research center pharmaceutical engineering

Diploma Thesis

(Diplomarbeit zur Erlangung des akademischen Grades eines Diplom-Ingenieurs der Studienrichtung Verfahrenstechnik an der Technischen Universität Graz)

Numerical Simulation of Nanoparticle Precipitation via the Population Balance Equation

Numerische Simulation von Nanopartikel Präzipitation mittels Populationsbilanzgleichung

Andreas Eitzlmayr

Advised by:

Dr.-Ing. Daniele Suzzi

Dipl.-Ing. Stefan Radl

Univ.-Prof. Dipl.-Ing. Dr.techn. Johannes G. Khinast

TU Graz

Graz University of Technology

Institute for Process and Particle Engineering

Graz, October 2010

Deutsche Fassung:
Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008
Genehmigung des Senates am 1.12.2008

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Graz, am ……. ……..
(Unterschrift)

Englische Fassung:

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

……. ……..
date (signature)

Acknowledgement

I want to thank everybody who supported me during my work. My special thanks is directed to my advisor Univ.-Prof. Dipl.-Ing. Dr.techn. Johannes G. Khinast for the interesting topic and lots of fruitful discussions, Dipl.-Ing. Stefan Radl and Dr.-Ing. Daniele Suzzi, who contributed with lots of knowledge, recommendations and feedback.

Furthermore I want to thank Univ.-Prof. Dr. Andreas Zimmer and Mag. Christina Petschacher (Institute of Pharmaceutical Sciences, University of Graz) for the kind cooperation and for the provided experimental data.

Also gratefully mentioned are Dipl.-Ing. Gregor Toschkoff for helpful discussions and feedback about the MATLAB Code, and Dipl.-Ing. Rafael Eder for helpful literature recommendations.

For the financial funding I want to acknowledge the Research Center Pharmaceutical Engineering and the research project “Nano-Health”, sponsored by the Austrian Federal Ministry of Transport, Innovation and Technology.

Last but not least, lots of thanks go to my parents for the possibilities they created for me, and to my family and friends for continuous personal reflection and encouragement.

Kurzfassung

Die Aufgabe dieser Arbeit war, ein numerisches Model für die Vorhersage eines Präzipitationsprozesses, speziell für die Herstellung von neuartigen Polyacrylsäure/Protamin Nanopartikeln, zu entwickeln. Damit sollte das Scale-up-Verhalten eines Mikroreaktors untersucht werden. Wegen der Komplexität des Prozesses war eine räumliche Auflösung (d.h. eine Simulation via Computational Fluid Dynamics) in diesem ersten Modellierungsschritt nicht das Ziel. Vielmehr wurde der für das Scale-up relevante Mischeinfluss mit einem geeigneten Vermischungsmodell berücksichtigt.

Als Modell für die Beschreibung des Präzipitationsprozesses wurde die Populationsbilanzgleichung (PBE) mit Nukleation, Wachstum und Aggregation gewählt. Nicht alle erforderlichen Parameter waren bekannt. Daher wurden in einem ersten Schritt Parameterstudien durchgeführt, um sinnvolle Bereiche für die Werte der unbekannten Parameter zu finden. Dann wurde die PBE für den mathematisch einfachsten Fall, ein ideal durchmischtes System, gelöst. Im nächsten Schritt wurde das Engulfment Modell für die Mikrovermischung mit der PBE gekoppelt, um eine Scale-up Abhängigkeit abzubilden. Zusätzlich wurden Scale-up relevante Ergebnisse durch die Analyse von charakteristischen Zeitmaßstäben erhalten.

In den Parameterstudien wurde die hohe Sensitivität der Nukleationsrate auf Variationen der Grenzflächenenergiekonstante K und der Übersättigung S gezeigt. Durch geeignetes Einstellen von K wurde die mittlere Teilchengröße eines experimentellen Ergebnisses reproduziert. Die Breite der errechneten Größenverteilung hingegen war deutlich schmäler. Im Gegensatz zu den Erwartungen war es nicht möglich, mit dem Engulfment Modell die Breite der Verteilung zu erhöhen. Durch die Analyse von charakteristischen Zeitmaßstäben wurden Kriterien für das Scale-up gefunden (d.h., Einströmgeschwindigkeit $u \sim d^{1/3}$, wobei d der Längenmaßstab des Reaktors ist, bzw. Scale-up basierend auf konstantem mittlerem spezifischem Energieeintrag ε = const.). Diese Kriterien sind im Fall eines konstanten Widerstandsbeiwertes des Reaktors identisch. Unsere Ergebnisse zeigen, dass ε=const. ein gut geeignetes Scale-up Kriterium zur Einhaltung der Mischzeit in einem Mikroreaktor ist. Sowohl die Resultate des gekoppelten Modells, als auch die Analyse der charakteristischen Zeitmaßstäbe zeigen, dass kleine Längenmaßstäbe in der Größenordnung von Millimetern, d.h. Mikroreaktoren, gut geeignet sind, um die erforderliche schnelle Vermischung zu erhalten.

Abstract

The goal of this work was to develop a numerical model for the prediction of polyacrylic-acid/protamine nano particle precipitation. The model should allow an investigation of the scale-up performance of a micro reactor. Due to the complexity of the process, a spatial resolution (i.e., a simulation via computational fluid dynamics) was not targeted in this first step of modelling. The scale-up relevant mixing influence was taken into account by an adequate mixing model.

As model for the description of the precipitation process, the Population Balance Equation (PBE), including nucleation, growth and aggregation was chosen. Not all of the required model parameters were known. Thus, the first step was to perform parameter studies in order to get reasonable ranges for the values of the unknown parameters. Then, the PBE was solved for the mathematically easiest case, a well-mixed system. In the next stage, the engulfment model for micro mixing was coupled with the PBE in order to investigate mixing effects during scale-up. Additionally, scale-up relevant results were obtained by the analysis of characteristic time scales.

In the parameter studies, the high sensitivity of the nucleation rate to variations of the interfacial energy constant K and the supersaturation S was investigated. The parameter K was adjusted to reproduce the mean particle size of an experimental result. The width of the calculated distribution, however, was predicted significantly smaller than in the experiment. It was not possible to increase the width of the distribution by using a mixing model (i.e., the engulfment model). Via the analysis of characteristic time scales, scale-up criteria have been identified (e.g., the inlet velocity should scale with $u \sim d^{1/3}$, here d is the length scale, or scaling based on a constant mean specific power input ε = const.). These two criteria are equal for a constant friction factor of the reactor. Our simulation results show, that ε = const. is a suitable scale-up relation to ensure identical mixing conditions in a micro reactor. Both, the results of the coupled model, as well as the analysis of time scales show, that relatively small length scales in the order of millimeters, i.e. microreactors, are well suited to obtain the required fast mixing.

Table of Contents

List of Figures

List of Tables

Abbreviations

CFD	Computational fluid dynamics
CIJR	Confined impinging jet reactor
CM	Classes method
PAA	Polyacrylic-acid
PBE	Population balance equation
PDF	Probability density function
PDI	Potential determining ions
PSD	Particle size distribution
pzc	Point of zero charge
QMOM	Quadrature method of moments
SMM	Standard method of moments

Nomenclature

Latin symbols:

A	Hamaker constant [J]
a	Surface to surface distance [m]
B'	Preexponential factor [1/m³s]
B_{agg}	Aggregation birth rate [1/m^{4}s]
B_{hom}	Homogeneous nucleation rate [1/m³s]
$B(a)$	Hydrodynamic correction [-]
c	Molar concentration in the liquid phase [mol/l]
c^*	Molar equilibrium concentration in the liquid phase (solubility) [mol/l]
c_M	Mass concentration in the liquid phase [g/l]
c_{MP}	Mass concentration of particles [g/l]
c_{PDI}	Molar concentration of potential determining ions (PDI) [mol/l]
$c_{PDI}^{\ pzc}$	Molar concentration of PDI at point of zero charge (pzc) [mol/l]
c_S	Molar concentration in the solid phase [mol/l]
D	Diffusion coefficient in the liquid phase [m²/s]
Da	Damköhler number [-]
D_{agg}	Aggregation death rate [1/m^{4}s]
d	Reactor inlet diameter [m]
E	Engulfment rate constant [1/s]
e	Unit charge [As]
F	Function F, introduced for the calculation of the charge numbers [-]
F_V	Volumetric flow rate [m³/s]
G	Growth rate [m/s]
ΔG	Free energy change for the formation of new solid phase [J]
ΔG_{cr}	Critical free energy for nucleation [J]
ΔG_S	Free energy change for surface generation [J]
ΔG_V	Free energy change for phase transformation [J]
Δg_{SL}	Specific free energy change for phase transformation [J/m³]

I	Ionic strength [mol/l]
J	Nucleation source term [1/m^{4}s]
K	Interfacial energy constant
K_0	Equilibrium constant for a chemical reaction [Unit depends on the reaction]
K_S	Solubility product [mol²/l²]
k	Boltzmann's constant [J/K]
k_N	Nucleation rate constant [1/m^{3}s]
L	Particle size (volume equivalent diameter) [nm]
L_i	Lower bound of class i [nm]
$L_{m,i}$	Mean size of class i [nm]
L_{mean}	Mean particle size of a distribution [nm]
M	Number of classes [-]
M_i	Molar mass of spezies i [g/mol]
m	Kinetic order of nucleation [-]
m_j	Moment of order j [m^j/m³]
N	Particle number density [1/m³]
N_A	Avogadro's number [1/mol]
n	Particle number density distribution [1/m^4]
n_i	Discrete particle number density distribution of class i [1/m^4]
n_{AA}	Number of acrylic acid-groups per PAA molecule [-]
n_{Cys}	Number of cysteine-groups per PAA molecule [-]
n_{Gua}	Number of guanidinium-groups per protamine molecule [-]
pH	pH-value [-]
pK_a	pK_a-value [-]
Re	Reynolds number [-]
r	Particle radius [m]
$r*$	Mean particle radius (Parameter for hydrodynamic correction) [m]
r_{cr}	Critical cluster radius [m]
r_h	Hydrodynamic radius [m]
r_i	Reaction rate [mol/m³s]

S	Supersaturation [-]
Sc	Schmidt number [-]
Sh	Sherwood number [-]
s	Standard deviation of the particle size distribution [nm]
$s_{relative}$	Relative standard deviation of the particle size distribution [nm]
T	Temperature [K]
t	Time [s]
t_{me}	Mesomixing time scale [s]
t_{mi}	Micromixing time scale [s]
$\mathbf{u}$	Velocity field [m/s]
u	Inlet velocity [m/s]
V	Micromixed volume [m³]
V_0	Initial value for the micromixed volume [m³]
V_m	Molecular volume [m³]
V_R	Reactor volume [m³]
W	Stability ratio [-]
w	Weight factor [-]
X	Conversion [-]
X_{me}	Mesomixed volume fraction [-]
X_{me}^{0}	Initial value for the mesomixed volume fraction [-]
X_{mi}	Micromixed volume fraction [-]
X_{mi}^{0}	Initial value for the micromixed volume fraction [-]
$\mathbf{x}$	Spatial coordinates [m]
x_i	Molar fraction of component i [-]
z	Charge number [-]

Greek symbols:

α	Aggregation efficiency [-]
β_{agg}	Aggregation kernel [m³/s]
β_{coll}	Collision kernel [m³/s]
β_{coll}^{*}	Dimensionless collision kernel [-]
Γ	Diffusion coefficient of the particles [m²/s]
ε	Mean specific power input [W/kg]
ε_0	Electrical field constant [As/Vm]
ε_r	Relative permittivity [-]
ε_S	Solid phase volume fraction [-]
ζ	Drag coefficient [-]
η	Dynamic viscosity [Pa·s]
κ	Reciprocal Debye length [1/m]
λ	Particle size ratio [-]
ν	Kinematic viscosity [m²/s]
ν_D	Dissociation number [-]
$\nu_{i,Nucleus}$	Amount of species i in one nucleus [mol]
ξ	Dimensionless surface to surface distance [-]
ρ_S	Solid density [kg/m³]
σ	Interfacial energy [N/m]
τ	Residence time [s]
τ_{coll}	Characteristic time scale for collisions [s]
τ_{growth}	Characteristic time scale for growth [s]
τ_m	Characteristic time scale for mixing [s]
τ_{nuc}	Characteristic time scale for nucleation [s]
ϕ_{el}	Electrostatic interaction potential [J]
ϕ_{total}	Total interaction potential [J]
ϕ_{VdW}	Van der Waals interaction potential [J]
ψ_P	Surface potential [V]

Subscripts:

1	Polyacrylic-acid
2	Protamine
AACOOH	Acrylic-acid
AACOO-	Dissociated acrylic-acid
CysCOOH	Cysteine-acid
CysCOO-	Dissociated cysteine-acid
CysSH	Cysteine-sulfide
CysS-	Dissociated cysteine-sulfide
GuaNH	Guanidinium
GuaNH2+	Protonated guanidinium
H3O+	Hydronium ions
OH-	Hydroxide ions

1 Introduction

The aging population, high expectations on quality of life and the changed lifestyle of people living in Europe demand an improved, more efficient and affordable health care. Serious diseases, like cancer, diabetes, cardiovascular diseases and infections are big challenges of medicine. Nanotechnology can provide an important contribution to face these challenges. The excellence of nanoparticles in diagnostics, imaging and intelligent therapy methods is generally accepted. For example, the project “Nano-Health”, sponsored by the Austrian Federal Ministry of Transport, Innovation and Technology, is focused on four different types of nanoparticles: particles based on (i) lipids, (ii) protamines, (iii) poly-lactic-acid-humanserumalbumin and (iv) thiomeres. The part of Nano-Health, where this work is related to, investigates the production of polyacrylic-acid/protamine nanoparticles by a precipitation process.

Precipitation is a simple, inexpensive and efficient method for the production of nanoparticles. In a precipitation process, the supersaturation of the desired product substance leads to a spontaneous particle formation. Precipitation is often used to transform dissolved components into solid particles, e.g. the precipitation of proteins and other hardly soluble organic components, or inorganic substances in chemical waste water treatment.

However, precipitation is a fast process and can lead to an extremely challenging process dynamics. This is due to highly sensitive kinetics of the initial particle nucleation step, which requires an adequate description of the mixing process down to the smallest relevant length scales. Thus, the numerical simulation of precipitation is still a challenge and requires sophisticated methods and models. Especially the precipitation of nanoparticles made of organic macromolecules, as investigated in this work, has never been simulated before.

The overall goal of this work was to develop a numerical model for the precipitation of polyacrylic-acid/protamine nanoparticles in order to investigate the scale-up behaviour. Due to the complexity of the process, a spatial resolution of the process was not the goal in this first stage of the project. Therefore, the simulation of a well-mixed system as a simple case was studied. Subsequently, the coupling with an appropriate mixing model was investigated, in order to capture general qualitative scale-up trends.

This thesis is structured as follows: the first step was to document the state-of-the-art in numerical simulation of precipitation processes, see Chapter 2. In Chapter 3 the parameters required for the calculations are documented and the material balances of the process have been calculated. To allocate ranges for the values of some unknown parameters and to get a deeper understanding of the models, parameter studies were performed in Chapter 4. In Chapter 5 the solution for a well-mixed system is documented. The solution for the precipitation coupled with a mixing model is shown in Chapter 6. The coupled model was used to predict trends during scale-up, see Chapter 7. Finally, the error due to the discretization of the equations was investigated in Chapter 8.

The results obtained in this work are generally valid for any reactor type or detailed geometrical configuration of the reactor. Only for the estimation of characteristic time scales a certain geometry had to be chosen. A so-called “confined impinging jet reactor“ (CIJR) has been used for this purpose. It can be expected, that the trends obtained for this reactor yield also acceptable results for any other geometry, if similar mixing regimes can be ensured.

2 Background

2.1 Precipitation

Precipitation is the spontaneous generation of solid particles in a liquid solution due to supersaturation. Precipitation is similar to crystallization, and sometimes it is also called “reactive crystallization” (see Aoun et al. [1] and Paschedag [2]). This is because the supersaturation in precipitation processes is typically generated by a chemical reaction. However, the supersaturation can also be generated by mixing with an antisolvent (see, e.g., Beck et al. [3]).

The initial supersaturation in a precipitation process is usually orders of magnitudes higher than in crystallization processes [2]. In the latter the initial supersaturation is typically in the metastable region and therefore not high enough to initiate homogeneous nucleation. Thus, crystallization is frequently initiated by seeding. In contrast, precipitation never requires seeding, but generates particles spontaneously due to the high supersaturation. The induction time, i.e., the time needed for nuclei generation, is typically in the order of milliseconds. For crystallization processes, however, the induction time can be in the order of minutes or more (Aoun et al. [1]).

Precipitation of inorganic salts has already been studied experimentally and numerically, e.g. by Gavi et al. [4], [5], Schwarzer et al. [6], Baldyga et al. [7] or Aoun et al. [1]. The precipitation of polyacrylic-acid and protamine was studied in this work. These substances are organic macromolecules and behave different from inorganic salts. However, we adapted the models used for inorganic salts to study precipitation of these organic molecules.

2.2 Definitions

Before starting to describe the precipitation model, some basic definitions are given, to distinguish between various terms used in literature.

- **Molecules** are the smallest units of the species involved in the process.
- **Ions** are charged molecules (for polyacrylic-acid and protamine the charges are caused by acidic and basic groups and depend on the pH-value).

- **Clusters** are local, temporary accumulations of ions in the solution (i.e., concentration fluctuations) with a density in the order of the solid density. Clusters are thermodynamically unstable and are part of the liquid phase.
- **Nuclei** are clusters, which have reached the critical size and become stable. Nuclei are part of the solid phase.
- **Particles** are units of solid phase larger than or equal to the critical nuclei size.
- **Nucleation** means the generation of nuclei.
- **Homogeneous nucleation** means the generation of nuclei without the contribution of a preexisting solid phase, as defined in Myerson [8].
- **Growth** is the enlargement of particles by attachment of molecules or ions.
- **Aggregation** means the unification of particles after their collision.
- **Agglomeration** is the sum of aggregation plus molecular growth of the bridge at the contact point, as described in Paschedag [2]. The mathematical model used in this work does not distinguish between agglomeration and aggregation.

2.3 Description of the Particle Population

The process steps influencing the particle population are nucleation, growth and aggregation. These processes, as well as the particle positions in a reactor have to be described mathematically, because they all depend on the local composition and flow field. In general two different approaches to describe the particle population can be distinguished:

- the tracking of individual particles (i.e., the Lagrangian particle tracking), and
- the calculation of the number density distribution of the particles.

In both approaches the particle positions have to be calculated as a function of the external variables (time and space) taking into account at least one internal variable to describe the particle population (e.g., the particle size). In an approach based on Lagrangian particle tracking, the population of the particles can be directly calculated. However, in the case of nanoparticle precipitation an extremely high number of particles is expected. Assuming 0.2 g/l spherical particles with 200 nm diameter and 2 g/ml solid density, the particle number is ca. $2 \cdot 10^{10}$ particles per ml. Clearly, this is too much for the tracking of individual particles, and an approach based on the number density distribution of particles is

favourable. Consequently, the unsteady, spatially inhomogeneous population balance equation will be adopted for the description of suspended nanoparticles.

2.3.1 The Population Balance Equation

The population balance equation (PBE) is a transport equation for a particle population in space, time and one or more internal coordinates. The latter describe the characteristics of the particles (e.g., their size, shape, color…). The PBE consists of terms for accumulation, convective and diffusive transport and source terms for nucleation, molecular growth and aggregation, as shown in Gavi et al. [4]. The unsteady, spatially inhomogeneous PBE for a single internal coordinate (i.e., the particle size L) can be written as:

$$\frac{\partial n(L,\mathbf{x},t)}{\partial t} + \nabla\left(\mathbf{u}(\mathbf{x},t)\cdot n(L,\mathbf{x},t)\right) = \nabla\left(\Gamma\cdot\nabla n(L,\mathbf{x},t)\right) + J\left(L,S(\mathbf{x},t)\right) - \frac{\partial\left(G(L,S(\mathbf{x},t))\cdot n(L,\mathbf{x},t)\right)}{\partial L} + B_{agg}\left(L,\mathbf{x},t\right) - D_{agg}\left(L,\mathbf{x},t\right) \quad \textbf{(2-1)}$$

The terms on the left hand side and the first term on the right hand side are known from other transport equations used in computational fluid dynamics (CFD). They account for local accumulation, convection and diffusion, where Γ is the diffusivity of the particles. The remaining terms on the right hand side are source terms accounting for nucleation, growth and aggregation, where J is the nucleation source term and G is the growth rate. The terms $B_{agg}(L,\mathbf{x},t)$ and $D_{agg}(L,\mathbf{x},t)$ accounting for birth and death of particles due to aggregation are defined as (see Marchisio et al. [9]):

$$B_{agg}\left(L,\mathbf{x},t\right) = \frac{L^2}{2}\int_{\tilde{L}=0}^{L} \frac{\beta_{agg}\left(\tilde{L},\sqrt[3]{L^3-\tilde{L}^3}\right)\cdot n\left(\tilde{L},\mathbf{x},t\right)\cdot n\left(\sqrt[3]{L^3-\tilde{L}^3},\mathbf{x},t\right)}{\left(L^3-\tilde{L}^3\right)^{2/3}}\cdot d\tilde{L} \quad \textbf{(2-2)}$$

$$D_{agg}\left(L,\mathbf{x},t\right) = n(L,\mathbf{x},t)\cdot\int_{\tilde{L}=0}^{\infty}\beta_{agg}\left(\tilde{L},L\right)\cdot n\left(\tilde{L},\mathbf{x},t\right)\cdot d\tilde{L} \quad \textbf{(2-3)}$$

The PBE has been used to describe e.g., nanoparticle precipitation for inorganic salts (see e.g. Gavi et al. [4], Baldyga et al. [7] and Schwarzer et al. [10]).

The source terms in the PBE accounting for nucleation, growth and aggregation have to be described by appropriate models. For the precipitation of inorganic salts, i.e., small ions with constant charge and a stoichiometric composition of the solid phase, accurate models already exist. In this work, the precipitation of large organic molecules with pH-dependent charge numbers and therefore non-stoichiometric composition of the solid phase is considered. A suitable mathematical description of such a precipitation process is more difficult, and has not been documented in literature before. Clearly, the challenge is to adopt existing models developed for inorganic salts and apply them to the system discussed in this work.

2.3.2 Nucleation

There are various mechanisms of nucleation, which can be divided into two major categories: Primary and secondary nucleation. Primary nucleation means the generation of new particles in the absence of preexisting particles, whereas secondary nucleation originates from particles already present in the system. A typical mechanism for secondary nucleation is particle generation due to attrition. Primary nucleation can be homogenous or heterogeneous. Homogenous nucleation takes place in a supersaturated solution without impurities. In contrast, heterogeneous nucleation means nucleation at preferential sites, such as phase boundaries or impurities like dust [8].

Depending on the supersaturation and solubility, a dominant nucleation mechanism will exist in the system. For high supersaturations, homogeneous nucleation can be expected to be dominant, as described by Mersmann [11], [12]. Unfortunately, the solubility of a mixture of polyacrylic-acid and protamine cannot be easily quantified. However it is expected to be extremely low, which was justified later in this work (see Chapter 4). Hence, the initial supersaturation is expected to be high enough for homogeneous nucleation.

