(1-x)*kp/(1+eps*x)+ktd*Y0_*Y2_+ktc*(Y0_*Y2_+(Y1_^2))

# Initial conditions:
M0_=9.0982
I_(0) = 0.01548
x(0) = 0
Y0_(0) = 1e-20
Y1_(0) = 1e-20
Y2_(0) = 1e-20
Q0_(0) = 1e-20
Q1_(0) = 1e-20
Q2_(0) = 1e-20

# Auxiliary equations
Mn_=(Q1_+Y1_)/(Q0_+Y0_)*MM_MMA
Mw_=(Q2_+Y2_)/(Q1_+Y1_)*MM_MMA
Pd_=Mw_/Mn_
phi_m_=(1-x)/(1+eps*x)
P_=Y0_
R_=Y0_
eps=(ro_M-ro_p)/ro_p
```

C. Reactormodel code

reactormodel.m:

```
clear,clc;
close all;
flag_var=0;

fprintf ( 1, '\n' );
fprintf ( 1, 'REACTOR MODEL \n' );
fprintf ( 1, '  Mass and solution polymerization in batch and semi-batch reactors.\n' );
fprintf ( 1, '  Version 1.2\n' );
fprintf ( 1, '\n' );
fprintf ( 1, '  Literature experiments with MMA:\n' );
fprintf ( 1, '   3) CCS Model. I0=0.01548, T=70°C.\n' );
fprintf ( 1, '  11) Ray, Saraf and Gupta model. I0=0.01548, T=70°C.\n' );
fprintf ( 1, '  20) Seth and Gupta model. Run SI (Step temperature increase from 50°C to 70°C).\n' );
fprintf ( 1, '  21) Seth and Gupta model. Run IA50a2 (Intermediate feed of monomer and initiator).\n'
);
fprintf ( 1, '  23) Curteanu and Bulacovschi model. Example of Figure 5 of Ray, Saraf, Gupta.\n' );
fprintf ( 1, '  24) Sangwai et al model. Example of Figure 5 of Ray, Saraf, Gupta.\n' );
fprintf ( 1, '  31) Ray, Saraf and Gupta model. Fig 6 from Seth 1995. Inititor BPO. fs0=0.4, T=70.\n'
);
fprintf ( 1, '  32) Seth and Gupta model. GDMA as CTA. T=60°C, I0=0.000188. Fig 1. \n' );
fprintf ( 1, '  33) Model AI01. Gao and Penlidis 1996, figure 40 - Non-Isothermal. \n' );
fprintf ( 1, '  41) Model AI02. SD70(20)50, I0=0.01548. \n' );
fprintf ( 1, '  G)  Multi-plotting.\n' );
fprintf ( 1, '  H)  DoE.\n' );
fprintf ( 1, '  I)  Multi-testing.\n' );
fprintf ( 1, '  J)  Run experiments with alternative models.\n' );

fprintf ( 1, '\n' );
option=input('Select option: ','s');
fprintf ( 1, '\n' );

if 'G'==upper(option)
    fprintf ( 1, '\n' );
    fprintf ( 1, 'REACTOR MODEL - Multiplotting \n' );
    fprintf ( 1, '  Allows the plotting of multiple data out of generated tables.\n' );
    fprintf ( 1, '\n' );
    vec_expt=input('Type a vector [...,n-2,n-1,n] with the n experiments to be compared: ');
    fprintf ( 1, '\n' );
    data_plot(vec_expt,[])

elseif 'H'==upper(option)

    vec_expt=input('Type a vector [...,n-2,n-1,n] with the n experiments to be processed: ');
    fprintf ( 1, '\n' );

    DOE=importdata('DOE.csv');

    [nrow,ncol]=size(vec_expt);
    [nrow_2,ncol_2]=size(DOE.data);

    tic % Start elapsed time computation

for p=1:ncol
    expt_index=num2str(vec_expt(1,p));
    table_name=strcat('DOE_results_',expt_index,'.csv');
    error_table={'Conversion','Mn','Mw'};
    for o=1:nrow_2
        option=strcat(num2str(p),'_',num2str(o));
        [parameters_input,experiment]=loading_inputs(vec_expt(1,p));
        experiment.thetas_calc_rule='BFC';
        experiment.DOE_trial=DOE.data(o,5:ncol_2);

        % Integrate the experiment:
        flag_optimizer=1;
        [t,y,error_]=polymerize(experiment,parameters_input,option,flag_optimizer);
        error_table{o+1,1}=error_.error_total_conversion;
        error_table{o+1,2}=error_.error_total_Mn;
        error_table{o+1,3}=error_.error_total_Mw;

    end

    fid = fopen(table_name, 'w') ;
    fprintf(fid, '%s,', error_table{1,1:end-1}) ;
    fprintf(fid, '%s\n', error_table{1,end}) ;
    fclose(fid) ;
```

```
    dlmwrite(table_name,error_table(2:end,:),'-append');

end

elseif 'I'==upper(option)
    multitesting_f(1)

elseif 'J'==upper(option)
    multitesting_f(0)

else
    option=str2num(option);
    [parameters_input,experiment]=loading_inputs(option);

    % Option to allow selection of BFC / IOP in Gupta's models.
    if (strcmp(experiment.model,'Ray_Saraf_Gupta')||strcmp(experiment.model,'Seth_Gupta')) &&...
            ~isempty(experiment.theta_t_RSG)
        fprintf ( 1, ...
            'Use Best Fit Correlation (BFC) or Individually Optimized Parameters (IOP)? : \n' );
        option2=input('Select option: ','s');
        experiment.thetas_calc_rule=option2;
        fprintf ( 1, '\n' );
    end

    tic % Start elapsed time computation

    % Integrate the experiment according to selected models:
    flag_optimizer=0;
    vec_expt=[];
    [t,y,error_]=polymerize(experiment,parameters_input,option,flag_optimizer);

end

function multitesting_f(option)

if option==1
    vec_expt=input('Type a vector [...,n-2,n-1,n] with the n experiments to be processed: ');
    fprintf ( 1, '\n' );
else
    vec_expt=input('Type the experiment to be processed: ');
    fprintf ( 1, '\n' );
end

tic % Start elapsed time computation

[~,ncol]=size(vec_expt);
table_name='Error_summary';
error_table={'Experiment','Conversion','Mn','Mw','Model','ODE_set'};
model_selected=input('Enter model to be used: ');
balance_selected=input('Enter equations set to be used: ');

for o=1:ncol

    table_name=strcat(table_name,'_',num2str(vec_expt(1,o)));
    expt_index=num2str(vec_expt(1,o));
    option=vec_expt(1,o);
    [parameters_input,experiment]=loading_inputs(vec_expt(1,o));
    experiment.model=model_selected;
    experiment.ODE_eqns=balance_selected;
    experiment.thetas_calc_rule='BFC';
    if ncol==1
        experiment.flag_plot=1;
    end
    % Integrate the experiment:
    flag_optimizer=1;
    [~,~,error_]=polymerize(experiment,parameters_input,option,flag_optimizer);
    error_table{o+1,1}=num2str(option);
    error_table{o+1,2}=error_.error_avg_conversion_minus_NaN;
    error_table{o+1,3}=error_.error_avg_Mn_minus_NaN;
    error_table{o+1,4}=error_.error_avg_Mw_minus_NaN;
    error_table{o+1,5}=model_selected;
    error_table{o+1,6}=balance_selected;
end

[model_n,ODE_set_n]=tag_options(model_selected,balance_selected);
table_name=strcat(table_name,'_',model_n,'_',num2str(ODE_set_n),'.csv');
table_name=fullfile(strcat(pwd,'\error'),table_name);
fid = fopen(table_name, 'w') ;

[rows,~]=size(error_table);
for i=1:rows
    fprintf(fid,'%s,',error_table{i,1:end-1});
    fprintf(fid,'%s\n',error_table{i,end});
```

```
end
fclose(fid);
disp('Files generated successfully.')

end


function [model_n,ODE_set_n]=tag_options(model_selected,balance_selected)
% Formats files identifiers.

    switch model_selected
        case 'CCS_1983'
            model_n='CCS';
        case 'Ray_Saraf_Gupta'
            model_n='RSG';
        case 'Seth_Gupta'
            model_n='SG';
        case 'Curteanu_Bulacovschi'
            model_n='CB';
        case 'Sangwai_et_al'
            model_n='S_et_al';
        case 'AI01'
            model_n='AI01';
        case 'AI02'
            model_n='AI02';
    end

    switch balance_selected
        case 'no_kfm_no_ktc_with_feedtanks'
            ODE_set_n=4;
        case 'kfm_ktc_feedtanks_depropagation'
            ODE_set_n=5;
        case 'no_ktc_with_feedtanks'
            ODE_set_n=6;
    end
end
```

polymerize.m:

```
function [t,y,error_]=polymerize(experiment,parameters_input,option,flag_optimizer)

counter_f=1; % Initializes counter
constants_table={'time_min','conversion_reactor','kp','log kp','kt','log kt','ktc',...
    'ktd','kfm','kd','conversion_global','f_I','T','P_n','kde'};
free_volume_table={'phi_m','phi_s','phi_p','V_fm','V_fs','V_fp','psi','V_m_star',...
    'V_s_star','V_p_star','psi_ref','Conversion_global','Time'};
derivatives_table={'dnMdt','dnIdt','dnSdt','dnCTAdt','dnY0dt','dnY1dt','dnY2dt',...
    'dnQ0dt','dnQ1dt','dnQ2dt','dTdt'};

% Sub vector initial values:

    y0(1)=0;
    y0(2)=experiment.monomer_content_mols;
    y0(3)=experiment.initiator_content_mols;
    y0(4)=experiment.solvent_content_mols;
    y0(5)=experiment.CTA_content_mols;
    y0(6)=realmin('double');
    y0(7)=realmin('double');
    y0(8)=realmin('double');
    y0(9)=realmin('double');
    y0(10)=realmin('double');
    y0(11)=realmin('double');
    y0(12)=experiment.temperature;
    y0(13)=experiment.monomer_feed_mols;
    y0(14)=experiment.initiator_feed_mols;
    y0(15)=experiment.monomer_content_mols;
    y0(16)=experiment.solvent_content_mols;
    y0(17)=experiment.solvent_feed_mols;
    y0(18)=experiment.CTA_content_mols;
    y0(19)=experiment.CTA_feed_mols;

% Integration with ODE15s:
    options = odeset('RelTol',1e-9);
    [t,y]=ode15s(@ODEfun,experiment.timespan_min,y0,options);

function dYfuncvecdt = ODEfun(t,Yfuncvec)

    % Load parameters
    ODE_eqns=experiment.ODE_eqns;
    MM_M=parameters_input.monomer.MW;
```

```
    % Solve ODEs for time step:
    [none_,dnMdt,dnIdt,dnSdt,dnCTAdt,dnY0dt,dnY1dt,dnY2dt,dnQ0dt,dnQ1dt,dnQ2dt,dTdt,...
        dFT_nMdt,dFT_nIdt,dnM_frdt,dnS_frdt,dFT_nSdt,dnCTA_frdt,dFT_nCTAdt,...
        kde,kp,kt,ktc,ktd,kfm,kd,kf,nM,nM_fr,FT_nM,f_I,V_r,T,nS_fr,nCTA,nY0,FVparam] = ...
        balances(experiment,parameters_input,Yfuncvec,ODE_eqns,t,flag_optimizer);

    % Store calculated constants in constants_table:
    counter_f=counter_f+1;
    constants_table{counter_f,1}=t;
    constants_table{counter_f,2}=1-nM/nM_fr;
    constants_table{counter_f,3}=kp;
    constants_table{counter_f,4}=log10(kp);
    constants_table{counter_f,5}=kt;
    constants_table{counter_f,6}=log10(kt);
    constants_table{counter_f,7}=ktc;
    constants_table{counter_f,8}=ktd;
    constants_table{counter_f,9}=kfm;
    constants_table{counter_f,10}=kd;
    constants_table{counter_f,11}=1-(nM+FT_nM)/(experiment.monomer_content_mols+...
        experiment.monomer_feed_mols);
    constants_table{counter_f,12}=f_I;
    if isempty(experiment.T_exp)
        constants_table{counter_f,13}=T;
    else
        constants_table{counter_f,13}=experiment.T_exp(t);
    end
    constants_table{counter_f,14}=(kp-kde/nM)*nM*nY0/(ktc*nY0^2+2*ktd*nY0^2+kfm*nM*...
        nY0+kf*nCTA*nY0);
    constants_table{counter_f,15}=kde;

    % Store calculated constants in free_volume_table:
    free_volume_table{counter_f,1}=FVparam.phi_m;
    free_volume_table{counter_f,2}=FVparam.phi_s;
    if free_volume_table{counter_f,2} <1e-10
        free_volume_table{counter_f,2}=0;
    end
    free_volume_table{counter_f,3}=FVparam.phi_p;
    free_volume_table{counter_f,4}=FVparam.V_fm;
    free_volume_table{counter_f,5}=FVparam.V_fs;
    free_volume_table{counter_f,6}=FVparam.V_fp;
    free_volume_table{counter_f,7}=FVparam.psi;
    free_volume_table{counter_f,8}=FVparam.V_m_star;
    free_volume_table{counter_f,9}=FVparam.V_s_star;
    free_volume_table{counter_f,10}=FVparam.V_p_star;
    free_volume_table{counter_f,11}=FVparam.psi_ref;
    free_volume_table{counter_f,12}=constants_table{counter_f,11};
    free_volume_table{counter_f,13}=constants_table{counter_f,1};

    % Store derivatives in derivatives_table:
    derivatives_table{counter_f,1}=dnMdt;
    derivatives_table{counter_f,2}=dnIdt;
    derivatives_table{counter_f,3}=dnSdt;
    derivatives_table{counter_f,4}=dnCTAdt;
    derivatives_table{counter_f,5}=dnY0dt;
    derivatives_table{counter_f,6}=dnY1dt;
    derivatives_table{counter_f,7}=dnY2dt;
    derivatives_table{counter_f,8}=dnQ0dt;
    derivatives_table{counter_f,9}=dnQ1dt;
    derivatives_table{counter_f,10}=dnQ2dt;
    derivatives_table{counter_f,11}=dTdt;

    dYfuncvecdt = [none_;dnMdt;dnIdt;dnSdt;dnCTAdt;dnY0dt;dnY1dt;dnY2dt;dnQ0dt;dnQ1dt;...
        dnQ2dt;dTdt;dFT_nMdt;dFT_nIdt;dnM_frdt;dnS_frdt;dFT_nSdt;dnCTA_frdt;dFT_nCTAdt];
end

% Creates csv files with results of the model and plot data

error_=data_format(y,t,MM_M,constants_table,free_volume_table,derivatives_table,...
    experiment,parameters_input,option);

if flag_optimizer==0||~isempty(experiment.flag_plot)
    data_plot([],option,experiment);
end

% Compute execution time
toc

end
```

balances.m:

```
function [none_,dnMdt,dnIdt,dnSdt,dnCTAdt,dnY0dt,dnY1dt,dnY2dt,dnQ0dt,...
    dnQ1dt,dnQ2dt,dTdt,dFT_nMdt,dFT_nIdt,dnM_frdt,dnS_frdt,dFT_nSdt,dnCTA_frdt,dFT_nCTAdt,...
    kde,kp,kt,ktc,ktd,kfm,kd,kf,nM,nM_fr,FT_nM,f_I,V_r,T,nS_fr,nCTA,nY0,FVparam] ...
    = balances(experiment,parameters_input,Yfuncvec,ODE_eqns,t,flag_optimizer)

%BALANCES solves the selected ODE set for the time step

% Variable instantaneous values:
none_=Yfuncvec(1);
nM=Yfuncvec(2);
nI=Yfuncvec(3);
nS=Yfuncvec(4);
nCTA=Yfuncvec(5);
nY0=Yfuncvec(6);
nY1=Yfuncvec(7);
nY2=Yfuncvec(8);
nQ0=Yfuncvec(9);
nQ1=Yfuncvec(10);
nQ2=Yfuncvec(11);
T=Yfuncvec(12);
FT_nM=Yfuncvec(13);
FT_nI=Yfuncvec(14);
nM_fr=Yfuncvec(15);
nS_fr=Yfuncvec(16);
FT_nS=Yfuncvec(17);
nCTA_fr=Yfuncvec(18);
FT_nCTA=Yfuncvec(19);

ro_p=parameters_input.monomer.ro_p;
ro_m=parameters_input.monomer.ro_m(T);
ro_i=parameters_input.initiator.ro_I;
ro_s=parameters_input.solvent.ro_s(T);
ro_f=parameters_input.CTA.ro_f;
MM_M=parameters_input.monomer.MW;
MM_I=parameters_input.initiator.MW;
MM_S=parameters_input.solvent.MW;
MM_CTA=parameters_input.CTA.MW;
fs0=experiment.fs0;

V_r=(nM*MM_M/ro_m+(nM_fr-nM)*MM_M/ro_p+nI*MM_I/ro_i+nS*MM_S/ro_s+nCTA*MM_CTA/ro_f)/1000;
f_I=parameters_input.initiator.f(fs0);
f_tr=parameters_input.CTA.f_tr;

Flow_rate_nI=experiment.initiator_F_nI;
Flow_rate_nM=experiment.monomer_F_nM;
Flow_rate_nS=experiment.solvent_F_nS;
Flow_rate_nCTA=experiment.CTA_F_nCTA;

feed_start_time_m=experiment.monomer_feed_start_time;
feed_start_time_i=experiment.initiator_feed_start_time;
feed_start_time_s=experiment.solvent_feed_start_time;
feed_start_time_f=experiment.CTA_feed_start_time;

[kde,kp,kt,ktc,ktd,kfm,kd,kf,f_I,T,FVparam] = constants_calc(experiment,parameters_input,T,...
    nY0,nY1,nQ0,nQ1,nM,nM_fr,V_r,f_I,t,nS,fs0,flag_optimizer);

switch ODE_eqns

    case 'kfm_ktc_feedtanks'

        none_ = 0;
        dnMdt = -(kp+kfm)*nM*nY0/V_r+feed_start(t,feed_start_time_m)*Flow_rate_nM*...
            heaviside(FT_nM);
        dnIdt = -kd*nI+feed_start(t,feed_start_time_i)*Flow_rate_nI*heaviside(FT_nI);
        dnSdt = feed_start(t,feed_start_time_s)*Flow_rate_nS*heaviside(FT_nS);
        dnCTAdt = -kf*nCTA*f_tr*nY0/V_r+feed_start(t,feed_start_time_f)*Flow_rate_nCTA*...
            heaviside(FT_nCTA);
        dnY0dt = 2*f_I*kd*nI - kt*nY0^2/V_r;
        dnY1dt = (kp*nM*nY0 - kt*nY0*nY1 + (kfm*nM+kf*nCTA*f_tr)*(nY0-nY1))/V_r;
        dnY2dt = (kp*nM*(2*nY1+nY0) + (kfm*nM+kf*nCTA*f_tr)*(nY0-nY2) - kt*nY2*nY0)/V_r;
        dnQ0dt = ((ktd+ktc/2)*nY0^2 + (kfm*nM+kf*nCTA*f_tr)*nY0)/V_r;
        dnQ1dt = (kt*nY0*nY1 + (kfm*nM+kf*nCTA*f_tr)*nY1)/V_r;
        dnQ2dt = (ktc*(nY0*nY2+nY1^2)+ktd*nY0*nY2+(kfm*nM+kf*nCTA*f_tr)*nY2)/V_r;
        dTdt = 0;
        dnM_frdt= feed_start(t,feed_start_time_m)*Flow_rate_nM*heaviside(FT_nM);
        dFT_nMdt= -dnM_frdt;
        dFT_nIdt= -feed_start(t,feed_start_time_i)*Flow_rate_nI*heaviside(FT_nI);
        dnS_frdt= feed_start(t,feed_start_time_s)*Flow_rate_nS*heaviside(FT_nS);
        dFT_nSdt= -dnS_frdt;
```

```
        dnCTA_frdt= feed_start(t,feed_start_time_f)*Flow_rate_nCTA*heaviside(FT_nCTA);
        dFT_nCTAdt= -dnCTA_frdt;

    case 'no_kfm_no_ktc_with_feedtanks'

        kfm=0;
        ktc=0;
        ktd=kt;
        none_ = 0;
        dnMdt = -(kp+kfm)*nM*nY0/V_r+feed_start(t,feed_start_time_m)*Flow_rate_nM*...
            heaviside(FT_nM);
        dnIdt = -kd*nI+feed_start(t,feed_start_time_i)*Flow_rate_nI*heaviside(FT_nI);
        dnSdt = feed_start(t,feed_start_time_s)*Flow_rate_nS*heaviside(FT_nS);
        dnCTAdt = -kf*nCTA*f_tr*nY0/V_r+feed_start(t,feed_start_time_f)*Flow_rate_nCTA*...
            heaviside(FT_nCTA);
        dnY0dt = 2*f_I*kd*nI - kt*nY0^2/V_r;
        dnY1dt = (kp*nM*nY0 - kt*nY0*nY1 + (kfm*nM+kf*nCTA*f_tr)*(nY0-nY1))/V_r;
        dnY2dt = (kp*nM*(2*nY1+nY0) + (kfm*nM+kf*nCTA*f_tr)*(nY0-nY2) - kt*nY2*nY0)/V_r;
        dnQ0dt = ((ktd+ktc/2)*nY0^2 + (kfm*nM+kf*nCTA*f_tr)*nY0)/V_r;
        dnQ1dt = (kt*nY0*nY1 + (kfm*nM+kf*nCTA*f_tr)*nY1)/V_r;
        dnQ2dt = (ktc*(nY0*nY2+nY1^2)+ktd*nY0*nY2+(kfm*nM+kf*nCTA*f_tr)*nY2)/V_r;
        dTdt = 0;
        dnM_frdt= feed_start(t,feed_start_time_m)*Flow_rate_nM*heaviside(FT_nM);
        dnS_frdt= feed_start(t,feed_start_time_s)*Flow_rate_nS*heaviside(FT_nS);
        dnCTA_frdt= feed_start(t,feed_start_time_f)*Flow_rate_nCTA*heaviside(FT_nCTA);
        dFT_nMdt= -dnM_frdt;
        dFT_nIdt= -feed_start(t,feed_start_time_i)*Flow_rate_nI*heaviside(FT_nI);
        dFT_nSdt= -dnS_frdt;
        dFT_nCTAdt= -dnCTA_frdt;

    case 'no_ktc_with_feedtanks'

        ktc=0;
        ktd=kt;
        none_ = 0;
        dnMdt = -(kp+kfm)*nM*nY0/V_r+feed_start(t,feed_start_time_m)*Flow_rate_nM*...
            heaviside(FT_nM);
        dnIdt = -kd*nI+feed_start(t,feed_start_time_i)*Flow_rate_nI*heaviside(FT_nI);
        dnSdt = feed_start(t,feed_start_time_s)*Flow_rate_nS*heaviside(FT_nS);
        dnCTAdt = -kf*nCTA*f_tr*nY0/V_r+feed_start(t,feed_start_time_f)*Flow_rate_nCTA*...
            heaviside(FT_nCTA);
        dnY0dt = 2*f_I*kd*nI - kt*nY0^2/V_r;
        dnY1dt = (kp*nM*nY0 - kt*nY0*nY1 + (kfm*nM+kf*nCTA*f_tr)*(nY0-nY1))/V_r;
        dnY2dt = (kp*nM*(2*nY1+nY0) + (kfm*nM+kf*nCTA*f_tr)*(nY0-nY2) - kt*nY2*nY0)/V_r;
        dnQ0dt = ((ktd+ktc/2)*nY0^2 + (kfm*nM+kf*nCTA*f_tr)*nY0)/V_r;
        dnQ1dt = (kt*nY0*nY1 + (kfm*nM+kf*nCTA*f_tr)*nY1)/V_r;
        dnQ2dt = (ktc*(nY0*nY2+nY1^2)+ktd*nY0*nY2+(kfm*nM+kf*nCTA*f_tr)*nY2)/V_r;
        dTdt = 0;
        dnM_frdt= feed_start(t,feed_start_time_m)*Flow_rate_nM*heaviside(FT_nM);
        dnS_frdt= feed_start(t,feed_start_time_s)*Flow_rate_nS*heaviside(FT_nS);
        dnCTA_frdt= feed_start(t,feed_start_time_f)*Flow_rate_nCTA*heaviside(FT_nCTA);
        dFT_nMdt= -dnM_frdt;
        dFT_nIdt= -feed_start(t,feed_start_time_i)*Flow_rate_nI*heaviside(FT_nI);
        dFT_nSdt= -dnS_frdt;
        dFT_nCTAdt= -dnCTA_frdt;

    case 'kfm_ktc_feedtanks_depropagation'

        none_ = 0;
        dnMdt = -(kp+kfm-kde/nM)*nM*nY0/V_r+feed_start(t,feed_start_time_m)*Flow_rate_nM*...
            heaviside(FT_nM);
        dnIdt = -kd*nI+feed_start(t,feed_start_time_i)*Flow_rate_nI*heaviside(FT_nI);
        dnSdt = feed_start(t,feed_start_time_s)*Flow_rate_nS*heaviside(FT_nS);
        dnCTAdt = -kf*nCTA*f_tr*nY0/V_r+feed_start(t,feed_start_time_f)*Flow_rate_nCTA*...
            heaviside(FT_nCTA);
        dnY0dt = 2*f_I*kd*nI - kt*nY0^2/V_r;
        dnY1dt = ((kp-kde/nM)*nM*nY0 - kt*nY0*nY1 + (kfm*nM+kf*nCTA*f_tr)*(nY0-nY1))/V_r;
        dnY2dt = ((kp-kde/nM)*nM*(2*nY1+nY0) + (kfm*nM+kf*nCTA*f_tr)*(nY0-nY2) - ...
            kt*nY2*nY0)/V_r;
        dnQ0dt = ((ktd+ktc/2)*nY0^2 + (kfm*nM+kf*nCTA*f_tr)*nY0)/V_r;
        dnQ1dt = (kt*nY0*nY1 + (kfm*nM+kf*nCTA*f_tr)*nY1)/V_r;
        dnQ2dt = (ktc*(nY0*nY2+nY1^2)+ktd*nY0*nY2+(kfm*nM+kf*nCTA*f_tr)*nY2)/V_r;
        dTdt = 0;
        dnM_frdt= feed_start(t,feed_start_time_m)*Flow_rate_nM*heaviside(FT_nM);
        dnS_frdt= feed_start(t,feed_start_time_s)*Flow_rate_nS*heaviside(FT_nS);
        dnCTA_frdt= feed_start(t,feed_start_time_f)*Flow_rate_nCTA*heaviside(FT_nCTA);
        dFT_nMdt= -dnM_frdt;
        dFT_nIdt= -feed_start(t,feed_start_time_i)*Flow_rate_nI*heaviside(FT_nI);
        dFT_nSdt= -dnS_frdt;
        dFT_nCTAdt= -dnCTA_frdt;

end

end
```

```
function [flag] = feed_start(t,t0)
% provides flag for splash feed
if ~isempty(t0)
    if t>=t0
        flag=1;
    else
        flag=0;
    end
else
    flag=0;
end

end
```

constants_calc.m:

```
function [kde,kp,kt,ktc,ktd,kfm,kd,kf,f_I,T,FVparam] = constants_calc(experiment,...
    parameters_input,T,nY0,nY1,nQ0,nQ1,nM,nM_fr,V_r,f_I,t,nS,fs0,flag_optimizer);
% constants_calc calculates the reaction constants based on selected model.

if ~isempty(experiment.T_exp)
    T=experiment.T_exp(t);
end

% Parameters loading
ro_m=parameters_input.monomer.ro_m(T);
ro_p=parameters_input.monomer.ro_p;
ro_s=parameters_input.solvent.ro_s(T);
M_MW=parameters_input.monomer.MW;
S_MW=parameters_input.solvent.MW;
phi_m=(nM*M_MW/ro_m)/(nM*M_MW/ro_m+(nM_fr-nM)*M_MW/ro_p+nS*S_MW/ro_s);
phi_s=(nS*S_MW/ro_s)/(nM*M_MW/ro_m+(nM_fr-nM)*M_MW/ro_p+nS*S_MW/ro_s);
phi_p=1-phi_m-phi_s;

% Calculation of kd
kd=parameters_input.initiator.kd(T);

switch experiment.model

    case 'AI01'

        gamma_optm=0.9956;
        [xi_13,psi,psi_ref,FVparam]=FV_param(T,nM_fr,nM,nS,parameters_input,gamma_optm);
        theta_t=@(T)10^(1.2432e2-1.0282e5*(1/T)+2.2589e7*(1/(T^2)));
        theta_p=@(T)10^(8.2054e1-7.7342e4*(1/T)+1.8149e7*(1/(T^2)));
        if strcmp(experiment.initiator,'AIBN')
            theta_f=@(T)10^(1.9908e2-1.4883e5*(1/T)+2.6903e7*(1/(T^2)));
        elseif strcmp(experiment.initiator,'BPO')
            theta_f=@(T)10^(-3.763e1+1.686e4*(1/T));
        end
        fs0=experiment.fs0;
        f0=parameters_input.initiator.f(fs0);
        f_I=1/(1/f0*(1+theta_f(T+273.15)*(1-phi_p)*(1/(exp(xi_13*(-psi+psi_ref))))));
        [kp,kt]=gupta_consts(xi_13,psi,psi_ref,theta_p,theta_t,T,nY0,nY1,nQ0,nQ1,V_r);

    case 'AI02'

        [~,~,~,FVparam]=FV_param(T,nM_fr,nM,nS,parameters_input);
        X=(nM_fr-nM)*M_MW/(nM_fr*M_MW+nS*S_MW);
        [kt0,kp0]=const0_calc(T);

        A11=-2.3085e-2; A12=8.0622e-1; A21=-2.3350e-1; A22=9.0358;
        A31=1.4003e-1; A32=-3.5822e1; A41=-1.7502e-3; A42=2.2250;
        B11=-4.020e-3; B12=1.246e-1; B21=-3.054e-1; B22=9.734;
        B31=2.706e-1; B32=8.278; B41=2.493e-1; B42=-4.459e1;

        A1=A11*(T)+A12; A2=A21*(T)+A22; A3=A31*(T)+A32; A4=A41*(T)+A42;
        B1=B11*(T)+B12; B2=B21*(T)+B22; B3=B31*(T)+B32; B4=B41*(T)+B42;

        kt=(kt0*exp(A1+A2*X+A3*X^2+A4*X^3))*60*1000;
        kp=(kp0*exp(B1+B2*X+B3*X^2+B4*X^3))*60*1000;

    case 'CCS_1983' % Model based on Chiu, Carrant and Song, 1983

        [~,~,~,FVparam]=FV_param(T,nM_fr,nM,nS,parameters_input);
        X=1-nM/nM_fr;
        eps=(ro_m-ro_p)/ro_p;
        phi_m=(1-X)/(1+eps*X);

        A=experiment.A_CCS;
        B=experiment.B_CCS;
        theta_p=experiment.theta_p_CCS;
```

```
        theta_t=experiment.theta_t_CCS;
        [kp,kt] = CCS_calc(A,B,theta_p,theta_t,T,nY0,V_r,phi_m,parameters_input);

    case 'Ray_Saraf_Gupta' % Model based on Ray, Saraf and Gupta, 1995.

        [xi_13,psi,psi_ref,FVparam]=FV_param(T,nM_fr,nM,nS,parameters_input);
        if strcmp(experiment.thetas_calc_rule,'BFC')
            % Equations from table 4:
            theta_t=@(T)10^(1.236550697605e2-1.0277928295174e5*(1/T)+...
                2.2665992726301e7*(1/(T^2)));
            theta_p=@(T)10^(1.58323e2-1.24953e5*(1/T)+2.5596e7*(1/(T^2)));
            f_I=parameters_input.initiator.f(fs0);
        else
            % Experimental data from table 5:
            theta_t=experiment.theta_t_RSG;
            theta_p=experiment.theta_p_RSG;
            f_I=experiment.f_exp;
        end
        [kp,kt]=gupta_consts(xi_13,psi,psi_ref,theta_p,theta_t,T,nY0,nY1,nQ0,nQ1,V_r);

    case 'Seth_Gupta' % Model based on Seth and Gupta, 1995.

        [xi_13,psi,psi_ref,FVparam]=FV_param(T,nM_fr,nM,nS,parameters_input);

        theta_t=@(T)10^(1.236550697605e2-1.0277928295174e5*(1/T)+...
            2.2665992726301e7*(1/(T^2)));
        theta_p=@(T)10^(8.03e1-7.5e4*(1/T)+1.765e7*(1/(T^2)));

        [kp,kt]=gupta_consts(xi_13,psi,psi_ref,theta_p,theta_t,T,nY0,nY1,nQ0,nQ1,V_r);
        [f_I] = f_SG(parameters_input,experiment,T,nM,V_r,xi_13,psi,psi_ref);

    case 'Curteanu_Bulacovschi' % Model based on Curtaneanu and Bulacovschi, 1999

        [~,~,~,FVparam]=FV_param(T,nM_fr,nM,nS,parameters_input);
        X=(nM_fr-nM)*M_MW/(nM_fr*M_MW+nS*S_MW);
        nI0=experiment.initiator_content_mols;
        [kp,kt]=CB_SBGS(X,nI0,T,experiment.model);

    case 'Sangwai_et_al'

        [~,~,~,FVparam]=FV_param(T,nM_fr,nM,nS,parameters_input);
        X=(nM_fr-nM)*M_MW/(nM_fr*M_MW+nS*S_MW);
        nI0=experiment.initiator_content_mols;
        [kp,kt]=CB_SBGS(X,nI0,T,experiment.model);

end

% Calculation of kinetic constants
ktcperktd_funct=parameters_input.monomer.ktcperktd;
ktd=kt/ktcperktd_funct(T)*1/(1+1/ktcperktd_funct(T));
ktc=kt-ktd;
kfm=kp*parameters_input.monomer.kfmperkp(T);
kf=parameters_input.CTA.kf(T,kp);
kde=parameters_input.monomer.kde(T);

end

function [kp,kt] = CCS_calc(A,B,theta_p,theta_t,T,nY0,V_r,phi_m,parameters_input)
% Model based on Chiu, Carrant and Song, 1983

kp0=parameters_input.monomer.kp0(T);
kp=1/(1/kp0 + theta_p*nY0/V_r/(exp(2.3*phi_m/(A+B*phi_m))));
kt0=parameters_input.monomer.kt0(T);
kt=1/(1/kt0 + theta_t*nY0/V_r/(exp(2.3*phi_m/(A+B*phi_m))));
end

function [xi_13,psi,psi_ref,FVparam]=FV_param(T,nM_fr,nM,nS,parameters_input,varargin)
% Model based on Ray, Saraf, Gupta, 1995

parameters_input.monomer.V_m_star;
parameters_input.monomer.V_p_star;
V_s_star=parameters_input.solvent.V_s_star;
V_m_star=parameters_input.monomer.V_m_star;
V_p_star=parameters_input.monomer.V_p_star;
M_jp=parameters_input.monomer.M_jp;
V_fm=parameters_input.monomer.V_fm(T);
V_fp=parameters_input.monomer.V_fp(T);
V_fs=parameters_input.solvent.V_fs(T);

if ~isempty(varargin)
    gamma_=varargin{1}(1,1);
else
    gamma_=1;
```

```
end

M_MW=parameters_input.monomer.MW/1000;
S_MW=parameters_input.solvent.MW/1000;

ro_m=(parameters_input.monomer.ro_m(T))*1000;
ro_s=(parameters_input.solvent.ro_s(T))*1000;
ro_p=parameters_input.monomer.ro_p*1000;

xi_13=V_m_star*M_MW/(V_p_star*M_jp);
xi_23=V_s_star*S_MW/(V_p_star*M_jp);

psi_ref=gamma_/V_fp;
phi_m=(nM*M_MW/ro_m)/(nM*M_MW/ro_m+(nM_fr-nM)*M_MW/ro_p+nS*S_MW/ro_s);
phi_s=(nS*S_MW/ro_s)/(nM*M_MW/ro_m+(nM_fr-nM)*M_MW/ro_p+nS*S_MW/ro_s);
phi_p=1-phi_m-phi_s;

psi=gamma_*(ro_m*phi_m*V_m_star/xi_13 + ro_s*phi_s*V_s_star/xi_23 + ...
    ro_p*phi_p*V_p_star)/(ro_m*phi_m*V_m_star*V_fm + ro_s*phi_s*V_s_star*V_fs +...
    ro_p*phi_p*V_p_star*V_fp);

FVparam.phi_m=phi_m;
FVparam.phi_s=phi_s;
FVparam.phi_p=phi_p;
FVparam.V_fm=V_fm;
FVparam.V_fs=V_fs;
FVparam.V_fp=V_fp;
FVparam.V_m_star=V_m_star;
FVparam.V_s_star=V_s_star;
FVparam.V_p_star=V_p_star;
FVparam.psi=psi;
FVparam.psi_ref=psi_ref;

end

function[kp,kt]=gupta_consts(xi_13,psi,psi_ref,theta_p,theta_t,T,nY0,nY1,nQ0,nQ1,V_r)
% Model based on Ray, Saraf, Gupta, 1995

[kt0,kp0]=const0_calc(T);

kt=(1/(1/kt0+theta_t(T+273.15)*(((nQ1+nY1)/(nQ0+nY0))^2)*((nY0/V_r)*1000)*...
    1/(exp(-psi+psi_ref))))*60*1000;
kp=(1/(1/kp0+theta_p(T+273.15)*(nY0/V_r*1000)*1/(exp(xi_13*(-psi+psi_ref)))))*60*1000;

end

function [f_I] = f_SG(parameters_input,experiment,T,nM,V_r,xi_13,psi,psi_ref)
% Model based on Seth and Gupta, 1995
    if strcmp(experiment.initiator,'AIBN')
        theta_f=@(T)10^(2.016e2-1.455e5*(1/T)+2.7e7*(1/(T^2)));
    elseif strcmp(experiment.initiator,'BPO')
        theta_f=@(T)10^(-3.763e1+1.686e4*(1/T));
    end
    f0=parameters_input.initiator.f(0);
    f_I=1/(1/f0*(1+theta_f(T+273.15)*nM/V_r*1/(exp(xi_13*(-psi+psi_ref)))));

end

function [kp,kt]=CB_SBGS(X,nI0,T,model)
% Model based on Curtaneanu and Bulacovschi 1999