The classical theory of nucleation is adopted for this work [8]. Thus, the driving force for nucleation is supersaturation and is defined as the ratio of an actual activity a of the supersaturated species to the thermodynamical equilibrium-activity a* (see Eqn. 2-4). For the reduction of complexity, the activity coefficients are often assumed to be equal to unity. Thus, the supersaturation is simply a ratio of concentrations [12]:

$$S = \frac{a}{a*} \cong \frac{c}{c*} \qquad \textbf{(2-4)}$$

In the classical nucleation theory, it is hypothized, that local fluctuations are forming clusters, which get stable after reaching a critical size. An Arrhenius-type of expression gives then the rate of nucleus formation [8], where k is the Boltzmann-constant and T the temperature:

$$B_{\text{hom}} = B' \cdot \exp\left(-\frac{\Delta G_{cr}}{k \cdot T}\right) \qquad \textbf{(2-5)}$$

This nucleation rate is the product of an exponential term involving ΔG_{cr}, and a preexponential factor, i.e., B'. In this form, the accurate prediction of the exponential term is much more important than the preexponential factor, because of the extremely high sensitivity of the nucleation rate to the exponent. This has already been demonstrated by calculations of Mersmann et al. [12].

To estimate the free energy, a thermodynamic consideration has to be done. Two different types of energy are relevant for the nucleation process: a certain amount of energy is consumed by the generation of the new surface, and another amount of energy is provided by the phase transformation from liquid to solid. Balancing these two energies equals the free energy change ΔG for the formation of the solid phase (under the assumption of spherical clusters) [8]:

$$\Delta G = \Delta G_S - \Delta G_V = 4\pi \cdot r^2 \cdot \sigma - \frac{4}{3} r^3 \pi \cdot \Delta g_{SL} \qquad \textbf{(2-6)}$$

The free energy changes versus the cluster size are shown in Figure 2-1. Clusters greater than the critical size observe a decrease of free energy, will grow and lead to nucleation. The critical cluster size is obtained by maximizing the free energy function (Eqn. 2-6):

$$r_{cr} = \frac{2\sigma}{\Delta g_{SL}} \qquad \textbf{(2-7)}$$

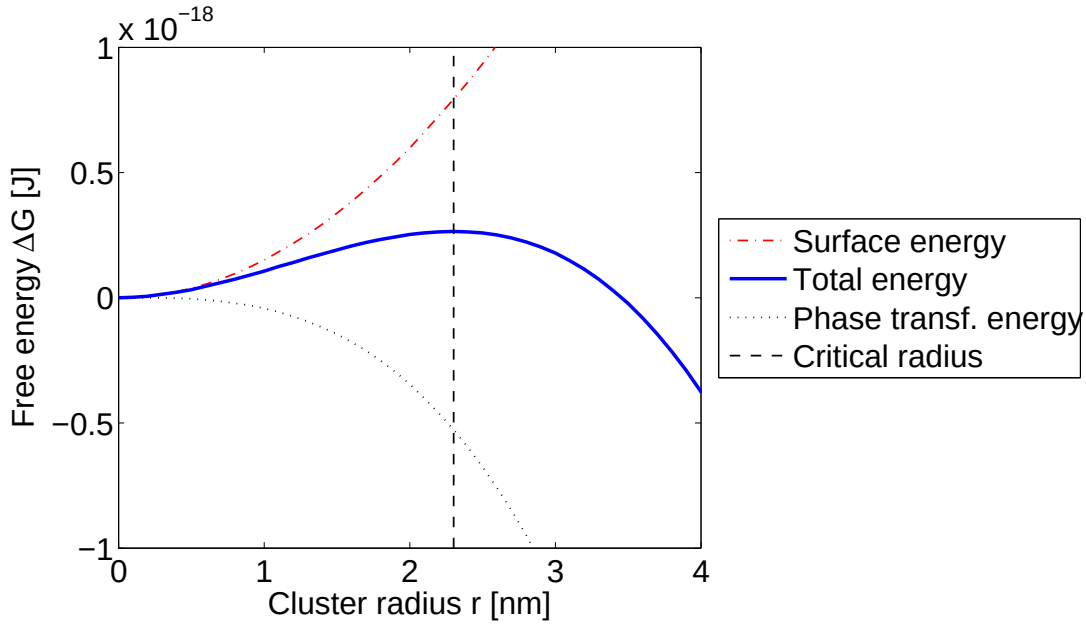


Figure 2-1: Free energy change versus cluster size for a nucleation process.

Substitution of Δg_{SL} in Eqn. 2-6 gives the critical free energy for nucleation:

$$\Delta G_{cr} = \frac{4\pi \cdot r_{cr}^{\ 2} \cdot \sigma}{3} \qquad \textbf{(2-8)}$$

Small particles (and also clusters) have a higher solubility than large ones, because of their higher specific surface. This effect is described by the Gibbs-Thomson equation, where a solution with concentration c is in equilibrium with particles of the radius r [8]:

$$\ln\left(\frac{c}{c*}\right) = \ln S = \frac{2 \cdot \sigma \cdot V_m}{\nu_D \cdot k \cdot T \cdot r} \qquad \textbf{(2-9)}$$

The parameter S is the supersaturation, c* the equilibrium concentration, ν_D the dissociation number and V_m the molecular volume. Substituting the critical particle size in Eqn. 2-8 yields in a relation for the critical free energy for nucleation:

$$\Delta G_{cr} = \frac{16\pi \cdot \sigma^3 \cdot V_m^{\ 2}}{3\left(k \cdot T \cdot \nu_D \cdot \ln S\right)^2} \qquad \textbf{(2-10)}$$

Substitution of ΔG_{cr} in Eqn. 2-5 gives [8]:

$$B_{\text{hom}}(S) = B' \cdot \exp\left(-\frac{16\pi \cdot \sigma^3 \cdot V_m{}^2}{3 \cdot (k \cdot T)^3 \cdot (\nu_D \cdot \ln S)^2} \right) \quad \textbf{(2-11)}$$

For crystallizations, the preexponential factor B' has a theoretical value of 10^{30} nuclei/cm³s [8]. However experiments suggest a value of 10^3 to 10^5 nuclei/cm³s for B'. According to Mersmann [11], the preexponential factor can be calculated from:

$$B' = 1{,}5 \cdot D \cdot (c \cdot N_A)^{7/3} \cdot \left(\frac{\sigma}{k \cdot T} \right)^{0{,}5} \cdot V_m \quad \textbf{(2-12)}$$

Here D is the diffusion coefficient of the supersaturated species with the concentration c and N_A is the Avogadro-number. For the estimation of the interfacial energy σ, Mersmann [13] has introduced an equation based on some fundamental thermodynamic relationships. A comparison with experimental data showed fairly good agreement for various inorganic systems. Because of its theoretical foundation it can also be used for organic systems.

$$\sigma = K \cdot k \cdot T \cdot (c_S \cdot N_A)^{2/3} \cdot \ln\left(\frac{c_S}{c^*} \right) \quad \textbf{(2-13)}$$

Here c_S is the molar concentration in the solid phase. The exact value of the interfacial energy constant K is difficult to determine and should be between 0.310 and 0.414 [12]. However, it is important for the determination of the nucleation rate, which is very sensitive to the value of the interfacial energy. To sum up, the nucleation rate for homogenous nucleation can be written as:

$$B_{\text{hom}}(S) = 1{,}5 \cdot D \cdot (c \cdot N_A)^{7/3} \cdot \left(\frac{\sigma}{k \cdot T} \right)^{0{,}5} \cdot V_m \cdot \exp\left(-\frac{16\pi \cdot \sigma^3 \cdot V_m{}^2}{3 \cdot (k \cdot T)^3 \cdot (\nu_D \cdot \ln S)^2} \right) \quad \textbf{(2-14)}$$

According to Myerson [8] this relation can be simplified in limited ranges of the supersaturation to:

$$B_{\text{hom}}(S) = k_N S^m \qquad \textbf{(2-15)}$$

This kinetic power law equation is frequently used in engineering literature, however the constants don't have a physical meaning and must be determined experimentally.
The nucleation rate B_{hom} has to be converted into a size distribution around the critical nuclei size. In the simplest case this is done using a uniform distribution within a fixed size interval ΔL around the critical size:

$$J(L,S) = \begin{cases} \dfrac{B_{\text{hom}}(S)}{\Delta L} & for \quad 2r_{cr} - \dfrac{\Delta L}{2} < L < 2r_{cr} + \dfrac{\Delta L}{2} \\ 0 \quad otherwise \end{cases} \qquad \textbf{(2-16)}$$

The required critical nuclei radius is obtained from Eqn. 2-9:

$$r_{cr} = \frac{2 \cdot \sigma \cdot V_m}{\nu_D \cdot k \cdot T \cdot \ln S} \qquad \textbf{(2-17)}$$

2.3.3 Molecular Growth

The growth of the particles is a complex phenomenon, which consists of a series of sub-steps, as described in the work of Mersmann et al. [11]. It can be described as a convective/diffusive transport of molecules to the interface and a subsequent surface integration step.
For the estimation of the growth rate it is important to separate surface integration limited and transport limited growth. In the case of small supersaturations, as often encountered in crystallization, the system is near equilibrium and, due to the principle of energy minimization, the crystal surfaces are smooth and homogeneous. This typically causes a surface integration limited growth rate. In crystallization processes at high supersaturations, inhomogeneous crystal surfaces are present and lots of possibilities for surface integration exist. In this case the growth rate is typically limited by the transport of molecules to the interface (see Mersmann et al. [11]). Thus, in previous simulations found

in the literature the surface integration step was considered to be instantaneous (see, e.g., Gavi et al. [5]).

For transport limited growth, the growth rate can be calculated based on a Sherwood number. The nanoparticles are assumed to have the same velocity than the liquid, meaning their Reynolds number is zero. Assuming their shape as spherical, the Sherwood number is equal to 2. This is the theoretical lower limit for mass transfer due to diffusion around a sphere in a static fluid. The Sherwood number Sh can be used to calculate the diffusive mass transfer rate and subsequently the growth rate, defined as the time derivative of the particle size (see Gavi et al. [4]):

$$G(L,S) = \frac{\partial L}{\partial t} = \frac{2}{L \cdot c_S} Sh \cdot D \cdot c^* \cdot (S-1) \qquad \textbf{(2-18)}$$

It was shown by various authors (e.g., Stahl et al. [14]), that growth rate dispersion, i.e., particles of identical size experience different growth rates given by a distribution function, is essential to correctly predict the size distribution of crystalline substances. Growth rate dispersion was not considered in this work as the organic precipitation product considered in this work is amorphous.