[kt0,kp0]=const0_calc(T);

switch model
    case 'Curteanu_Bulacovschi'
        [A1,A2,A3,A4,B1,B2,B3,B4]=CB_constants(nI0,T);
    case 'Sangwai_et_al'
        [A1,A2,A3,A4,B1,B2,B3,B4]=SBGS_consts(T);

end

kt=(kt0*exp(A1+A2*X+A3*X^2+A4*X^3))*60*1000;
kp=(kp0*exp(B1+B2*X+B3*X^2+B4*X^3))*60*1000;

end

function [A1,A2,A3,A4,B1,B2,B3,B4]=CB_constants(nI0,T)

% Assign constants from table I

if nI0==0.01548
    if T==50
```

```
        consts_CB = num2cell([-0.39 5,-48.32,22.20,0.11,-2.81,14.25,-18.45]);
    else
        consts_CB = num2cell([-0.23,3,-22.83,-0.95,0.50,-9.53,38.32,-39.55]);
    end
end
if nI0==0.0258
    if T==50
        consts_CB = num2cell([-0.33,5,-39.52,12.23,0.31,-6.92,31.57,-36.45]);
    elseif T==60
        consts_CB = num2cell([-0.21,3.76,-25.37,0.39,0.47,-9.43,39.58,-42.28]);
    elseif T==70
        consts_CB = num2cell([-0.39,6.31,-27.51,1.78,0.42,-8.20,33.92,-36.52]);
    else
        consts_CB = num2cell([-1.36,18.72,-56.99,22.28,-0.21,0.91,5.65,-14.78]);
    end
end

[A1,A2,A3,A4,B1,B2,B3,B4] = deal(consts_CB{:});

end

function [A1,A2,A3,A4,B1,B2,B3,B4]=SBGS_consts(T)

A11=-2.293e-2; A12=7.973e-1; A21=-2.379e-1; A22=8.784;
A31=1.385e-1; A32=-3.609e1; A41=-1.745e-3; A42=2.223;
B11=-4.020e-3; B12=1.246e-1; B21=-3.054e-1; B22=9.734;
B31=2.706e-1; B32=8.278; B41=2.493e-1; B42=-4.459e1;

A1=A11*(T)+A12;
A2=A21*(T)+A22;
A3=A31*(T)+A32;
A4=A41*(T)+A42;
B1=B11*(T)+B12;
B2=B21*(T)+B22;
B3=B31*(T)+B32;
B4=B41*(T)+B42;

end

function [kt0,kp0]=const0_calc(T)

R=8.314e-3;
Ep=18.22;
Etd=2.937;
Et=Etd;
kp0=4.917e2*exp(-Ep/(R*(T+273.15)));
kt0=9.8e4*exp(-Et/(R*(T+273.15)));

end
```

data_format.m:

```
function [error_] =
data_format(y,t,MM_M,constants_table,free_volume_table,derivatives_table,experiment,parameters_input,
option)
%DATA FORMAT Creates csv files with results of the model

option=num2str(option);
ro_m=parameters_input.monomer.ro_m;
ro_p=parameters_input.monomer.ro_p;
ro_s=parameters_input.solvent.ro_s;
ro_I=parameters_input.initiator.ro_I;
ro_f=parameters_input.CTA.ro_f;
MM_I=parameters_input.initiator.MW;
MM_S=parameters_input.solvent.MW;
MM_CTA=parameters_input.CTA.MW;

% Format calculated data to csv file:
[rown,coln]=size(y);

model_results={'None','[M]','[I]','[S]','[CTA]','[Y0]','[Y1]','[Y2]','[Q0]','[Q1]',...
    '[Q2]','Temp (°C)','nM_feedtank','nI_feedtank','nM_fr','nS_fr','FT_nS','nCTA_fr',...
    'nCTA_feedtank','X_reactor','X_global','Volume','t'};
for i=1:rown
    for j=1:coln
        model_results{i+1,j}=y(i,j);
    end
    for k=1:rown
       model_results{k+1,23}=t(k,1);
    end
    if ~isempty(experiment.T_exp)
```

```
        for m=1:rown
            model_results{m+1,12}=experiment.T_exp((model_results{m+1,23}));
        end
    end

    model_results{i+1,20}=1-model_results{i+1,2}/model_results{i+1,15};
    model_results{i+1,21}=1-(model_results{i+1,2}+model_results{i+1,13})/...
        (experiment.monomer_content_mols+experiment.monomer_feed_mols);
    model_results{i+1,22}=(model_results{i+1,2}*MM_M/ro_m(model_results{i+1,12})+...
        (model_results{i+1,15}-model_results{i+1,2})*MM_M/ro_p+model_results{i+1,3}*...
        MM_I/ro_I+model_results{i+1,4}*MM_S/ro_s(model_results{i+1,12})+...
        model_results{i+1,5}*MM_CTA/ro_f)/1000;


    if model_results{i+1,4}<1e-9
        model_results{i+1,4}=0;
    end
    if model_results{i+1,5}<1e-9
        model_results{i+1,5}=0;
    end
end

for i=2:rown
    for j=2:11
        model_results{i,j}=model_results{i,j}/model_results{i,22};
    end
end

% Calculate MW results:
MW_results={'r_n','r_w','M_n','M_w','PDI','X_reactor','X_global'};

for i=2:size(model_results)
    MW_results{i,1}=model_results{i,10}/model_results{i,9};
    MW_results{i,2}=model_results{i,11}/model_results{i,10};
    MW_results{i,3}=MW_results{i,1}*MM_M;
    MW_results{i,4}=MW_results{i,2}*MM_M;
    MW_results{i,5}=MW_results{i,2}/MW_results{i,1};
end

% Copying conversion 'X' from model_results to MW_results:
for i=1:rown
    MW_results{i+1,6}=model_results{i+1,16};
    MW_results{i+1,7}=model_results{i+1,21};
end

% Save model results:
name1=strcat('model_results_',option,'.csv');
name1=fullfile(strcat(pwd,'\CSV'),name1);
fid = fopen(name1, 'w');
fprintf(fid, '%s,', model_results{1,1:end-1}) ;
fprintf(fid, '%s\n', model_results{1,end}) ;
fclose(fid) ;
dlmwrite(name1, model_results(2:end,:), '-append') ;

% Save derivatives:
name8=strcat('derivatives_',option,'.csv');
name8=fullfile(strcat(pwd,'\CSV'),name8);
fid = fopen(name8, 'w');
fprintf(fid, '%s,', derivatives_table{1,1:end-1}) ;
fprintf(fid, '%s\n', derivatives_table{1,end}) ;
fclose(fid) ;
dlmwrite(name8, derivatives_table(2:end,:), '-append') ;

% Save MW results:
name2=strcat('MW_results_',option,'.csv');
name2=fullfile(strcat(pwd,'\CSV'),name2);
fid = fopen(name2, 'w') ;
fprintf(fid, '%s,', MW_results{1,1:end-1}) ;
fprintf(fid, '%s\n', MW_results{1,end}) ;
fclose(fid) ;
dlmwrite(name2, MW_results(2:end,:), '-append') ;

% Save constant tables in a file:
table_formatted_ = clean_data_table(constants_table,1);
table_formatted = clean_data_table(table_formatted_,11);
constants_table=table_formatted;

name3=strcat('constants_table_',option,'.csv');
name3=fullfile(strcat(pwd,'\CSV'),name3);
fid = fopen(name3, 'w') ;
fprintf(fid, '%s,', constants_table{1,1:end-1}) ;
fprintf(fid, '%s\n', constants_table{1,end}) ;
fclose(fid) ;
dlmwrite(name3, constants_table(2:end,:), '-append') ;
```

```
name7=strcat('free_volume_table_',option,'.csv');
name7=fullfile(strcat(pwd,'\CSV'),name7);
fid = fopen(name7, 'w') ;
fprintf(fid, '%s,', free_volume_table{1,1:end-1}) ;
fprintf(fid, '%s\n', free_volume_table{1,end}) ;
fclose(fid) ;
dlmwrite(name7, free_volume_table(2:end,:), '-append') ;

names.model_data1=name1;
names.model_data2=name2;
names.model_data3=name3;
names.model_data7=name7;

% Create datafit_check _ Conversion
datafit_check={'Time_X','X_expt','X_model','E_X','empty','empty','E_X_weighted'};

[rown,~]=size(experiment.Conv_x');
expt_data_x=experiment.Conv_x';
expt_data_y=experiment.Conv_y';

table_formatted_ = clean_data_table(constants_table,1);
table_formatted = clean_data_table(table_formatted_,11);

x=table_formatted(2:end,1);
y=table_formatted(2:end,11);

x_=cell2mat(x);
y_=cell2mat(y);
a=[x_,y_];
b=unique(a,'rows');
x=b(:,1);
y=b(:,2);

count_datafit=0;
for d=1:rown
    try
        count_datafit=count_datafit+0.2;
        datafit_check{d+1,1}=expt_data_x(d,1);
        datafit_check{d+1,2}=expt_data_y(d,1);
        datafit_check{d+1,3}=interp1(x,y,datafit_check{d+1,1});
        datafit_check{d+1,4}=((datafit_check{d+1,2}-datafit_check{d+1,3})/...
            datafit_check{d+1,2})^2;
        datafit_check{d+1,5}=0;
        datafit_check{d+1,6}=0;
        datafit_check{d+1,7}=datafit_check{d+1,4}*(1+count_datafit);
    catch
        datafit_check{d+1,4}=NaN;
    end

end

error_.error_total_conversion=sum([datafit_check{2:end,4}]);
error_.error_total_conversion_weighted=sum([datafit_check{2:end,7}]);
error_.error_total_conversion_minus_NaN=0;

Count_NaN=0;
[rows,~]=size(datafit_check);

for d=2:rows
    if ~isnan(datafit_check{d,4})
        Count_NaN=Count_NaN+1;
        error_.error_total_conversion_minus_NaN=datafit_check{d,4}+...
            error_.error_total_conversion_minus_NaN;
    end
end
error_.error_avg_conversion_minus_NaN=error_.error_total_conversion_minus_NaN/Count_NaN;

name4=strcat('datafit_check_conv_',option,'.csv');
name4=fullfile(strcat(pwd,'\datafit'),name4);
fid = fopen(name4, 'w') ;
fprintf(fid, '%s,', datafit_check{1,1:end-1}) ;
fprintf(fid, '%s\n', datafit_check{1,end}) ;
fclose(fid) ;
dlmwrite(name4, datafit_check(2:end,:), '-append') ;


% Create datafit_check _ Mn
datafit_check_Mn={'Conversion_Mn','Mn_expt','Mn_model','E_Mn'};
[rown,~]=size(experiment.Mn_x');
expt_data_x=experiment.Mn_x';
expt_data_y=experiment.Mn_y';
```

```
try
    x=MW_results(3:end,7);
    y=MW_results(3:end,3);
    x_=cell2mat(x);
    y_=cell2mat(y);
    a=[x_,y_];
    b=unique(a,'rows');
    x=b(:,1);
    y=b(:,2);

    for i=1:rown
            datafit_check_Mn{i+1,1}=expt_data_x(i,1);
            datafit_check_Mn{i+1,2}=expt_data_y(i,1);
            try
                datafit_check_Mn{i+1,3}=interp1(x,y,datafit_check_Mn{i+1,1});
                datafit_check_Mn{i+1,4}=((datafit_check_Mn{i+1,2}-...
                    datafit_check_Mn{i+1,3})/datafit_check_Mn{i+1,2})^2;
            catch
                datafit_check_Mn{i+1,3}=NaN;
                datafit_check_Mn{i+1,4}=NaN;
            end
    end
catch
    datafit_check_Mn{i+1,4}=NaN;
end

error_.error_total_Mn=sum([datafit_check_Mn{2:end,4}]);
error_.error_total_Mn_minus_NaN=0;
Count_NaN=0;
[rows,~]=size(datafit_check_Mn);

for d=2:rows
    if ~isnan(datafit_check_Mn{d,4})
        Count_NaN=Count_NaN+1;
        error_.error_total_Mn_minus_NaN=datafit_check_Mn{d,4}+...
            error_.error_total_Mn_minus_NaN;
    end
end

error_.error_avg_Mn_minus_NaN=error_.error_total_Mn_minus_NaN/Count_NaN;

name5=strcat('datafit_check_Mn_',option,'.csv');
name5=fullfile(strcat(pwd,'\datafit'),name5);
fid = fopen(name5, 'w') ;
fprintf(fid, '%s,', datafit_check_Mn{1,1:end-1}) ;
fprintf(fid, '%s\n', datafit_check_Mn{1,end}) ;
fclose(fid) ;

try
    dlmwrite(name5, datafit_check_Mn(2:end,:), '-append') ;
catch
   msg=strcat('File ',name5,' could not be generated');
   disp(msg);
end