2.3.4 Aggregation

The aggregation terms of the PBE account for the combination of two particle size classes (i.e., binary collisions of particles). The aggregation rate for a certain particle size is calculated by integration over all collision partners. The quantity accounting for the kinetics of aggregation is the so-called aggregation kernel. It is defined as a rate constant for aggregation [11], analogous to the kinetics of a chemical reaction of 2nd order:

$$-\frac{dN}{dt} = \beta_{agg} \cdot N^2 \qquad \textbf{(2-19)}$$

The aggregation kernel is formulated as the product of a collision kernel β_{coll}, i.e., the collision frequency between two particles, and an aggregation efficiency α, which describes the probability, that a collision is successful. The latter is determined by the particle interaction forces [4].

$$\beta_{agg}(L_1, L_2) = \beta_{coll}(L_1, L_2) \cdot \alpha(L_1, L_2) \quad \textbf{(2-20)}$$

In principle there are two mechanisms of aggregation: the orthocinetic and the pericinetic aggregation. Pericinetic aggregation is caused by the Brownian motion of particles, which is the dominant mechanism for particles smaller than 1 micrometer. For larger particles the Brownian motion is negligible and the aggregation is dominated by hydrodynamics [11], this is called orthocinetic aggregation. Pericinetic aggregation was originally described by Smoluchowski [15], and the collision kernel is (see, e.g., Gavi et al. [4]):

$$\beta_{coll}(L_1, L_2) = \frac{2 \cdot k \cdot T}{3 \cdot \eta}(L_1 + L_2) \cdot \left(\frac{1}{L_1} + \frac{1}{L_2} \right) \quad \textbf{(2-21)}$$

For the description of the aggregation efficiency the so-called stability ratio W is used:

$$\alpha(L_1, L_2) = \frac{1}{W(L_1, L_2)} \quad \textbf{(2-22)}$$

The stability ratio takes into account the influence of the particle interaction forces on the aggregation and, if relevant, the viscous resistance of the continuous medium. The latter effects the approach of two particles and therefore reduces the aggregation efficiency. In the work of Rollie [16] the stability ratio for two differently sized particles is defined as follows:

$$W(L_1, L_2) = \left(\frac{L_1}{2} + \frac{L_2}{2} \right) \int_0^\infty \frac{B(a)}{\left(\frac{L_1}{2} + \frac{L_2}{2} + a \right)^2} \exp\left(\frac{\phi_{total}(a)}{k \cdot T} \right) da \quad \textbf{(2-23)}$$

Here, a is the particle surface to surface distance, B the hydrodynamic correction and ϕ_{total} the total interaction potential. The hydrodynamic correction B accounts for the aggregation resistance due to the viscous influence of the fluid. It is defined as the ratio of a particle diffusity for an infinitely diluted system to the actual particle diffusity. The following

empirical correlation is typically used to estimate the hydrodynamic correction, where the value of the fluid viscosity does obviously not appear [16]:

$$B(a) \approx \frac{6\left(\frac{a}{r*}\right)^2 + 13\left(\frac{a}{r*}\right) + 2}{6\left(\frac{a}{r*}\right)^2 + 4\left(\frac{a}{r*}\right)} \qquad \textbf{(2-24)}$$

$$\text{with } r* = \frac{L_1 \cdot L_2}{L_1 + L_2} \qquad \textbf{(2-25)}$$

The total interaction potential energy ϕ_{total} is a superposition of the potentials resulting from the attractive Van der Waals forces and the electrostatic repulsion forces [16]:

$$\phi_{total}(a) = \phi_{VdW}(a) + \phi_{el}(a) \qquad \textbf{(2-26)}$$

Rollie [16] describes the Van der Waals potential ϕ_{VdW} for two particles with radii r_1 and r_2 by the Hamaker model for the Van der Waals interaction energy, where A is the Hamaker constant:

$$\phi_{VdW}(a) = -\frac{A}{6}\left(\frac{2r_1r_2}{a^2 + 2a(r_1 + r_2)} + \frac{2r_1r_2}{a^2 + 2a(r_1 + r_2) + 4r_1r_2} + \ln\left(\frac{a^2 + 2a(r_1 + r_2)}{a^2 + 2a(r_1 + r_2) + 4r_1r_2}\right)\right) \qquad \textbf{(2-27)}$$

The electrostatic potential ϕ_{el} describes the effect of the ionic double layer. Schwarzer and Peukert [17] used an equation based on the Gouy-Chapman model, which is valid for surface potentials up to 100mV. For two different particles with radii r_1 and r_2, but equal surface potentials ψ_P, the electrostatic interaction potential can be calculated according to:

$$\phi_{el}(a) = 128\pi \cdot \frac{c_t \cdot N_A \cdot k \cdot T}{\kappa^2} \cdot \tanh^2\left(\frac{z \cdot e \cdot \psi_P}{4 \cdot k \cdot T}\right) \cdot \frac{r_1 \cdot r_2}{r_1 + r_2 + a} \cdot \exp(-\kappa \cdot a) \quad \textbf{(2-28)}$$

Here, e is the unit charge and the concentration c_t is the total concentration of all ions, meaning $c_t = c_1 + c_2$. The Debye length κ and the ionic strength I are defined as follows:

$$\kappa = \sqrt{\frac{2 \cdot e^2 \cdot N_A}{\varepsilon_0 \cdot \varepsilon_r \cdot k \cdot T} \cdot I} \quad \textbf{(2-29)}$$

$$I = \frac{1}{2} \sum_{i=ions} c_i \cdot z_i^2 \quad \textbf{(2-30)}$$

The surface potential of the particles can be approximated by the following relation of Gavi et al. [4]. The surface potential in this approximation is determined by one type of ions, i.e., the so-called “potential determining ions” (PDI):

$$\psi_P = \frac{k \cdot T}{z \cdot e} \cdot \ln\left(\frac{c_{PDI}}{c_{PDI}^{\ pzc}}\right) \quad \textbf{(2-31)}$$

Here, c_{PDI} is the concentration of PDI and c_{PDI}^{pzc} is the concentration of PDI at the point of zero charge (pzc). The charge number z cancels out after the substitution of ψ_P in Eqn. 2-28.

2.4 Solution Methods for the PBE

The PBE used in this work, i.e., with nucleation, growth and aggregation, is a partial integro-differential equation, even in the well-mixed case, where the spatial derivatives are zero. In addition, it is coupled with the species balance equations for the liquid phase, i.e., partial differential equations in the case of a spatially resolved model, and ordinary differential equations in the well-mixed case.

There exist several approaches to solve the PBE, and here we focus on the most popular numerical methods. An overview about the solution methods is provided in Paschedag [2] and Marchisio et al. [18].

2.4.1 Classes Methods

The most obvious approach to solve a PBE is to discretize the internal coordinate analogous to the external coordinates, i.e. to define discrete particle size classes. The discretization could be equidistant or non-equidistant and is called classes method (CM). Moreover, the CM could be adaptive, which is favorable for strongly changing distributions as in the case of precipitation processes. Unfortunately, in combination with CFD the adaptive (internal) discretization cannot be used, because every computational cell requires the same internal discretization [2].

For more than one internal coordinate the computational effort for CM increases strongly with the number of internal coordinates. In practice, the application of CM for more than one internal coordinate is only useful for calculations without external coordinates. Thus, the CM is prohibitively expensive for CFD [2].

The main disadvantage of CM is the large number of classes required for good accuracy. For every single class a scalar transport equation has to be solved. Especially in the case of precipitation processes, where the particle size distribution changes strongly due to homogeneous nucleation and growth, the number of classes has to be large for sufficiently accurate results [18].

2.4.2 Monte-Carlo-Methods

Monte-Carlo-Simulations are based on the modeling of random events. There is no discretization required, but the evolution of the particle population is calculated based on discrete random events. These events occur within a predefined probability. In this type of statistical calculations, a sufficiently large number of particles has to be considered and a very small time-step has to be used. This makes the method numerically expensive [2]. Especially for CFD, where spatial inhomogeneities are considered, the required particle number would be inacceptable high [18].

The consideration of more than one internal coordinate is relatively simple in Monte-Carlo-Simulations and the computational demand increases only moderately with the number of internal coordinates. Also, as the history of the particles is known, a realistic modeling of breakage and particle morphology is possible. Due to the high numerical effort a combination of Monte-Carlo method for solving the PBE with CFD is not feasible today. However, the coupling of Monte-Carlo methods with Lattice-Boltzmann

simulations seems feasible, because both approaches are based on statistical considerations of particles [2].

2.4.3 Methods of Moments

Methods of moments for solving the PBE are frequently used in combination with CFD. In the standard method of moments (SMM), the internal coordinate is integrated, and the particle size distribution is represented by its moments m_j [11], whereas the j^{th} moment of a number density distribution n(L) is defined as [2]:

$$m_j = \int_0^\infty n \cdot L^j dL \qquad \textbf{(2-32)}$$

Some moments have a physical meaning: the 0^{th} moment is proportional to the particle number, the second moment is proportional to the total surface and the third moment is proportional to the total volume of the particle population. A wide range of particle size distributions is well described by only three or four moments. For each moment a scalar transport equation has to be solved, hence the computational demand is much lower than using the CM [2].

A main disadvantage of the SMM is, that size-independent terms for growth and aggregation have to be used to close the system. To avoid this, the quadrature method of moments (QMOM) has been developed, which is based on the same idea as the SMM. QMOM overcomes the closure problem by using an *ad hoc* quadrature approximation. QMOM is a presumed PSD method, where the PSD is represented by a superposition of delta functions. A drawback of QMOM is, that it cannot represent bimodal PSD. Its main advantage is the combination of good accuracy and relatively low computation cost, which makes it ideal for coupling with CFD [18].

2.4.4 Selection of an Appropriate Solution Method

In this work no spatial resolution was considered for the solution of the PBE, i.e., it is not required to choose a solution method which can be coupled with CFD. There is only one internal coordinate considered (i.e., the particle size), thus the most obvious approach is the CM. Its relatively high computational effort is no handicap in this work.

Monte Carlo methods are more complex and do not provide any advantage compared to CM, because the particle morphology is not considered in this work. Methods of moments are also more complex than CM, and their lower computation effort is no benefit for this work. Moreover, the moment transformation causes a loss of information, thus the CM with a sufficiently high number of classes is more accurate than a QMOM with three or four moments, as typically used. Thus, the classes method was used as solution method for the PBE in this work.

2.5 Description of the Continuous Phase

2.5.1 Basic Flow Considerations

In the precipitation process we are facing a two phase flow. The continuous phase is liquid and the dispersed phase is constituted by the solid particles. In multiphase flow problems we usually have to solve one set of flow equations for each phase in order to get the complete flow field. In the case of nanoparticles, the influence of the precipitated particles on the flow field is negligible, because the particles are smaller than the Kolmogorov length [10]. Also, the volume fraction of the solid phase, as well as the mass loading are small (i.e., in the order of 10^{-4}) and consequently there is no significant momentum transfer between the two phases. Hence, there is only the need to calculate the liquid flow field, which means solving a single phase flow.

Furthermore, for computational fluid dynamic (CFD) simulations of turbulent flow an appropriate approach has to be chosen. Thus, these simulations are based on the (i) Reynolds-averaged-Navier-Stokes equations, the (ii) filtered Navier-Stokes equations, or the (iii) fully resolved Navier-Stokes equations to reconstruct chaotic turbulent fluid motion. On the one hand the simulation should allow to study problems of industrial relevance, on the other hand it has to be detailed enough to yield physically reasonable results for mixing. The latter is especially critical for precipitation processes, as micro mixing, i.e., mixing down to the molecular scale, is essential for the process. The filtered Navier-Stokes equation, i.e., the so-called Large Eddy Simulation approach, is known to accurately predict flow and species transport in micro reactors. This is supported by recent studies of Marchisio [19] and Radl et al. [20].

2.5.2 Micromixing Models

Micromixing effects play a major role in the outcome of a precipitation process [21]. Mixing down to the molecular scale is the precondition for precipitation processes, otherwise no supersaturation is generated. Due to the nature of precipitation processes, the resulting PSD is sensitive to variations in the concentration field, as shown by Schwarzer et al. [10]. According to Gavi et al. [4] and Baldyga et al. [22] the kinetics of precipitation processes should be applied to completely micromixed regions. This state of micromixedness depends on the local fluid motion and molecular diffusion. Consequently micromixing has to be described by an appropriate model, if the resolution of the flow model is not sufficiently fine. For engineering applications this is typically the case.