% Create datafit_check _ Mw
datafit_check_Mw={'Conversion_Mw','Mw_expt','Mw_model','E_Mw'};

try
    [rown,~]=size(experiment.Mw_x');
    expt_data_x=experiment.Mw_x';
    expt_data_y=experiment.Mw_y';

    x=MW_results(3:end,7);
    y=MW_results(3:end,4);
    x_=cell2mat(x);
    y_=cell2mat(y);
    a=[x_,y_];
    b=unique(a,'rows');
    x=b(:,1);
    y=b(:,2);

for i=1:rown
    datafit_check_Mw{i+1,1}=expt_data_x(i,1);
    datafit_check_Mw{i+1,2}=expt_data_y(i,1);
    try
        datafit_check_Mw{i+1,3}=interp1(x,y,datafit_check_Mw{i+1,1});
        datafit_check_Mw{i+1,4}=((datafit_check_Mw{i+1,2}-...
            datafit_check_Mw{i+1,3})/datafit_check_Mw{i+1,2})^2;
    catch
        datafit_check_Mn{i+1,3}='error';
        datafit_check_Mn{i+1,4}='error';
    end

end
```

```
catch
    datafit_check_Mn{i+1,4}=NaN;
end

error_.error_total_Mw=sum([datafit_check_Mw{2:end,4}]);
error_.error_total_Mw_minus_NaN=0;
Count_NaN=0;
[rows,~]=size(datafit_check_Mw);
for d=2:rows
    if ~isnan(datafit_check_Mw{d,4})
        Count_NaN=Count_NaN+1;
        error_.error_total_Mw_minus_NaN=datafit_check_Mw{d,4}+...
            error_.error_total_Mw_minus_NaN;
    end
end
error_.error_avg_Mw_minus_NaN=error_.error_total_Mw_minus_NaN/Count_NaN;

name6=strcat('datafit_check_Mw_',option,'.csv');
name6=fullfile(strcat(pwd,'\datafit'),name6);
fid = fopen(name6, 'w') ;
fprintf(fid, '%s,', datafit_check_Mw{1,1:end-1}) ;
fprintf(fid, '%s\n', datafit_check_Mw{1,end}) ;
fclose(fid) ;
try
    dlmwrite(name6, datafit_check_Mw(2:end,:), '-append') ;
catch
    msg=strcat('File ',name6,' could not be generated');
    disp(msg);
end
end

function [table_formatted] = clean_data_table(table,k)
[nrow,~]=size(table);

table_formatted_(1,:)=table(1,:);
table_formatted(1,:)=table(1,:);
table_formatted_(nrow,:)=table(nrow,:);
j=1;

for i=nrow:-1:3
    if table{i-1,k}~=table{i,k}
        if table{i-1,k}<table{i,k}
            table_formatted_(nrow-j,:)=table(i-1,:);
            j=j+1;

        end
    end
end

j=2;
for i=2:nrow
    if ~isempty(table_formatted_{i,1})
        table_formatted(j,:)=table_formatted_(i,:);
        j=j+1;
    end
end

end
```

data_plot.m:

```
function [] = data_plot(vec_expt,option,experiment)
%DATA_PLOT plots data calculated in the model

if isempty(vec_expt)
    n_expt=1;
else
    n_expt=size(vec_expt,2);
end

% Model data plot
figure('Name','Integration variables as function of conversion');
limits_table=getlimitstable();

for k=1:n_expt
    [names,experiment]=expt_data_select(vec_expt,k,option);
    model_data=importdata(names.model_data1);
    model_data_T.data=table2array(readtable(names.model_data3));
    [limits_table]=model_data_plot(model_data,model_data_T,k,n_expt,limits_table);
```

```
end

% MW data plot
figure('Name','MW distribution as function of conversion')
limits_table=getlimitstable();

for k=1:n_expt
    [names,experiment]=expt_data_select(vec_expt,k,option);
    model_data2.data=table2array(readtable(names.model_data2));
    model_data2.colheaders={'r_n'  'r_w'  'M_n'  'M_w'  'PDI'  'X_reactor'  'X_global'};
    [limits_table]=MW_data_plot(experiment,k,n_expt,model_data2,limits_table);
end

% Constants plot
figure('Name','(Pseudo)constants as function of conversion');
limits_table=getlimitstable();

for k=1:n_expt
    [names,experiment]=expt_data_select(vec_expt,k,option);
    model_data3.data=table2array(readtable(names.model_data3));
    model_data3.colheaders={'time_min'  'conversion_reactor'  'kp'  'log kp'...
        'kt'  'log kt'  'ktc'  'ktd'  'kfm'  'kd'  'conversion_global'  'f_I'...
        'T'  'P_n'  'kde'};
    [limits_table]=constants_data_plot(k,n_expt,model_data3,limits_table);
end

% Temperature plot
if ~isempty(experiment.T_exp)
    figure('Name','Temperature as function of time');
    plot(model_data3.data(:,1),model_data3.data(:,13));
    title('Temperature as function of time');
    xlabel('Time (min)') ;
    ylabel('Temperature (°C)');
end

% Free volume plot
figure('Name','Free Volume Parameters as function of time');
limits_table=getlimitstable();

for k=1:n_expt
    [names,~]=expt_data_select(vec_expt,k,option);
    model_data7=importdata(names.model_data7);
    [limits_table]=free_volume_data_plot(model_data7,limits_table);
end

% Conversion plot
figure('Name','Conversion as function of time');

for k=1:n_expt

[names,experiment]=expt_data_select(vec_expt,k,option);
model_data3.data=table2array(readtable(names.model_data3));
model_data3.colheaders={'time_min'  'conversion_reactor'  'kp'  'log kp'  'kt'...
    'log kt'  'ktc'  'ktd'  'kfm'  'kd'  'conversion_global'  'f_I'  'T'  'P_n'  'kde'};
conversion_data_plot(model_data3,experiment,k,n_expt);

end

end

function [limits_table]=model_data_plot(model_data,model_data_T,k,n_expt,limits_table)

for i=1:12
    if i==1
        subplot(4,3,1);
        x=model_data.data(:,21);
        y=model_data.data(:,22);
        hold on
        plot(x,y);
        [limits_table]=format_axis(limits_table,i,model_data,21,22,0,0);
        title(model_data.textdata(1,22));
    elseif i==12
            subplot(4,3,12);
            x=model_data_T.data(:,11);
            y=model_data_T.data(:,13);
            hold on
            plot(x,y);
            [limits_table]=format_axis(limits_table,i,model_data_T,11,13,0,0);
            title(model_data.textdata(1,i));
    else
        subplot(4,3,i);
        x=model_data.data(:,21);
        y=model_data.data(:,i);
        hold on
```

```
        plot(x,y);
        [limits_table]=format_axis(limits_table,i,model_data,21,i,0,0);
        title(model_data.textdata(1,i));
    end

    if k==n_expt
        hold off
    end
end
end

function [limits_table]=MW_data_plot(experiment,k,n_expt,model_data2,limits_table)

for i=1:4
    subplot(3,2,i);
    x=model_data2.data(2:end,7);
    y=model_data2.data(2:end,i);
    plot(x,y);
    [limits_table]=format_axis(limits_table,i,model_data2,7,i,1,0);
    hold on

    % Plot experimental data:
    if i==3
        if ~isempty(experiment.Mn_x)
            plot(experiment.Mn_x,experiment.Mn_y,'r*');
            model_expt(:,1)=experiment.Mn_x';
            model_expt(:,2)=experiment.Mn_y';
            [limits_table]=format_axis(limits_table,i,model_expt,1,2,0,0);
        end

        if ~isempty(experiment.Mn_x2)
            model_expt=[];
            plot(experiment.Mn_x2,experiment.Mn_y2,'yd');
            model_expt(:,1)=experiment.Mn_x2';
            model_expt(:,2)=experiment.Mn_y2';
            [limits_table]=format_axis(limits_table,i,model_expt,1,2,0,0);
        end

    end

    if i==4
        if ~isempty(experiment.Mw_x)
            model_expt=[];
            plot(experiment.Mw_x,experiment.Mw_y,'r*');
            model_expt(:,1)=experiment.Mw_x';
            model_expt(:,2)=experiment.Mw_y';
            [limits_table]=format_axis(limits_table,i,model_expt,1,2,0,0);

        end
        if ~isempty(experiment.Mw_x2)
            model_expt=[];
            plot(experiment.Mw_x2,experiment.Mw_y2,'yd');
            model_expt(:,1)=experiment.Mw_x';
            model_expt(:,2)=experiment.Mw_y';
            [limits_table]=format_axis(limits_table,i,model_expt,1,2,0,0);
        end
    end

    title(model_data2.colheaders(1,i));
    if k==n_expt
        hold off
    end
end

subplot(3,2,[5 6]);
hold on
x=model_data2.data(:,7);
y=model_data2.data(:,5);
model_expt=[];
model_expt(:,1)=x;
model_expt(:,2)=y;
plot(x,y);
[limits_table]=format_axis(limits_table,5,model_expt,1,2,0,0);


if k==n_expt
    hold off
end
title(model_data2.colheaders(1,5));

end

function [limits_table]=constants_data_plot(k,n_expt,model_data3,limits_table)
```

```
for i=3:12

    if i==12
        i=14;
    end

    if i==11
        i=12; % quick fix to include plot of f
    end

    if i==12
        subplot(2,5,9); % quick fix to include plot of f
    elseif i==14
        subplot(2,5,10); % quick fix to include plot of P_n
    else
        subplot(2,5,i-2);
    end

    x=model_data3.data(:,11);
    y=real(model_data3.data(:,i));
    plot(x,y);
    [limits_table]=format_axis(limits_table,i,model_data3,11,i,0,0);
    model_data3.colheader={'time_min'  'conversion_reactor'  'kp'  'log kp'  'kt'  'log kt'...
        'ktc'  'ktd'  'kfm'  'kd'  'conversion_global'  'f_I'  'T'  'P_n'  'kde'};
    title(model_data3.colheader(1,i));
    hold on

    if k==n_expt
        hold off
    end

end

end

function [limits_table]=free_volume_data_plot(model_data7,limits_table)

for f=1:8
    subplot(3,3,f);
    x=model_data7.data(:,13);
    y=model_data7.data(:,f);
    plot(x,y);
    [limits_table]=format_axis(limits_table,f,model_data7,13,f,0,0);
    model_data7.colheader=model_data7.textdata;
    title(model_data7.colheader(1,f));
end

end

function []=conversion_data_plot(model_data3,experiment,k,n_expt)
% Plot conversion data (model and experimental)

x=model_data3.data(:,2);
y=model_data3.data(:,11);
plot(model_data3.data(2:end,1),model_data3.data(2:end,11));

hold on
if ~isempty(experiment.Conv_x)
    scatter(experiment.Conv_x,experiment.Conv_y);
end

if ~isempty(experiment.Conv_x2)
    scatter(experiment.Conv_x2,experiment.Conv_y2,'d');
end

title('Conversion as function of time');
xlabel('Time (min)')
ylabel('Conversion')
if k==n_expt
    hold off
end
end

function [names,experiment]=expt_data_select(vec_expt,k,option)

if isempty(vec_expt)
    vec_expt=option;
end
names.model_data1=strcat('model_results_',num2str(vec_expt(1,k)),'.csv');
names.model_data1=fullfile(strcat(pwd,'\CSV'),names.model_data1);
names.model_data2=strcat('MW_results_',num2str(vec_expt(1,k)),'.csv');
names.model_data2=fullfile(strcat(pwd,'\CSV'),names.model_data2);
names.model_data3=strcat('constants_table_',num2str(vec_expt(1,k)),'.csv');
```

```
names.model_data3=fullfile(strcat(pwd,'\CSV'),names.model_data3);
names.model_data7=strcat('free_volume_table_',num2str(vec_expt(1,k)),'.csv');
names.model_data7=fullfile(strcat(pwd,'\CSV'),names.model_data7);

experiment=exp_database(vec_expt(1,k));

end

function [limits_table]=getlimitstable()

limits_table(1:20,1)=inf;
limits_table(1:20,2)=-inf;
limits_table(1:20,3)=inf;
limits_table(1:20,4)=-inf;

end

function [limits_table]=format_axis(limits_table,i,model_data,x_axis_column,...
    y_axis_column,mwflag,freevolflag)

if class(model_data)=='struct'
    limits_table(i,1)=min(limits_table(i,1),min(model_data.data(:,x_axis_column)));
    limits_table(i,2)=max(limits_table(i,2),max(model_data.data(:,x_axis_column)));
    limits_table(i,3)=min(limits_table(i,3),min(model_data.data(:,y_axis_column)));

    if mwflag==1
        nrow=size(model_data.data);
        nrow=nrow(1,1);
        cut=round(nrow*0.2);
        limits_table(i,4)=max(limits_table(i,4),max(model_data.data(cut:nrow,y_axis_column)));
    else
        limits_table(i,4)=max(limits_table(i,4),max(model_data.data(:,y_axis_column)));
    end

    xmin=limits_table(i,1);
    xmax=limits_table(i,2);
    ymin=limits_table(i,3);
    ymax=limits_table(i,4);

    if ymax==0
        ymax=inf;
    end

    if freevolflag==1
        ymax=inf;
    end

    try
        axis([xmin,xmax,ymin*0.95,ymax*1.05]);
    catch
    end
else
    limits_table(i,1)=min(limits_table(i,1),min(model_data(:,x_axis_column)));
    limits_table(i,2)=max(limits_table(i,2),max(model_data(:,x_axis_column)));
    limits_table(i,3)=min(limits_table(i,3),min(model_data(:,y_axis_column)));
    limits_table(i,4)=max(limits_table(i,4),max(model_data(:,y_axis_column)));
    xmin=limits_table(i,1);
    xmax=limits_table(i,2);
    ymin=limits_table(i,3);
    ymax=limits_table(i,4);

    if ymax==0
        ymax=inf;
    end

    if freevolflag==1
        ymax=inf;
    end
    axis([xmin,xmax,ymin*0.95,ymax*1.05]);
end
end
```

loading_inputs.m:

```
function [parameters_input,experiment]=loading_inputs(option)

% Load experiment from database:
experiment=exp_database(option);

% Load parameters from database:
```

```
parameters_input.monomer =      monomer_database(experiment.monomer);
parameters_input.initiator =    initiator_database(experiment.initiator);
if isempty(experiment.solvent)
    parameters_input.solvent =      solvent_database('Benzene');
else
    parameters_input.solvent =      solvent_database(experiment.solvent);
end

if isempty(experiment.CTA)
    parameters_input.CTA =      CTA_database('GDMA');
else
    parameters_input.CTA =      CTA_database(experiment.CTA);
end

end
```

monomer_database.m:

```
function [mon_parameters]=monomer_database(option)

switch option

    case 'MMA'

        monomer.title=      'MMA';
        monomer.MW=         100.121;
        monomer.ro_m=       @(T)(966.5 - 1.1*(T))/1000;
        monomer.ro_p=       1.2;
        monomer.kt0=        @(T)5.88e9*exp(-701/(1.987*(T+273.15)));
        monomer.kp0=        @(T)2.95e7*exp(-4353/(1.987*(T+273.15)));
        monomer.Tg_M=       106;
        monomer.Tg_p=       114;
        monomer.delta_H=    -13900;
        monomer.Cp_M=       0.4;
        monomer.Cp_p=       0.4;
        monomer.ktcperktd=  @(T)3.956e-4*exp(4.09/(1.9858775e-3*(T+273.15)));
        monomer.kfmperkp=   @(T)9.48e3*exp(-13.88/(1.9858775e-3*(T+273.15)));
        monomer.kde=        @(T)3.888e13*exp(-1.12*18253/(1.987*(T+273.15)));
        monomer.V_m_star=   8.22e-4;
        monomer.V_p_star=   7.70e-4;
        monomer.M_jp=       0.18781;
        monomer.V_fm=       @(T)0.149+2.9e-4*T;
        monomer.V_fp=       @(T)V_fp_pMMA(T);
        [mon_parameters]=   monomer;
end
end

function V_fp_pMMA = V_fp_pMMA(T)

if T<105
    V_fp_pMMA=0.0194+1.3e-4*(T-105);         % For T<Tg 105°C (Soh and Sundberg)
else
    V_fp_pMMA=0.0194+3e-4*(T-105);           % For T>=Tg 105°C (Soh and Sundberg)
end

end
```

initiator_database.m:

```
function [init_parameters]=initiator_database(option)

switch option
    case 'AIBN'
        initiator.name='AIBN';
        initiator.MW=164.21;
        initiator.f=@(fs0)0.58;
        initiator.kd=@(T)6.32e16*exp(-15.43e3/(T+273.15));
        initiator.ro_I=1.1;

    case 'BPO'
        initiator.name='BPO';
        initiator.MW=242.23;
        initiator.f=@(fs0)0.5414-0.178974*(fs0);
        initiator.kd=@(T)(1.69e14*exp(-125.4/(8.314e-3*(T+273.15))))*60;
        initiator.ro_I=1.334;
end
[init_parameters]=initiator;
end
```

solvent_database.m:

```
function [sol_parameters]=solvent_database(option)

switch option

    case 'Benzene'

        solvent.title=   'Benzene';
        solvent.MW=      78.11;
        solvent.ro_s=    @(T)(844.18-1.07165*(T+273.15-323.1))/1000;
        solvent.V_s_star=9.01e-4;
        solvent.V_fs=    @(T)0.025+1e-3*(T+273.15-171.1);

end
[sol_parameters]=solvent;
end
```

CTA_database.m:

```
function [CTA_parameters]=CTA_database(option)

switch option

    case 'n-Dodecylmercaptan'
        CTA.name='n-Dodecylmercaptan';
        CTA.MW=202.4;
        CTA.kf=@(T,kp)7.5e-5*exp(25289/(8.314*(T+273.15)))*kp;
        CTA.ro_f=0.85;
        CTA.f_tr=1;

    case 'GDMA'
        CTA.name='GDMA';
        CTA.MW=210.27;
        CTA.kftrperkp=1.26;
        CTA.kf=@(T,kp)CTA.kftrperkp*kp;
        CTA.ro_f=1.313;
        CTA.f_tr=0.961;
end
[CTA_parameters]=CTA;

end
```

exp_database.m:

```
function [experiment]=exp_database(option)
% EXP_DATABASE selects the experiment from the database

switch option

    case 3
        experiment.title=                       'CCS Model. I0=0.01548, T=70°C';
        experiment.model=                       'CCS_1983';
        experiment.timespan_min=                [0 120];            % [min]
        experiment.monomer=                     'MMA';
        experiment.monomer_content_mols=        9.366;              % [mol]
        experiment.monomer_feed_mols=           0;
        experiment.monomer_feed_start_time=     [];
        experiment.monomer_F_nM=                0;                  % [mol/min]
        experiment.initiator=                   'AIBN';
        experiment.initiator_content_mols=      0.01548;            % [mol]
        experiment.initiator_feed_mols=         0;
        experiment.initiator_feed_start_time=   [];                 % [min]
        experiment.initiator_F_nI=              0;                  % [mol/min]
        experiment.solvent=                     [];
        experiment.solvent_content_mols=        0;                  % [mol]
        experiment.solvent_feed_mols=           0;
        experiment.solvent_feed_start_time=     [];
        experiment.solvent_F_nS=                0;
        experiment.CTA=                         [];
        experiment.CTA_content_mols=            0;                  % [mol]
        experiment.CTA_feed_mols=               0;
        experiment.CTA_feed_start_time=         [];
        experiment.CTA_F_nCTA=                  0;
        experiment.temperature=                 70;                 % [°C]
        experiment.theta_t_CCS=                 8.3e1;              % [min]
        experiment.theta_p_CCS=                 2.5e2;              % [min]
```

```
    experiment.A_CCS=                       0.152;
    experiment.B_CCS=                       0.03;
    experiment.f_exp=                       [];
    experiment.T_exp=                       [];
    experiment.volume_reactor=              Vr_vol_t0(experiment); % [L]
    experiment.ODE_eqns=                    'no_kfm_no_ktc_with_feedtanks';
    experiment.thetas_calc_rule=            'BFC';
    experiment.fs0=                         fs0_calc(experiment);
    experiment.flag_plot=                   [];
    % Experimental data:
    [experiment.Conv_x,experiment.Conv_y]=  experimental_data(...
        'Conversion I0=0.01548, T=70');
    [experiment.Mn_x,experiment.Mn_y]=      experimental_data('Mn I0=0.01548, T=70');
    [experiment.Mw_x,experiment.Mw_y]=      experimental_data('Mw I0=0.01548, T=70');
    [experiment.Conv_x2,experiment.Conv_y2]=experimental_data('No data selected');
    [experiment.Mn_x2,experiment.Mn_y2]=    experimental_data('No data selected');
    [experiment.Mw_x2,experiment.Mw_y2]=    experimental_data('No data selected');

case 11
    experiment.title=                       ...
        'Ray, Saraf and Gupta model. I0=0.01548, T=70°C.';
    experiment.model=                       'Ray_Saraf_Gupta';
    experiment.timespan_min=                [0 120];            % [min]
    experiment.monomer=                     'MMA';
    experiment.monomer_content_mols=        9.366;              % [mol]
    experiment.monomer_feed_mols=           0;
    experiment.monomer_feed_start_time=     [];
    experiment.monomer_F_nM=                0;                  % [mol/min]
    experiment.initiator=                   'AIBN';
    experiment.initiator_content_mols=      0.01548;            % [mol]
    experiment.initiator_feed_mols=         0;
    experiment.initiator_feed_start_time=   [];                 % [min]
    experiment.initiator_F_nI=              0;                  % [mol/min]
    experiment.solvent=                     [];
    experiment.solvent_content_mols=        0;                  % [mol]
    experiment.solvent_feed_mols=           0;
    experiment.solvent_feed_start_time=     [];
    experiment.solvent_F_nS=                0;
    experiment.CTA=                         [];
    experiment.CTA_content_mols=            0;                  % [mol]
    experiment.CTA_feed_mols=               0;
    experiment.CTA_feed_start_time=         [];
    experiment.CTA_F_nCTA=                  0;
    experiment.temperature=                 70;                 % [°C]
    experiment.volume_reactor=              Vr_vol_t0(experiment); % [L]
    experiment.ODE_eqns=                    'no_kfm_no_ktc_with_feedtanks';
    experiment.theta_t_RSG=                 @(T)4.48209e16;     % [s]
    experiment.theta_p_RSG=                 @(T)2.54906e11;     % [s]
    experiment.f_exp=                       0.6493;
    experiment.T_exp=                       [];
    experiment.thetas_calc_rule=            [];
    experiment.fs0=                         fs0_calc(experiment);
    experiment.DOE_trial=                   [];
    experiment.flag_plot=                   [];
    % Experimental data:
    [experiment.Conv_x,experiment.Conv_y]=  experimental_data(...
        'Conversion I0=0.01548, T=70');
    [experiment.Mn_x,experiment.Mn_y]=      experimental_data('Mn I0=0.01548, T=70');
    [experiment.Mw_x,experiment.Mw_y]=      experimental_data('Mw I0=0.01548, T=70');
    [experiment.Conv_x2,experiment.Conv_y2]=experimental_data('No data selected');
    [experiment.Mn_x2,experiment.Mn_y2]=    experimental_data('No data selected');
    [experiment.Mw_x2,experiment.Mw_y2]=    experimental_data('No data selected');

case 20
    experiment.title=                       ...
        'Seth and Gupta model. Step T incr. 50°C to 70°C)';
    experiment.model=                       'Seth_Gupta';
    experiment.timespan_min=                [0 200];            % [min]
    experiment.monomer=                     'MMA';
    experiment.monomer_content_mols=        9.352;              % [mol]
    experiment.monomer_feed_mols=           0;
    experiment.monomer_feed_start_time=     [];
    experiment.monomer_F_nM=                0;                  % [mol/min]
    experiment.initiator=                   'AIBN';
    experiment.initiator_content_mols=      0.0258;             % [mol]
    experiment.initiator_feed_mols=         0;
    experiment.initiator_feed_start_time=   [];                 % [min]
    experiment.initiator_F_nI=              0;                  % [mol/min]
    experiment.solvent=                     [];
    experiment.solvent_content_mols=        0;                  % [mol]
    experiment.solvent_feed_mols=           0;
    experiment.solvent_feed_start_time=     [];
    experiment.solvent_F_nS=                0;
    experiment.CTA=                         [];
    experiment.CTA_content_mols=            0;                  % [mol]
    experiment.CTA_feed_mols=               0;
    experiment.CTA_feed_start_time=         [];
```

```
    experiment.CTA_F_nCTA=                  0;
    experiment.temperature=                 50;                 % [°C]
    experiment.volume_reactor=              Vr_vol_t0(experiment); % [L]
    experiment.ODE_eqns=                    'no_kfm_no_ktc_with_feedtanks';
    experiment.theta_t_RSG=                 [];                 % [s]
    experiment.theta_p_RSG=                 [];                 % [s]
    experiment.f_exp=                       [];
    experiment.T_exp=                       @(t)SI(t);
    experiment.thetas_calc_rule=            'BFC';
    experiment.fs0=                         fs0_calc(experiment);
    experiment.flag_plot=                   [];
    % Experimental data:
    [experiment.Conv_x,experiment.Conv_y]=  experimental_data('Conversion SI');
    [experiment.Mn_x,experiment.Mn_y]=      experimental_data('No data selected');
    [experiment.Mw_x,experiment.Mw_y]=      experimental_data('No data selected');
    [experiment.Conv_x2,experiment.Conv_y2]=experimental_data('No data selected');
    [experiment.Mn_x2,experiment.Mn_y2]=    experimental_data('No data selected');
    [experiment.Mw_x2,experiment.Mw_y2]=    experimental_data('No data selected');

case 21
    experiment.title=                       ...
        'Seth and Gupta model. Run IA50a2 (Intermediate feed of monomer and initiator).';
    experiment.model=                       'Seth_Gupta';
    experiment.timespan_min=                [0 250];            % [min]
    experiment.monomer=                     'MMA';
    experiment.monomer_content_mols=        400*0.940/100.121;  % [mol]
    experiment.monomer_feed_mols=           75*0.940/100.121;   % [mol]
    experiment.monomer_feed_start_time=     170;                % [min]
    experiment.monomer_F_nM=                75*0.940/100.121;   % [mol/min]
    experiment.initiator=                   'AIBN';
    experiment.initiator_content_mols=      0.4*0.01548;        % [mol]
    experiment.initiator_feed_mols=         0.0403;
    experiment.initiator_feed_start_time=   170;                   % [min]
    experiment.initiator_F_nI=              0.0403;             % [mol/min]
    experiment.solvent=                     [];
    experiment.solvent_content_mols=        0;                  % [mol]
    experiment.solvent_feed_mols=           0;
    experiment.solvent_feed_start_time=     [];
    experiment.solvent_F_nS=                0;
    experiment.CTA=                         [];
    experiment.CTA_content_mols=            0;                  % [mol]
    experiment.CTA_feed_mols=               0;
    experiment.CTA_feed_start_time=         [];
    experiment.CTA_F_nCTA=                  0;
    experiment.temperature=                 50;                 % [°C]
    experiment.volume_reactor=              Vr_vol_t0(experiment); % [L]
    experiment.ODE_eqns=                    'no_kfm_no_ktc_with_feedtanks';
    experiment.theta_t_RSG=                 [];                 % [s]
    experiment.theta_p_RSG=                 [];                 % [s]
    experiment.f_exp=                       [];
    experiment.T_exp=                       @(t)IA50a2(t);
    experiment.thetas_calc_rule=            'BFC';
    experiment.fs0=                         fs0_calc(experiment);
    experiment.flag_plot=                   [];
    % Experimental data:
    [experiment.Conv_x,experiment.Conv_y]=  experimental_data('Conversion IA50a2');
    [experiment.Mn_x,experiment.Mn_y]=      experimental_data('No data selected');
    [experiment.Mw_x,experiment.Mw_y]=      experimental_data('No data selected');
    [experiment.Conv_x2,experiment.Conv_y2]=experimental_data('No data selected');
    [experiment.Mn_x2,experiment.Mn_y2]=    experimental_data('No data selected');
    [experiment.Mw_x2,experiment.Mw_y2]=    experimental_data('No data selected');

case 23
    experiment.title=                       ...
        'Curteanu and Bulacovschi model. Example of Figure 5 of Ray, Saraf, Gupta.';
    experiment.model=                       'Curteanu_Bulacovschi';
    experiment.timespan_min=                [0 120];            % [min]
    experiment.monomer=                     'MMA';
    experiment.monomer_content_mols=        9.366;              % [mol]
    experiment.monomer_feed_mols=           0;
    experiment.monomer_feed_mols=           0;
    experiment.monomer_feed_start_time=     [];
    experiment.monomer_F_nM=                0;                  % [mol/min]
    experiment.monomer_feed_start_time=     [];
    experiment.initiator=                   'AIBN';
    experiment.initiator_content_mols=      0.01548;            % [mol]
    experiment.initiator_feed_mols=         0.0132;
    experiment.initiator_feed_start_time=   25;                 % [min]
    experiment.initiator_F_nI=              0.0132;             % [mol/min]
    experiment.solvent=                     [];
    experiment.solvent_content_mols=        0;                  % [mol]
    experiment.solvent_feed_mols=           0;
    experiment.solvent_feed_start_time=     [];
    experiment.solvent_F_nS=                0;
    experiment.CTA=                         [];
    experiment.CTA_content_mols=            0;                  % [mol]
    experiment.CTA_feed_mols=               0;
```

```
    experiment.CTA_feed_start_time=         [];
    experiment.CTA_F_nCTA=                  0;
    experiment.temperature=                 70;                 % [°C]
    experiment.volume_reactor=              Vr_vol_t0(experiment); % [L]
    experiment.ODE_eqns=                    'no_kfm_no_ktc_with_feedtanks';
    experiment.theta_t_RSG=                 @(T)4.48209e16;     % [s]
    experiment.theta_p_RSG=                 @(T)2.54906e11;     % [s]
    experiment.f_exp=                       0.6493;
    experiment.T_exp=                       [];
    experiment.thetas_calc_rule=            [];
    experiment.fs0=                         fs0_calc(experiment);
    experiment.flag_plot=                   [];
    % Experimental data:
    % (I0=0.01548, T=70):
    [experiment.Conv_x,experiment.Conv_y]=  experimental_data(...
        'Conversion I0=0.01548, T=70');
    [experiment.Mn_x,experiment.Mn_y]=      experimental_data('Mn I0=0.01548, T=70');
    [experiment.Mw_x,experiment.Mw_y]=      experimental_data('Mw I0=0.01548, T=70');
    % (I0=0.0258, T=70):
    [experiment.Conv_x2,experiment.Conv_y2]=experimental_data(...
        'Conversion I0=0.0258, T=70');
    [experiment.Mn_x2,experiment.Mn_y2]=    experimental_data('Mn I0=0.0258, T=70');
    [experiment.Mw_x2,experiment.Mw_y2]=    experimental_data('Mw I0=0.0258, T=70');

case 24
    experiment.title=                       ...
        'Sangwai et al model. Example of Figure 5 of Ray, Saraf, Gupta.';
    experiment.model=                       'Sangwai_et_al';
    experiment.timespan_min=                [0 120];            % [min]
    experiment.monomer=                     'MMA';
    experiment.monomer_content_mols=        9.366;              % [mol]
    experiment.monomer_feed_mols=           0;
    experiment.monomer_feed_mols=           0;
    experiment.monomer_feed_start_time=     [];
    experiment.monomer_F_nM=                0;                  % [mol/min]
    experiment.monomer_feed_start_time=     [];
    experiment.initiator=                   'AIBN';
    experiment.initiator_content_mols=      0.01548;            % [mol]
    experiment.initiator_feed_mols=         0.0132;
    experiment.initiator_feed_start_time=   25;                 % [min]
    experiment.initiator_F_nI=              0.0132;             % [mol/min]
    experiment.solvent=                     [];
    experiment.solvent_content_mols=        0;                  % [mol]
    experiment.solvent_feed_mols=           0;
    experiment.solvent_feed_start_time=     [];
    experiment.solvent_F_nS=                0;
    experiment.CTA=                         [];
    experiment.CTA_content_mols=            0;                  % [mol]
    experiment.CTA_feed_mols=               0;
    experiment.CTA_feed_start_time=         [];
    experiment.CTA_F_nCTA=                  0;
    experiment.temperature=                 70;                 % [°C]
    experiment.volume_reactor=              Vr_vol_t0(experiment); % [L]
    experiment.ODE_eqns=                    'kfm_ktc_feedtanks';
    experiment.theta_t_RSG=                 @(T)4.48209e16;     % [s]
    experiment.theta_p_RSG=                 @(T)2.54906e11;     % [s]
    experiment.f_exp=                       0.6493;
    experiment.T_exp=                       [];
    experiment.thetas_calc_rule=            [];
    experiment.fs0=                         fs0_calc(experiment);
    experiment.flag_plot=                   [];
    % Experimental data:
    % (I0=0.01548, T=70):
    [experiment.Conv_x,experiment.Conv_y]=  experimental_data(...
        'Conversion I0=0.01548, T=70');
    [experiment.Mn_x,experiment.Mn_y]=      experimental_data('Mn I0=0.01548, T=70');
    [experiment.Mw_x,experiment.Mw_y]=      experimental_data('Mw I0=0.01548, T=70');
    % (I0=0.0258, T=70):
    [experiment.Conv_x2,experiment.Conv_y2]=experimental_data(...
        'Conversion I0=0.0258, T=70');
    [experiment.Mn_x2,experiment.Mn_y2]=    experimental_data('Mn I0=0.0258, T=70');
    [experiment.Mw_x2,experiment.Mw_y2]=    experimental_data('Mw I0=0.0258, T=70');

case 31
    experiment.title=                       'Ray 1995. BPO-Bz-MMA @70°C, fs0=0.4';
    experiment.model=                       'Ray_Saraf_Gupta';
    experiment.timespan_min=                [0 300];            % [min]
    experiment.monomer=                     'MMA';
    experiment.monomer_content_mols=        (1000-41.3/1000*242.23/1.334)*.6*...
        0.940/100.121;
    experiment.monomer_feed_mols=           0;
    experiment.monomer_feed_mols=           0;
    experiment.monomer_feed_start_time=     [];
    experiment.monomer_F_nM=                0;                  % [mol/min]
    experiment.monomer_feed_start_time=     [];
    experiment.initiator=                   'BPO';
    experiment.initiator_content_mols=      41.3/1000;          % [mol]
```

```
    experiment.initiator_feed_mols=         0;
    experiment.initiator_feed_start_time=   0;                  % [min]
    experiment.initiator_F_nI=              0;                  % [mol/min]
    experiment.solvent=                     'Benzene';
    experiment.solvent_content_mols=        (1000-41.3/1000*242.23/1.334)*...
        0.4*0.876/78.11;
    experiment.solvent_feed_mols=           0;
    experiment.solvent_feed_start_time=     [];
    experiment.solvent_F_nS=                0;
    experiment.CTA=                         [];
    experiment.CTA_content_mols=            0;                  % [mol]
    experiment.CTA_feed_mols=               0;
    experiment.CTA_feed_start_time=         [];
    experiment.CTA_F_nCTA=                  0;
    experiment.temperature=                 70;                 % [°C]
    experiment.volume_reactor=              Vr_vol_t0(experiment); % [L]
    experiment.ODE_eqns=                    'kfm_ktc_feedtanks';
    experiment.theta_t_RSG=                 [];                 % [s]
    experiment.theta_p_RSG=                 [];                 % [s]
    experiment.f_exp=                       [];
    experiment.T_exp=                       [];
    experiment.thetas_calc_rule=            'BFC';
    experiment.fs0=                         fs0_calc(experiment);
    experiment.flag_plot=                   [];
    % Experimental data:
    [experiment.Conv_x,experiment.Conv_y]=  experimental_data(...
        'Seth, 1995, figure 6, fs0=0.4');
    [experiment.Mn_x,experiment.Mn_y]=      experimental_data('No data selected');
    [experiment.Mw_x,experiment.Mw_y]=      experimental_data('No data selected');
    [experiment.Conv_x2,experiment.Conv_y2]=experimental_data('No data selected');
    [experiment.Mn_x2,experiment.Mn_y2]=    experimental_data('No data selected');
    [experiment.Mw_x2,experiment.Mw_y2]=    experimental_data('No data selected');

case 32
    experiment.title=                       ...
        'Destruel et al. GDMA as CTA. T=60°C, I0=0.0258';
    experiment.model=                       'Seth_Gupta';
    experiment.timespan_min=                [0 1000];%[0 250];          % [min]
    experiment.monomer=                     'MMA';
    experiment.monomer_content_mols=        9.3621;%3;  % [mol]
    experiment.monomer_feed_mols=           0;
    experiment.monomer_feed_start_time=     [];
    experiment.monomer_F_nM=                0;                  % [mol/min]
    experiment.monomer_feed_start_time=     [];
    experiment.initiator=                   'AIBN';
    experiment.initiator_content_mols=      0.00188;            % [mol]
    experiment.initiator_feed_mols=         0;
    experiment.initiator_feed_start_time=   0;                  % [min]
    experiment.initiator_F_nI=              0;                  % [mol/min]
    experiment.solvent=                     [];
    experiment.solvent_content_mols=        0;                  % [mol]
    experiment.solvent_feed_mols=           0;
    experiment.solvent_feed_start_time=     [];
    experiment.solvent_F_nS=                0;
    experiment.CTA=                         'GDMA';
    experiment.CTA_content_mols=            0.0159;                 % [mol]
    experiment.CTA_feed_mols=               0;
    experiment.CTA_feed_start_time=         [];
    experiment.CTA_F_nCTA=                  0;
    experiment.temperature=                 60;                 % [°C]
    experiment.volume_reactor=              Vr_vol_t0(experiment); % [L]
    experiment.ODE_eqns=                    'no_kfm_no_ktc_with_feedtanks';
    experiment.theta_t_RSG=                 [];                 % [s]
    experiment.theta_p_RSG=                 [];                 % [s]
    experiment.f_exp=                       [];
    experiment.T_exp=                       [];
    experiment.thetas_calc_rule=            'BFC';
    experiment.fs0=                         fs0_calc(experiment);
    experiment.flag_plot=                   [];
    % Experimental data:
    [experiment.Conv_x,experiment.Conv_y]=  experimental_data(...
        'Destruel, 1991, figure 1, example 3');
    [experiment.Mn_x,experiment.Mn_y]=      experimental_data('No data selected');
    [experiment.Mw_x,experiment.Mw_y]=      experimental_data('No data selected');
    [experiment.Conv_x2,experiment.Conv_y2]=experimental_data('No data selected');
    [experiment.Mn_x2,experiment.Mn_y2]=    experimental_data('No data selected');
    [experiment.Mw_x2,experiment.Mw_y2]=    experimental_data('No data selected');

case 33
    experiment.title=                       ...
        'Gao and Penlidis 1996, figure 40 - Non-Isothermal';
    experiment.model=                       'AI01';
    experiment.timespan_min=                [0 60];           % [min]
    experiment.monomer=                     'MMA';
    experiment.monomer_content_mols=        (1000-0.057*164.21/1.1)/100.121*0.94;
    experiment.monomer_feed_mols=           0;
    experiment.monomer_feed_start_time=     [];
```