Before implementing the PBE in a CFD-simulation, it is useful to calculate systems with concentrated parameters (e.g., a well-mixed reactor). Their reduced complexity enables to get first results within a fraction of the time needed for a CFD model. Typically, general trends of the real-world system can be already qualitatively correctly predicted. The simplest case is to calculate the process in a well-mixed system. This can be easily extended by a mixing model. Therefore, a model to mimick concentration variances for a well-mixed system is required. Some of the most popular models for these purposes are detailed here:

a) Population Balance for Fluid Elements - the Coalescence Dispersion Model

Ulbert et al [23] used a concentration distribution p(c,t) to describe the microlevel segregation in a crystallizer, that is perfectly mixed on the macrolevel. The (average) macrolevel concentration is written as:

$$\langle c \rangle(t) = \frac{\int_{c_S}^{c_{\max}} c \cdot p(c,t) \cdot dc}{\int_{c_S}^{c_{\max}} p(c,t) \cdot dc} \qquad \textbf{(2-33)}$$

The concentration distribution is calculated by a PBE for fluid elements (Eqn. 2-34) including the so-called coalescence-dispersion model to represent the micromixing process. R(t) is the rate of concentration degradation due to nucleation and growth, S(t) represents the production rate of solid phase, and K is a micro mixing parameter.

$$\frac{\partial p(c,t)}{\partial t} + R(t)\frac{\partial p(c,t)}{\partial c} = -\frac{S(t)}{\upsilon_f}\delta(c) + \frac{p_{in}(c,t) - p(c,t)}{\tau} + K\left(2\int_{c_S}^{c_{\max}}\int_{c_S}^{c_{\max}} p(c',t)\cdot p(c'',t)\cdot\delta\left(\frac{c'+c''}{2} - c\right)\cdot dc'\cdot dc'' - p(c,t)\int_{c_S}^{c_{\max}} p(c',t)\cdot dc'\right) \quad \textbf{(2-34)}$$

The mean rates of nucleation $\tilde{B}(t)$ and growth $\tilde{G}(t)$, which are used in the PBE for the particle population, are calculated by integration over the concentration distribution:

$$\tilde{B}(t) = \frac{\int_{c_S}^{c_{\max}} B(c,c_S)\cdot p(c,t)\cdot dc}{\int_{c_S}^{c_{\max}} p(c,t)\cdot dc} \quad \textbf{(2-35)}$$

$$\tilde{G}(t) = \frac{\int_{c_S}^{c_{\max}} G_0(c,c_S)\cdot p(c,t)\cdot dc}{\int_{c_S}^{c_{\max}} p(c,t)\cdot dc} \quad \textbf{(2-36)}$$

Thus, two population balance equations have to be solved to model the precipitation process.

b) ADCR Model

The axial dispersion-coalescence/redispersion (ADCR) model, developed by Lakatos [24], is a relatively new approach. In this model macromixing is described by an axial dispersion model. Micromixing is described by the coalescence and redispersion of fluid elements. This model is similar to the coalescence dispersion model described in a), however the source terms of the population balance equation are not equal. The model was verified with experimental data for the case of a tubular reactor [24].

c) Presumed Probability Density Functions

Baldyga et al. [25] used a Beta probability density function (PDF) to describe the microlevel segregation of the concentration field. This is similar to the approach of Ulbert et al., with the exception, that the distribution of the concentration is assumed to be a Beta function. Hence, the population balance equation for the concentration distribution is not solved. Unfortunately, presumed PDF approaches were only used to describe chemical

reactions without subsequent phase change. It is currently unclear, how to incorporate nucleation in a presumed PDF approach. The nucleation rate depends on the supersaturation, while the presumed PDF describes the concentrations. The dependency of the nucleation rate on the concentrations is much more complex (i.e., strongly nonlinear) compared to the situation where only a reaction occurs. This makes it difficult to describe the precipitation process with a presumed concentration-PDF.

d) Segregated Feed Model

Zauner et al. [26] tried to use a so called segregated feed model, where instead of a concentration distribution (which accounts for every value of the local concentration) two well-mixed compartments are used to model spatial concentration gradients. The first compartment is called reaction plume and is located near the feeding point of a batch stirred tank reactor. The second compartment represents the remaining part of the reactor and is called bulk. The total volume V_{tot} is the sum of the compartment volumes V_f and V_b:

$$V_{tot} = V_f + V_b \tag{2-37}$$

The compartment volumes are not constant. There is a permanent feed flow Q_f into the system, which increases the total volume, and a convective exchange flow from the reaction plume to the bulk. The latter is characterized by a mesomixing time scale t_{meso}:

$$\frac{dV_{tot}}{dt} = Q_f \tag{2-38}$$

$$\frac{dV_f}{dt} = Q_f - \frac{V_f}{t_{meso}} \tag{2-39}$$

The total transfer of a component j between the two compartments $u_{j,fb}$ is the sum of the convective contribution, determined by the mesomixing time scale t_{meso} and the diffusive contribution, determined by the micromixing time scale t_{micro}:

$$u_{j,fb} = \frac{V_f \cdot c_{j,f}}{t_{meso}} + \frac{V_f \cdot (c_{j,f} - c_{j,b})}{t_{micro}} \quad \textbf{(2-40)}$$

The mass balances for a species j in the two compartments are:

$$\frac{d(V_f c_{j,f})}{dt} = r_{j,f} V_f + Q_f c^0_{j,f} - u_{j,fb} \quad \textbf{(2-41)}$$

$$\frac{d(V_b c_{j,b})}{dt} = r_{j,b} V_b + u_{j,fb} \quad \textbf{(2-42)}$$

Here $r_{j,f}$ and $r_{j,b}$ are concentration sink terms due to nucleation and growth. In both compartments the population balance equation for the PSD has to be solved. After stopping the feed flow, the feed volume decreases to zero (see Eqn. 2-39), while the bulk volume gets equal to the total volume, which contains the final particles after the process is finished.

e) Engulfment Model

The engulfment model, initially developed by Baldyga et al. [27], describes the time dependent mixing of two precursors. In principle, the model describes mixing as a batch process. It can be applied to a continuous mixing process by following a feed volume portion in a Lagrangian manner [14]. The time dependent mixing state of the volume portion is characterized by the mesomixed volume fraction X_{me} and the micromixed volume fraction X_{mi}. Ståhl et al. [14] compared the segregated feed model described in d) with the engulfment model. The latter showed a better prediction than the segregated feed model.

As shown in Figure 2-2, the initial state is a macromixed one “i”, i.e., there are regions of pure precursors distributed in the reactor. The micromixed volume is defined as sum of regions, where A and B are mixed down to the molecular scale, shown by continuously colored regions “iv” in Figure 2-2 (notice, that there is micromixed volume also between the mesomixed regions). Mesomixing is the precondition for micromixing, hence the mesomixed volume is defined as the micromixed volume plus the sum of regions containing pure B at any scale between the macroscale and the molecular scale (see the

dotted regions “iii” in Figure 2-2). The difference between the total volume and the mesomixed volume are regions where only pure B in a macromixed state exists “ii”.

For the mathematical description, the considered feed volume portion is divided into two parts, each of them assumed to be well mixed. The first part is the micromixed volume. It contains the species with time dependent concentrations c_i and chemical reactions can occur in this part. The second part is the difference between the total volume and the micromixed volume, called non-micromixed volume. Here only pure B exists in macromixed and mesomixed states. It contains the constant concentrations $\langle c_i \rangle$, chemical reactions are impossible in this region.

With increasing time the micromixed volume increases by consuming parts of the non-micromixed volume. Finally, the micromixed volume reaches the value of the total volume of the feed portion and the mixing process is complete.

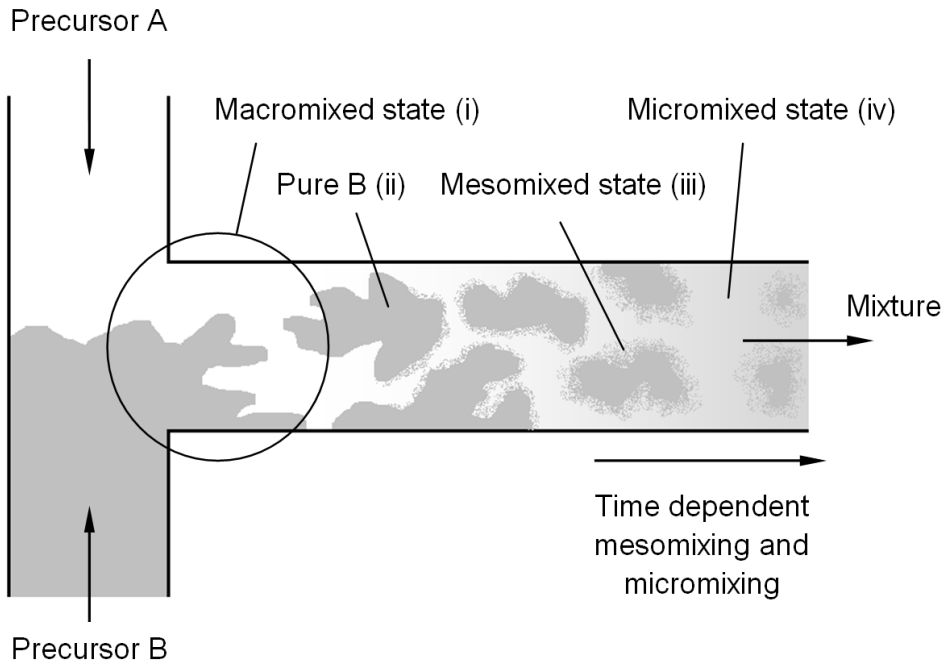


Figure 2-2: Mixing process as described by the engulfment model in a continuous mixer.

Model equations to describe the micromixed volume V(t), the micromixed volume fraction $X_{mi}(t)$ and the mesomixed volume fraction $X_{me}(t)$ have been established. Eqn. 2-45 is mentioned for completeness, it is equivalent to Eqn. 2-43 with the condition $X_{mi}(t) = V(t)/V_{total}$, where V_{total} is the total volume of the feed portion. For continuous processes the volumes V(t) and V_{total} can be replaced by volumetric flow rates.

$$\frac{dX_{mi}}{dt} = E \cdot \left(1 - \frac{X_{mi}}{X_{me}}\right) \cdot X_{mi} \quad \textbf{(2-43)}$$

$$\frac{dX_{me}}{dt} = \frac{1}{t_{me}} \cdot \left(1 - X_{me}\right) \cdot X_{me} \quad \textbf{(2-44)}$$

$$\frac{dV}{dt} = E \cdot \left(1 - \frac{X_{mi}}{X_{me}}\right) \cdot V \quad \textbf{(2-45)}$$

With the engulfment constant E, being the inverse of the micromixing time t_{mi}, and the mesomixing time t_{me} as parameters [14]:

$$t_{mi} = \frac{1}{E} = 12.7 \cdot \sqrt{\frac{\nu}{\varepsilon}} \quad \textbf{(2-46)}$$

$$t_{me} = 2 \cdot \left(\frac{d^2}{\varepsilon}\right)^{1/3} \quad \textbf{(2-47)}$$

Here, ν is the cinematic viscosity, ε the mean specific power input and d a length scale, wherefore often the inlet diameter of the reactor is used. The concentration of a species i c_i in the micromixed volume is described as:

$$\frac{dc_i}{dt} = E\left(1 - \frac{X_{mi}}{X_{me}}\right) \cdot \left(\langle c_i \rangle - c_i\right) + r_i \quad \textbf{(2-48)}$$

Here r_i is a reaction rate and $\langle c_i \rangle$ is the concentration of the species i in the non-micromixed volume, which is time independent.

For the initial conditions the macromixed state in Figure 2-2 has to be considered. Here the micromixed, and also the mesomixed volume, are equal to the volume of the precursor A, thus the initial conditions for the volume fractions are:

$$X_{mi}(t=0) = X_{me}(t=0) = \frac{V(t=0)}{V_{total}} = \frac{V_A}{V_A + V_B} \quad \textbf{(2-49)}$$

2.5.3 Selection of an Appropriate Mixing Model

In this work it was sufficient to choose a relatively simple mixing model, which is able to predict general trends. The population balance model for fluid elements described in a) and the ADCR Model described in b) are too complex for this intention, because they would require to solve a second PBE for the fluid elements. The application of a presumed PDF approach to a precipitation process is currently not possible, as described in c). A comparison of the segregated feed model and the engulfment model in combination with a precipitation process (described in d) and e)) showed, that the engulfment model was able to give better predictions [14]. Hence, the engulfment model was used as mixing model in this work.

3 Parameters and Basic Engineering

3.1 Materials

The two components forming the solid particles in this precipitation process are a polyacrylic-acid and a protamine. Unfortunately, the molecular structures of these materials are not exactly known. Thus, the parameters used for the calculations are based on the following assumptions about the structures of the molecules.

The used polyacrylic-acid is assumed to contain 51 acrylic-acid groups, 3 cysteine-acid groups and 3 cysteine-sulfide groups. The pK_a values are: acrylic-acid 4.26 [28], cysteine-acid 1.9 [29] and cysteine-sulfide 8.4 [29]. The molar mass of the polyacrylic-acid is assumed to be 5400 g/mol.

The used protamine is assumed to contain 22 guanidinium groups with a pK_a value of 12.1 [30]. The molar mass of protamine is assumed to be 4300 g/mol.

3.1.1 Estimation of the Diffusion Coefficients

The diffusion coefficient D_i for a small spherical particle in a liquid with the viscosity η and the temperature T can be calculated from the hydrodynamic radius $r_{h,i}$ using the Stokes-Einstein equation [31]:

$$D_i = \frac{k \cdot T}{6\pi \cdot \eta \cdot r_{h,i}} \qquad \textbf{(3-1)}$$

Here, k is the Boltzmann constant. According to Lochmann et al. [31] the hydrodynamic radius for the protamine is 1.35 nm. This corresponds to a diffusion coefficient in water at 22°C of $1.60 \cdot 10^{-10}$ m²/s (η = 0.001 Pas [32]).

Unfortunately, the diffusion coefficient for the polyacrylic-acid has not been measured, nor is its hydrodynamic radius known exactly. Due to the molar masses, polyacrylic-acid is slightly larger than the protamine, and its hydrodynamic radius has been assumed to be 1.4 nm. This corresponds to a diffusion coefficient of $1.54 \cdot 10^{-10}$ m²/s.

Furthermore, for the calculation of the nucleation rate (Eqn. 2-14) a mean diffusion coefficient D is required. It is calculated by the molar concentrations of the species c_1 (i.e., polyacrylic-acid) and c_2 (i.e., protamine):

$$D = \frac{c_1 \cdot D_1 + c_2 \cdot D_2}{c_1 + c_2} \qquad \textbf{(3-2)}$$

All material parameters used for the calculations are summarized in Table 3-1.

	Polyacrylic-acid	Protamine
Molecular weight [g/mol]	5400	4300
Number of acidic groups		
Acrylic-acidic group ($pK_a = 4.26$)	51	0
Cysteine-acidic group ($pK_a = 1.9$)	3	0
Cysteine-sulfide group ($pK_a = 8.4$)	3	0
Number of basic groups		
Guanidinium group ($pK_a = 12.1$)	0	22
Hydrodynamic radius [nm]	1.4	1.35
Diff. coefficient in H_2O at 22°C [m²/s]	$1.54 \cdot 10^{-10}$	$1.60 \cdot 10^{-10}$
Initial mass concentration [g/l]	0.2	0.6
Initial molar concentration [mol/l]	$3.7 \cdot 10^{-5}$	$1.4 \cdot 10^{-4}$
Density of the solid material [kg/m³]	1400	

Table 3-1: Parameters for the polyacrylic-acid/protamine system.

3.1.2 Process Parameters

The precipitation process is operated at room temperature. For the calculations a constant temperature of 22°C was assumed. Also, the enthalpy change due to the reaction between the protamine and the polyacrylic-acid and the subsequent phase change was assumed to be negligible.

The precursor concentrations are shown in Table 3-1, and correspond to a ratio of the mass concentrations of polyacrylic-acid to protamine of 1:3. This is to get an excess of protamine to stabilize the final particles and prevent further aggregation.

In Section 3.1.3 the change of the concentrations during the process, i.e., the formation of particles, is considered. The concentration change of polyacrylic-acid and protamine depend on each other, hence it is beneficial to define the conversion X (i.e., the relative amount of precipitated polyacrylic-acid) as:

$$c_1 = c_{10} \cdot (1 - X) \quad \textbf{(3-3)}$$

Assuming a constant molar ratio of the components in the product, i.e., constant charge numbers of the ions z_1 and z_2 during the process (see Section 3.1.3), the conversion also determines the actual concentration of protamine c_2 with the initial concentration c_{20}:

$$c_2 = c_{20} - X \cdot c_{10} \cdot \frac{z_1}{z_2} \quad \textbf{(3-4)}$$

3.1.3 Charge Numbers

Polyacrylic-acid and protamine are organic macromolecules with a number of weak acidic and basic groups, as described in Section 3.1. The actual charge number of the molecules depends on the actual pH-value. The numbers of groups per molecule and their pK_a-values are known (see Table 3-1). Using this information and the actual species concentrations, the pH-value and the molecule charge numbers can be calculated. The following chemical reactions have to be considered:

Dissociation of cysteine-acidic group: $\mathbf{R_1\text{-}COOH + H_2O \leftrightarrow R_1\text{-}COO^- + H_3O^+}$

Dissociation of cysteine-sulfide group: $\mathbf{R_2\text{-}SH + H_2O \leftrightarrow R_2\text{-}S^- + H_3O^+}$

Dissociation of acrylic-acidic group: $\mathbf{R_3\text{-}COOH + H_2O \leftrightarrow R_3\text{-}COO^- + H_3O^+}$

Protonation of guanidinium-cationic group: $\mathbf{R_4\text{-}NH_2^+ + H_2O \leftrightarrow R_4\text{-}NH + H_3O^+}$

Dissociation of water: $\mathbf{2\ H_2O \leftrightarrow H_3O^+ + OH^-}$

Note, that the reaction for the base guanidinium is written as acidic-reaction in order to apply the pK_a-value. Hence, the OH^- Ions don't appear in this reaction equation. The equilibrium constants $K_{()}$ are defined by the concentrations of the reaction partners $c_{()}$ as follows (for the subscripts see the list of symbols):

$$K_{CysCOOH} = \frac{c_{CysCOO-} \cdot c_{H3O+}}{c_{CysCOOH}} \quad \textbf{(3-5)}$$

$$K_{CysSH} = \frac{c_{CysS-} \cdot c_{H3O+}}{c_{CysSH}} \quad \textbf{(3-6)}$$

$$K_{AACOOH} = \frac{c_{AACOO-} \cdot c_{H3O+}}{c_{AACOOH}} \quad \textbf{(3-7)}$$

$$K_{GuaNH} = \frac{c_{GuaNH} \cdot c_{H3O+}}{c_{GuaNH2+}} \quad \textbf{(3-8)}$$

$$K_W = c_{OH-} \cdot c_{H3O+} \quad \textbf{(3-9)}$$

To get the equilibrium constants $K_{()}$, the definition of the pK_a values is used:

$$K_i = 10^{-pK_{a,i}} \quad \textbf{(3-10)}$$

Furthermore, the following molar balance equations have to be fulfilled, where n_{Cys} is the number of cysteine groups per molecule, n_{AA} the number of acrylic-acid groups per molecule and n_{Gua} the number of guanidinium-cationic groups per molecule:

$$c_{CysCOO-} + c_{CysCOOH} = c_1 \cdot n_{Cys} \quad \textbf{(3-11)}$$

$$c_{CysS-} + c_{CysSH} = c_1 \cdot n_{Cys} \quad \textbf{(3-12)}$$

$$c_{AACOO-} + c_{AACOOH} = c_1 \cdot n_{AA} \quad \textbf{(3-13)}$$

$$c_{GuaNH} + c_{GuaNH2+} = c_2 \cdot n_{Gua} \quad \textbf{(3-14)}$$

Additionally the total charge balance equation has to be guaranteed:

$$c_{CysCOO-} + c_{CysS-} + c_{AACOO-} + c_{OH-} = c_{GuaNH2+} + c_{H3O+} \quad \textbf{(3-15)}$$

This is a system of 10 algebraic equations and 10 unknowns. Substituting the concentrations of neutral groups ($c_{CysCOOH}$, c_{CysSH}, c_{AACOOH} and c_{GuaNH}, see Eqn. 3-5 to Eqn. 3-8) by the concentrations of charged groups ($c_{CysCOO-}$, c_{CysS-}, c_{AACOO-} and $c_{GuaNH2+}$, see Eqn. 3-11 to Eqn. 3-14) yields:

$$c_{CysCOO-} = \frac{n_{Cys} \cdot c_1 \cdot K_{CysCOOH}}{c_{H3O+} + K_{CysCOOH}} \quad \textbf{(3-16)}$$

$$c_{CysS-} = \frac{n_{Cys} \cdot c_1 \cdot K_{CysSH}}{c_{H3O+} + K_{CysSH}} \quad \textbf{(3-17)}$$

$$c_{AACOO-} = \frac{n_{AA} \cdot c_1 \cdot K_{AACOOH}}{c_{H3O+} + K_{AACOOH}} \quad \textbf{(3-18)}$$

$$c_{GuaNH2+} = \frac{n_{Gua} \cdot c_2 \cdot c_{H3O+}}{c_{H3O+} + K_{GuaNH}} \quad \textbf{(3-19)}$$

All quantities except c_{H3O+} in the charge balance Eqn. 3-15 can be substituted by Eqn. 3-16 to Eqn. 3-19 and Eqn. 3-9. Thus, Eqn. 3-15 can be modified to yield an expression, which is here abbreviated with F (which has to be zero then):

$$F = \frac{n_{Cys} \cdot c_1 \cdot K_{CysCOOH}}{c_{H3O+} + K_{CysCOOH}} + \frac{n_{Cys} \cdot c_1 \cdot K_{CysSH}}{c_{H3O+} + K_{CysSH}} + \frac{n_{AA} \cdot c_1 \cdot K_{AACOOH}}{c_{H3O+} + K_{AACOOH}} + \frac{K_W}{c_{H3O+}} - \frac{n_{Gua} \cdot c_2 \cdot c_{H3O+}}{c_{H3O+} + K_{GuaNH}} - c_{H3O+} = 0 \quad \textbf{(3-20)}$$

In this function, the only unknown quantity is c_{H3O+}, which is related to the pH-value:

$$pH = -\log(c_{H3O+}) \quad \textbf{(3-21)}$$

Therefore, the pH-value of the system can be calculated when solving for c_{H3O+} in Eqn. 3-20. With the value for c_{H3O+}, the number of charged groups per molecule $z_{()}$ can be

calculated by using Eqn. 3-16 to Eqn. 3-19 and $z_{()}=c_{()}/c_1$ for the polyacrylic-acid, and $z_{()}=c_{()}/c_2$ for protamine:

$$z_{CysCOO-} = \frac{n_{Cys} \cdot K_{CysCOOH}}{c_{H3O+} + K_{CysCOOH}} \quad \textbf{(3-22)}$$

$$z_{CysS-} = \frac{n_{Cys} \cdot K_{CysSH}}{c_{H3O+} + K_{CysSH}} \quad \textbf{(3-23)}$$

$$z_{AACOO-} = \frac{n_{AA} \cdot K_{AACOOH}}{c_{H3O+} + K_{AACOOH}} \quad \textbf{(3-24)}$$

$$z_{GuaNH2+} = \frac{n_{Gua} \cdot c_{H3O+}}{c_{H3O+} + K_{GuaNH}} \quad \textbf{(3-25)}$$

The total charge numbers for polyacrylic-acid z_1 and protamine z_2 are the sum of the associated numbers of charged groups per molecule $z_{()}$:

$$z_1 = -(z_{CysCOO-} + z_{CysS-} + z_{AACOO-}) \quad \textbf{(3-26)}$$

$$z_2 = z_{GuaNH2+} \quad \textbf{(3-27)}$$

The System was solved in MATLAB R2008a (for the code, see Appendix B/I). For six different conversions (i.e., see Eqns. 3-3 and 3-4) between 0 and 1 the function F was calculated and the result are shown in Figure 3-1. Clearly, the conversion does not have any significant impact on F, and a close-up of the region around pH 10.6 (i.e., Figure 3-2) shows, that the zero of F is defined, and is unaffected by the conversion. This is because the solid phase is assumed to be uncharged. Hence, when increasing the conversion the charge sum of polyacrylic-acid ($c_{CysCOO-} + c_{CysS-} + c_{AACOO-}$) and protamine ($c_{GuaNH2+}$) in the liquid phase are reduced by the same amount (see Eqn. 3-15). Consequently, the values of c_{H3O+} and c_{OH-} remain the same, i.e., the pH-value doesn't change when the conversion is varied. Notice, the pH-value of the educts, i.e., the polyacrylic-acid and the protamine, are

of course not identical. However, in the mixed system, the pH-value does not change during precipitation. Also, the charge numbers of the involved molecules (which depend on the pH-value) are constant during the process. These results in the pH-value and charge numbers of molecules are shown in Table 3-2.