```
        experiment.monomer_F_nM=                0;                  % [mol/min]
        experiment.monomer_feed_start_time=     [];
        experiment.initiator=                   'AIBN';
        experiment.initiator_content_mols=      0.057;              % [mol]
        experiment.initiator_feed_mols=         0;
        experiment.initiator_feed_start_time=   0;                  % [min]
        experiment.initiator_F_nI=              0;                  % [mol/min]
        experiment.solvent=                     [];
        experiment.solvent_content_mols=        0;                  % [mol]
        experiment.solvent_feed_mols=           0;
        experiment.solvent_feed_start_time=     [];
        experiment.solvent_F_nS=                0;
        experiment.CTA=                         [];
        experiment.CTA_content_mols=            0;                  % [mol]
        experiment.CTA_feed_mols=               0;
        experiment.CTA_feed_start_time=         [];
        experiment.CTA_F_nCTA=                  0;
        experiment.temperature=                 70;                 % [°C]
        experiment.volume_reactor=              Vr_vol_t0(experiment); % [L]
        experiment.ODE_eqns=                    'kfm_ktc_feedtanks_depropagation';
        experiment.theta_t_RSG=                 [];                 % [s]
        experiment.theta_p_RSG=                 [];                 % [s]
        experiment.f_exp=                       [];
        experiment.T_exp=                       @(t)NonIsothermal_1(t);
        experiment.thetas_calc_rule=            'BFC';
        experiment.fs0=                         fs0_calc(experiment);
        experiment.flag_plot=                   [];
        % Experimental data:
        [experiment.Conv_x,experiment.Conv_y]=  experimental_data(...
            'Gao and Penlidis 1996, figure 40 - Non-Isothermal');
        [experiment.Mn_x,experiment.Mn_y]=      experimental_data('No data selected');
        [experiment.Mw_x,experiment.Mw_y]=      experimental_data('No data selected');
        [experiment.Conv_x2,experiment.Conv_y2]=experimental_data('No data selected');
        [experiment.Mn_x2,experiment.Mn_y2]=    experimental_data('No data selected');
        [experiment.Mw_x2,experiment.Mw_y2]=    experimental_data('No data selected');

    case 41
        experiment.title=                       'SD70(20)50 0.01548 - Modelo Sangwai';
        experiment.model=                       'AI02';
        experiment.timespan_min=                [0 400];            % [min]
        experiment.monomer=                     'MMA';
        experiment.monomer_content_mols=        (1000-0.01548/1.1*164.21)*0.94/100.121;
        experiment.monomer_feed_mols=           0;
        experiment.monomer_feed_mols=           0;
        experiment.monomer_feed_start_time=     [];
        experiment.monomer_F_nM=                0;                  % [mol/min]
        experiment.monomer_feed_start_time=     [];
        experiment.initiator=                   'AIBN';
        experiment.initiator_content_mols=      0.01548;            % [mol]
        experiment.initiator_feed_mols=         0;
        experiment.initiator_feed_start_time=   [];                 % [min]
        experiment.initiator_F_nI=              0;              % [mol/min]
        experiment.solvent=                     [];
        experiment.solvent_content_mols=        0;                  % [mol]
        experiment.solvent_feed_mols=           0;
        experiment.solvent_feed_start_time=     [];
        experiment.solvent_F_nS=                0;
        experiment.CTA=                         [];
        experiment.CTA_content_mols=            0;                  % [mol]
        experiment.CTA_feed_mols=               0;
        experiment.CTA_feed_start_time=         [];
        experiment.CTA_F_nCTA=                  0;
        experiment.temperature=                 70;                 % [°C]
        experiment.volume_reactor=              Vr_vol_t0(experiment); % [L]
        experiment.ODE_eqns=                    'kfm_ktc_feedtanks_depropagation';
        experiment.theta_t_RSG=                 [];     % [s]
        experiment.theta_p_RSG=                 [];     % [s]
        experiment.f_exp=                       [];
        experiment.T_exp=                       @(t)SangwaiSD70_20_50_01548_0258(t);
        experiment.thetas_calc_rule=            [];
        experiment.fs0=                         fs0_calc(experiment);
        experiment.flag_plot=                   [];
        % Experimental data:
        [experiment.Conv_x,experiment.Conv_y]=  experimental_data(...
            'Sangwai 2005 - SI50(60)70_I0=0.01548');
        [experiment.Mn_x,experiment.Mn_y]=      experimental_data('No data selected');
        [experiment.Mw_x,experiment.Mw_y]=      experimental_data('No data selected');
        [experiment.Conv_x2,experiment.Conv_y2]=experimental_data('No data selected');
        [experiment.Mn_x2,experiment.Mn_y2]=    experimental_data('No data selected');
        [experiment.Mw_x2,experiment.Mw_y2]=    experimental_data('No data selected');

end

end

function [fs0]=fs0_calc(experiment)
```

```
T=experiment.temperature;

if isempty(experiment.solvent)
    fs0=0;
else
    mon_parameters=monomer_database(experiment.monomer);
    ro_m=mon_parameters.ro_m;
    MM_M=mon_parameters.MW;
    sol_parameters=solvent_database(experiment.solvent);
    ro_s=sol_parameters.ro_s;
    MM_S=sol_parameters.MW;
    fs0=(experiment.solvent_content_mols*MM_S/ro_s(T))/(experiment.solvent_content_mols*...
        MM_S/ro_s(T)+experiment.monomer_content_mols*MM_M/ro_m(T));
end
end

function [Vr0]=Vr_vol_t0(experiment)

    monomer_mols=experiment.monomer_content_mols;
    monomer_name=experiment.monomer;
    initiator_mols=experiment.initiator_content_mols;
    initiator_name=experiment.initiator;
    solvent_mols=experiment.solvent_content_mols;
    solvent_name=experiment.solvent;

    T=experiment.temperature;

    mon_parameters=monomer_database(monomer_name);
    init_parameters=initiator_database(initiator_name);
    if isempty(experiment.solvent)
        solv_parameters=solvent_database('Benzene');
    else
        solv_parameters=solvent_database(solvent_name);
    end

    ro_m=mon_parameters.ro_m;
    MM_M=mon_parameters.MW;
    ro_I=init_parameters.ro_I;
    MM_I=init_parameters.MW;
    ro_s=solv_parameters.ro_s;
    MM_S=solv_parameters.MW;

    Vr0=(monomer_mols*MM_M/ro_m(25)+initiator_mols*MM_I/ro_I+solvent_mols*...
        MM_S/ro_s(25))/1000;

end

% Temperature profile functions:

function [T]=SangwaiSD70_20_50_01548_0258(t)
    if t<=7.85
        T=46.825-0.737*t+1.5014*t^2-0.133*t^3;
    elseif t<=20
        T=70;
    elseif t<=27.55
        T=70.557-2.8423*(t-20)-1.3506*(t-20)^2+0.3662*(t-20)^3-0.0247*(t-20)^4;
    else
        T=50;
    end
end

function [T]=SI(t)
    if t<=7.13
        T=22.188-2.62392*t+1.2611*t^2-0.046637*t^3;
    elseif t<=120
        T=50;
    elseif t<=124.8
        T=6786.5-111.538*t+0.4616*t^2;
    else
            T=70;
    end
end

function [T]=IA50a2(t)
    if t<=9.16
        T=21.4078-0.424793*t+0.329944*t^2;
    else
            T=50;
    end
end

function [T]=NonIsothermal_1(t)
```

```
if t<29.65
        T=-0.0040705253902793*t^2 + 0.219022984053829*t + 343.269664553352;
    elseif t<36.73
        T=0.0660248252664815*t^2 - 3.91290996938007*t + 404.298952765181;
    elseif t<40.87
        T=-0.0391108146868646*t^4 + 6.97045232740974*t^3 - 453.833777571986*t^2 ...
            + 12881.0817303124*t - 134725.863780163;
    elseif t<40.92
        T=503.33333345658*t - 20180.0000050982;
    elseif t<45.54
        T=3.6660329194392*t^2 - 331.154316759586*t + 7824.97283787778;
    elseif t<60
        T=0.0171846347939919*t^2 - 2.0476190666331*t + 404.459017054792;
    else
        T=70;

end
T=T-273.15;

end
```

experimental_data.m:

```
function [x,y] = experimental_data(option)
%EXPERIMENTAL_DATA Selects the experimental data to be used as comparison
% Data extracted from original articles with
% https://automeris.io/WebPlotDigitizer/index.html

switch option

    case 'No data selected'
        x=[];
        y=[];

    case 'Conversion I0=0.01548, T=70' %%%%%
        % Experiment W. Y. Chiu, G. M. Carratt, D. S. Soong, 1982, I0=0.01548, T=70°C
        x=[10.32 15.28 20.24 32.15 50.01 60.52 65.27 69.97 85.05 90.22 100.15]; % Time
        y=[0.055 0.087 0.118 0.193 0.280 0.380 0.485 0.826 0.899 0.907 0.915]; % Conversion

    case 'Mn I0=0.01548, T=70'
        % Experiment W. Y. Chiu, G. M. Carratt, D. S. Soong, 1982, I0=0.01548, T=70°C
        x=[0.0591 0.0859 0.1225 0.3811 0.8279];
        y=[174532 174532 173088 188091 297114];

    case 'Mw I0=0.01548, T=70'
        % Experiment W. Y. Chiu, G. M. Carratt, D. S. Soong, 1982, I0=0.01548, T=70°C
        x=[0.0573 0.0574 0.0574 0.0575 0.0793 0.0793 0.0794 0.1207 0.3800 0.3776 0.3802...
            0.4902 0.8298];
        y=[312309 328280 342212 365743 317544 336570 350853 304617 627818 670988 735234 ...
            991721 1633038];

    case 'Conversion I0=0.0258, T=70' %%%%
        x=[4.90  9.60  20.00 34.90 45.10 55.29 55.29 60.20 65.10 70.00 74.90 79.80 79.80];
        y=[0.043 0.081 0.151 0.251 0.333 0.515 0.526 0.760 0.868 0.889 0.899 0.911 0.899];

    case 'Mn I0=0.0258, T=70'
        x=[0.0408 0.0408 0.084 0.081 0.142 0.140 0.333 0.528 0.531 0.764 0.762 0.891 ...
            0.891 0.891 0.888];
        y=[134369 118979 129779 115918 132054 120017 135542 161267 182128 222417 251188 ...
            212961 234321 251188 271620];

    case 'Mw I0=0.0258, T=70'
        x=[0.041 0.044 0.082 0.082 0.082 0.140 0.140 0.143 0.336 0.333 0.534 0.532 ...
            0.763 0.890 0.890 0.890];
        y=[242608 269270 269270 240509 218586 232293 257822 281226 346435 381181 833210 ...
            924778 1129354 1297799 1379185 1452995];

    case 'Conversion SI' %%%%
        x=[15.182 29.868 43.920 59.089 74.305 89.411 104.62 114.375 124.402 129.414 ...
            130.97 134.103]; % Time
        y=[0.020 0.023 0.059 0.085 0.093 0.141 0.152 0.167 0.282 0.440 0.469 0.517];

    case 'Conversion IA50a2'%%%%
        x=[30.06 59.61 89.67 119.19 149.78 159.24 171.47 173.51 188.78 199.28 203.95 ...
            210 214.86 219.88 221.56 226.6]; % Time
        y=[0.027 0.049 0.0757 0.105 0.129 0.147 0.132 0.151 0.171 0.196 0.220 0.300 ...
            0.415 0.489 0.606 0.819]; % Conversion

    case 'Destruel, 1991, figure 1, example 3' %%%%
        x=[240.1 398.40 480.3 499.9 538.1 568.97 628.4 677.6];
        y=[0.148 0.282 0.339 0.400 0.414 0.469 0.490 0.578];
```

```
    case 'Seth, 1995, figure 6, fs0=0.4' %%%%
        x=[9.6 19.7 29.7 40.1 59.8 70.2 80.2 89.9 99.7 109.4 119.8 129.8 139.9 159.6 ...
            179.4 199.8 219.9 239.7];
        y=[0.086 0.138 0.186 0.243 0.327 0.389 0.417 0.467 0.488 0.517 0.559 0.603 ...
            0.628 0.712 0.802 0.892 0.953 0.991];

    case 'Gao and Penlidis 1996, figure 40 - Non-Isothermal' %%%%
        x=[4.929 9.624 14.872 20.202 25.061 30 35.260 39.866 45.054 50.053 55.374];
        y=[0.035 0.089 0.13 0.183 0.255 0.331 0.427 0.772 0.893 0.890 0.903];

    case 'Sangwai 2005 - SI50(60)70_I0=0.01548' %%%%
        x=[75.8 106 125.4 135 145.9 150.4 157.7 165.3 168.5 171.7];
        y=[0.131 0.196 0.209 0.237 0.247 0.278 0.295 0.326 0.367 0.395];

end

end
```