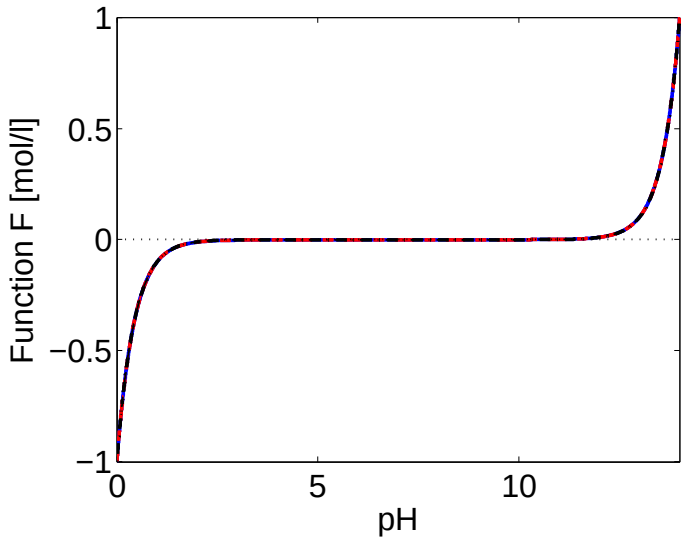


Figure 3-1: Function F (Eqn. 3-20) used for the determination of the pH-value.

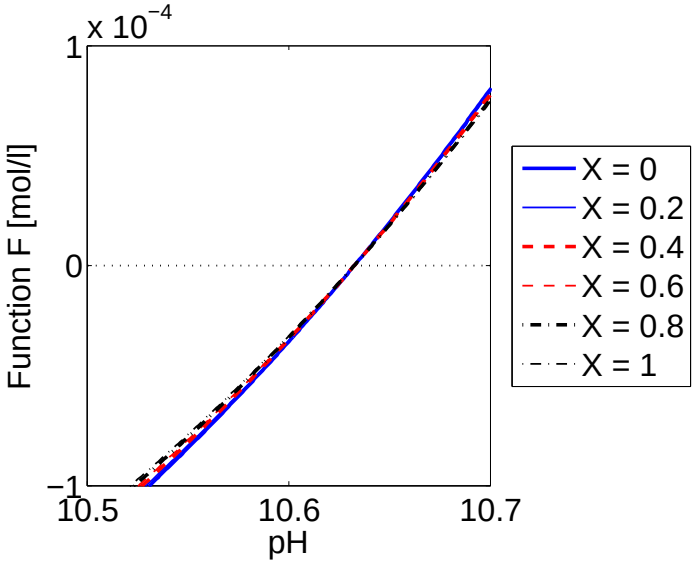


Figure 3-2: Close-up of function F around pH=10.6.

The number of charged groups as function of pH can be calculated via Eqn. 3-22 to Eqn. 3-25. These curves are shown in Figure 3-3, where the ordinate is scaled to unity (the absolute charge numbers can be calculated from Table 3-1. The charge numbers are not required to be integers, because they are average numbers over a large amount of

molecules. Clearly, with increasing pH value the acidic groups are dissociated (depending on their pKa value) and form charged polyacrylic-acid molecules. At a pH above ca. 10, the guanidinium cation is de-protonated, and loses its charge. When using a ratio between polyacrylic acid and protamine of 1:3, the resulting pH (i.e., pH 10.6) leads to practically completely dissociated acid molecules (which are strongly charged), while the deprotonation of the protamine is not complete.

pH =	10.633
Cysteine-acid $z_{CysCOO-}$ =	3.000
Cysteine-sulfide z_{CysS-} =	2.983
Acrylic-acid z_{AACOO-} =	51.000
Guanidinium $z_{CuaNH2+}$ =	21.274
Polyacrylic-acid z_1 =	-56.983
Protamine z_2 =	21.274

Table 3-2: Results for the pH-value and the charge numbers.

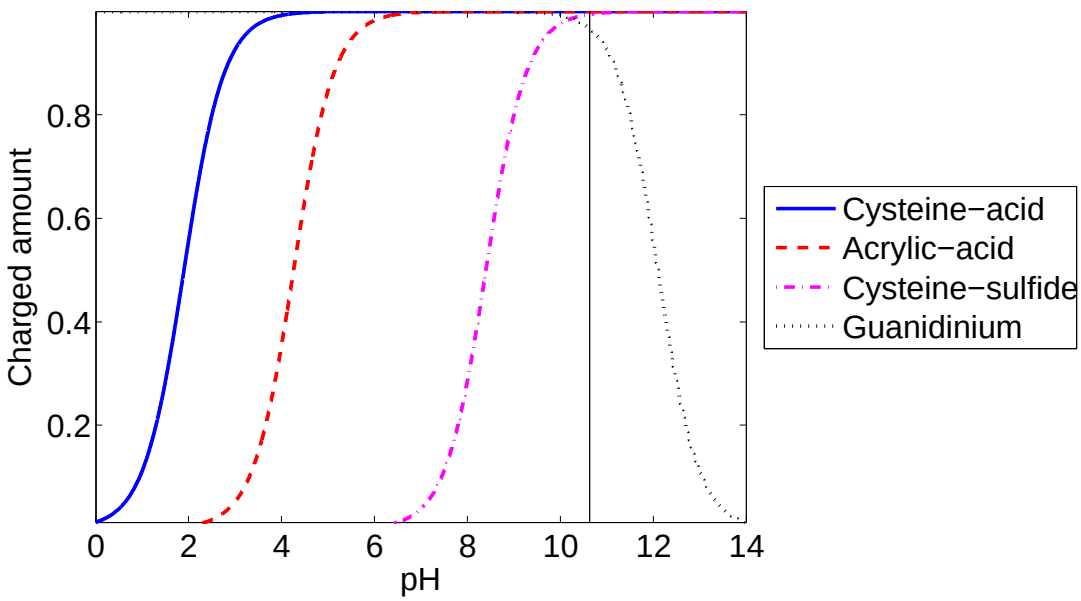


Figure 3-3: Amount of charged groups as function of the pH-value.

3.1.4 Solid Density, Solid Concentration and Molecular Volume

The solid density of the final particles ρ_S is assumed to be constant and is given in Table 3-1. The particle porosity is assumed to be zero, which was confirmed by experiments. The concentrations of the two components polyacrylic-acid and protamine in the solid phase c_{S1} and c_{S2} are calculated from the condition of electrical neutrality of the particle (Eqn. 3-28) and a mass balance (Eqn. 3-29, M_1 and M_2 are the molar masses):

$$c_{S1}z_1 + c_{S2}z_2 = 0 \quad \textbf{(3-28)}$$

$$c_{S1}M_1 + c_{S2}M_2 = \rho_S \quad \textbf{(3-29)}$$

$$c_{S1} = \frac{\rho_S z_2}{M_1 z_2 - M_2 z_1} \quad \textbf{(3-30)}$$

$$c_{S2} = -\frac{z_1}{z_2} c_{S1} \quad \textbf{(3-31)}$$

The total concentration in the solid phase c_S is the sum of the two component concentrations:

$$c_S = c_{S1} + c_{S2} \quad \textbf{(3-32)}$$

The molecular volume V_m, i.e., the mean volume per molecule in the solid phase, is calculated from the total solid concentration c_S as (where N_A is Avogadro's number):

$$V_m = \frac{1}{c_S N_A} \quad \textbf{(3-33)}$$

Data used for the mass balance calculations are shown in Table 3-3, and the results for the solid concentrations and the molecular volume are given in Table 3-4.

Polyacrylic-acid solution		
Molar mass of PAC	M_1 =	5400 g/mol
Charge number of PAC	z_1 =	-56.98
Initial mass conc. of PAC	c_{M1} =	0.2 g/l
Protamine solution		
Molar mass of protamine	M_2 =	4300 g/mol
Charge number of protamine	z_2 =	21.27
Initial mass conc. of protamine	c_{M2} =	0.6 g/l
Volume flow ratio	VFR = V_1/V_2 =	1
Particles		
Solid density	ρ_s =	1400 kg/m³
Expected particle diameter	L =	140 nm

Table 3-3: Data used for the mass balances.

Mixture		
Mixed mass conc. of PAC	$c_{M1mix} = c_{M1} \cdot VFR/(VFR+1)$=	0.1 g/l
Mixed mass conc. of protamine	$c_{M2mix} = c_{M2} / (VFR+1)$ =	0.3 g/l
Mixed molar conc. of PAC	$c_{1mix} = c_{M1mix} / M_1$ =	$1.85 \cdot 10^{-5}$ mol/l
Mixed molar conc. of protamine	$c_{2mix} = c_{M2mix} / M_2$ =	$6.98 \cdot 10^{-5}$ mol/l
Particles		
Mass of a single particle	$m_P = \rho_s \cdot L^3 \cdot \pi/6$	$2.01 \cdot 10^{-18}$ kg
Solid mass ratio prot./PAC	$w_{21} = -z_1/z_2 \cdot M_2/M_1$ =	2.133
Solid molar frac. of PAC	$x_1 = 1 / (1 - z_1/z_2)$ =	0.272
Solid molar frac. of protamine	$x_2 = 1 - x_1$ =	0.728
Mean solid molar mass	$M_S = M_1 \cdot x_1 + M_2 \cdot x_2$ =	4599 g/mol
Solid conc. of PAC	$c_{S1} = z_2 \cdot \rho_S / (z_2 M_1 - z_1 M_2)$ =	0.083 mol/l
Solid concentration of protamine	$c_{S2} = - c_{S1} \cdot z_1 / z_2$ =	0.222 mol/l
Total solid concentration	$c_S = c_{S1} + c_{S2} = \rho_S / M_S$ =	0.304 mol/l
Molecular volume	$V_m = 1/(c_S \cdot N_A)$ =	$5.46 \cdot 10^{-27}$ m³
Nanoparticle suspension		
Particle mass concentration	$c_{MP} = c_{M1mix} \cdot (1 + w_{21})$ =	0.313 g/l
Protamine excess concentration	$c_{M2,ex} = c_{M2mix} - c_{M1mix} \cdot w_{21}$ =	0.087 g/l
Particle number density	$N = c_{MP} / m_P$ =	$1.56 \cdot 10^{17}$ m^{-3}
Solid phase volume fraction	$\varepsilon_s = c_{MP} / \rho_s$ =	$2.24 \cdot 10^{-4}$

Table 3-4: Mass balance results.

3.1.5 Solubility

The experimental determination of the solubility of the product molecule (made up from polyacrylic-acid and protamine) turned out to be difficult. Thus, we assumed, that the solubility can be described by a solubility product, similar to an inorganic salt. In order to determine a value for the solubility, the influence of the equilibrium concentration c* on the nucleation rate was studied as shown in Chapter 4. Based on the results of this parameter study, the arbitrary value of 10^{-10} mol/l for c*, i.e., 10^{-20} mol^2/l^2 for the solubility product K_S, was chosen. To get a more precise value for K_S, the solubility has to be investigated experimentally using an appropriate technique.

3.2 Mass Balances

To test the plausibility of the simulation results, overall mass balances of the process have been calculated (see Figure 3-4, here c_{Mi} are the precursor mass concentrations in stream i, c_{Mimix} are the mixed mass concentrations, c_{imix} are the mixed molar concentrations, V_i are the precursor volume flow rates, c_{MP} is the mass concentration of the particles, N is the number density of the particles, c_{M2ex} the mass concentration of the protamine excess and ε_S the volume fraction of the solid phase). To simplify these considerations the process was divided into two sub steps, i.e., mixing and particle formation. Although they are running simultaneously in reality, they can be considered to be separated for the calculation of the mass balances.

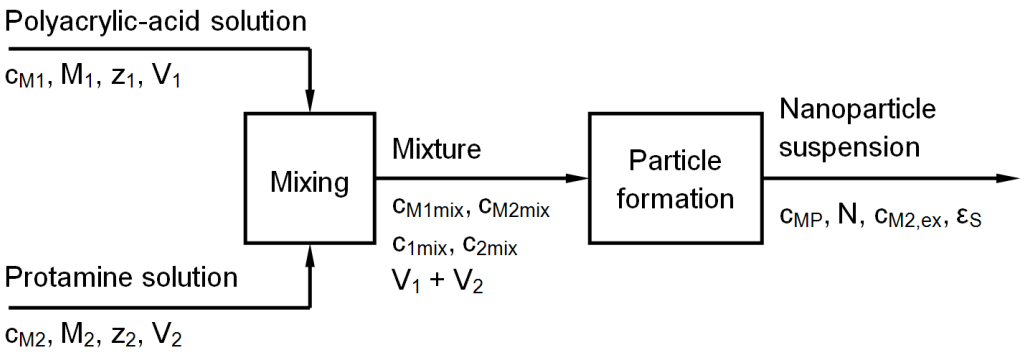


Figure 3-4: Basic flow sheet.

The data used for the calculation of the mass balances are shown in Table 3-3, while the calculation results are shown in Table 3-4. For the estimation of nanoparticle quantities,

the particles are assumed to be equally sized with a diameter L. For the determination of the solid mass ratio between protamine and polyacrylic-acid, as well as the solid molar fractions, the electro-neutrality condition Eqn. 3-28 was used. The solid concentrations were calculated using Eqn. 3-30 to Eqn. 3-32.

3.3 Estimation of Characteristic Time Scales

To isolate the rate limiting step in the precipitation reactor, the characteristic time scale of each single process step, i.e., mixing, nucleation, growth and aggregation (see Figure 3-5), has been analyzed. The reactions involved in particle formation, i.e., proton transfer reactions, as well as the reaction between the polyacrylic acid and the protamine, have been assumed to occur instantaneously. This is justified, since typical reaction time scales of proton transfer reactions are in the order of 40 µs [33].

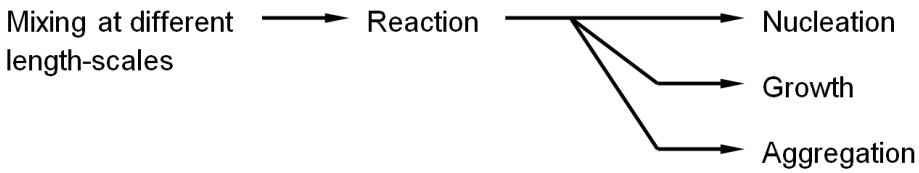


Figure 3-5: Schematic representation of process steps involved in precipitation.

3.3.1 Characteristic Time for Mixing

To estimate a characteristic time for mixing a reactor geometry has to be chosen. The experimental reactor will be designed as a confined impinging jet reactor (CIJR). Mixing in CIJRs has been already studied by Johnson and Prud'homme [34].

The characteristic time τ_m for the total mixing process (mixing at all length scales) found by Johnson and Prud'homme [34] is valid in the range of 150<Re<3,000 (the Reynolds number is defined by the inlet diameter d, the mean inlet velocity u and the kinematic viscosity ν) and for a Schmidt number of 1,000:

$$\tau_m = K_{CIJ} \cdot \frac{\nu^{1/2}\Delta^{3/2}d^{1/2}}{u^{3/2}} \cdot \frac{1}{2\left(\frac{\rho_1}{\rho_3}\right)^{1/2}\left(1+\frac{m_1}{m_2}\right)^{1/2}} \qquad \textbf{(3-34)}$$

Although the Schmidt number in the polyacrylic-acid/protamine system is in the order of 6,000, this correlation was used in lack of a better suited one. The constant K_{CIJ} is equal to 1,470 for the used reactor [34]. With equal feeding mass flows $m_1=m_2$, equal densities of precursors and product $\rho_1=\rho_3$, as well as the geometric ratio $\Delta=4.76$, the relation simplifies to:

$$\tau_m = 5400 \cdot \sqrt{\frac{\nu \cdot d}{u^3}} \qquad \textbf{(3-35)}$$

For an inlet velocity of 0.3 – 6 m/s (corresponding to 150<Re<3,000) and an inlet diameter of 0.5 mm the characteristic mixing time is in the range of 0.008 – 0.73 s.

3.3.2 Characteristic Time for Nucleation

A characteristic time for nucleation was calculated according to Baldyga et al. [7]:

$$\tau_{nuc} = \frac{N}{B_{\text{hom}}} \qquad \textbf{(3-36)}$$

Here N is the number density of the particles and B_{hom} is the homogeneous nucleation rate. The estimation of the nucleation time has to be interpreted carefully, because the high sensitivity of the nucleation rate to its parameters may cause errors of more than one order of magnitude. As shown in Table 3-4, the expected final particle number density N is in the order of $1.56 \cdot 10^{17}$ 1/m³. With the initial conditions and parameters given in Table 3-3 and Table 3-1, as well as an interfacial energy constant K of 0.414, a nucleation rate B_{hom} of $5.8 \cdot 10^{18}$ 1/m³s is obtained. This results in a characteristic nucleation time of 0.03 s. This value represents a lower bound, because the number of nuclei has to be larger than the final particle number used in this calculation.

3.3.3 Characteristic Time for Growth

A characteristic time for growth according to Baldyga et al. [7] is obtained as the ratio of the dissolved amount of species to the mass transfer by growth (where M_S is the mean molar mass of the solid):

$$\tau_{growth} = \frac{c \cdot M_S}{\rho_S \cdot G \cdot m_2} \qquad \textbf{(3-37)}$$

Substituting the growth rate G given by Eqn. 2-18, and a relation for the total particle surface m_2 (thus, particles of the same size are assumed yielding $m_2=L^2 \cdot \pi \cdot N$), the following relation for the growth timescale is obtained:

$$\tau_{growth} = \frac{c \cdot M_S \cdot c_S}{2 \cdot \rho_S \cdot Sh \cdot D \cdot (c - c^*) \cdot L \cdot \pi \cdot N} \cong \frac{1}{2 \cdot Sh \cdot D \cdot L \cdot \pi \cdot N} \qquad \textbf{(3-38)}$$

Note, $c_S=\rho_S/M_S$, and c/(c-c*) cancels out because of the low solubility of the product molecule (i.e., c*<<c). The growth timescale is proportional to the inverse of the particle size, meaning, that the growth of large particles leads to a higher consumption of dissolved molecules than the growth of small particles. This is caused by the higher absolute surface of larger particles.

Using a characteristic particle size of the growing particles L of 50 nm, a particle Sherwood number Sh of 2, a mean value for the diffusion coefficients of polyacrylic-acid and protamine ($D = 1.57 \cdot 10^{-10}$ m²/s) and the same particle number density as for the nucleation time scale ($N = 1.56 \cdot 10^{17}$ 1/m³), a growth timescale of 0.06 s is obtained.

3.3.4 Characteristic Time for Aggregation

Here we consider a characteristic time of particle collisions as a time scale for aggregation. This is justified by the assumption that the aggregation efficiency, which describes the amount of “successful” collisions, is in the order of unit for significant aggregation. A time scale for collisions according to Gavi et al. [5] is:

$$\tau_{coll} = \frac{1}{\beta_{coll} \cdot N} \qquad \textbf{(3-39)}$$

The collision kernel (Eqn. 2-21) depends on the size ratio of two colliding particles. Therefore the collision time scale was calculated for three different size ratios of 1:1, 1:10 and 1:100. For the particle number density N, the final value of $1.56 \cdot 10^{17}$ 1/m³ was used

again. During the aggregation the particle number density has to be larger, because it is reduced by aggregation. Therefore the collision time scale is an upper bound, i.e., it will be smaller in the initial stages of the precipitation process. The results are shown in Table 3-5.

L_1/L_2	1	0.1	0.01
τ_{coll} [s]	0.59	0.20	0.02

Table 3-5: Characteristic collision times

The value of 0.59 s for the 1:1 size ratio is not relevant, because aggregation of equally sized collision partners only takes place at small particle sizes below 20 nm. For larger ratios the aggregation efficiency is close to zero (shown in Chapter 4). The number of these small particles (below 20 nm) is expected to be orders of magnitude higher than the number of the large final particles, used for the calculation of the time scales. According to Eqn. 3-39 the collision time scale is inversely proportional to the particle number. Thus, the exact time between 1:1 collisions can be expected to be significantly lower than 0.59 s. The timescale of 0.20 s for the 1:10 size ratio was taken as an upper bound for the collision time. Size ratios smaller than 1:10 lead to smaller collision times, which have, however, a low probability.

3.3.5 Comparison of the Characteristic Time Scales

The characteristic times of the single process steps, calculated in the Sections 3.3.1 to 3.3.4, are summarized in Table 3-6.

Mixing	0.008 – 0.73 s
Proton transfer reactions	approx. 40 μs
Nucleation	> 0.03 s
Growth	0.06 s
Collisions	< 0.20 s

Table 3-6: Characteristic time scales of the investigated precipitation process.

The mixing time strongly depends on the Re number. Even for the largest considered Reynolds number of 3,000, the mixing time is significant compared to the nucleation time. Thus, in the considered range of Reynolds numbers the process is controlled by mixing and a well-mixed calculation will not give reliable results. Mixing will influence the product and is affected by the physical dimensions of the reactor (see Eqn. 3-35). Thus, the product size distribution will depend on the length scale of the reactor, and scale-up is a relevant issue for this process.

The particles produced in this process should be in the range of 100 nm. To get a small particle size, the number of particles has to be high, i.e., a high nucleation rate is required. In order to produce a supersaturation high enough for the required nucleation rate, mixing has to be fast compared to nucleation. For the calculation of the mixing time, a typical microreactor was considered. As obvious in Eqn. 3-35, the mixing time can be kept low by a small length scale. Thus, fast mixing can be done in a microreactor. The characteristic times for nucleation, growth and aggregation are in the same order of magnitude, i.e., they are expected to run in parallel, and are mainly influenced by the mixing. The already shown dependency of the aggregation rate on the size ratio of the collision partners (Table 3-5) highlights, that aggregation cannot significantly accelerate the growth of particles near the mean particle size (slow aggregation for the 1:1 ratio). However, aggregation reduces the amount of fine particles by attachment to larger ones (fast aggregation for different sized particles).

4 Computational Models

4.1 Basic Model Assumptions

For the development of the mathematical model the following basic model assumptions have been made:

- The precursors are in aqueous solutions, i.e., polyacrylic-acid and protamine are assumed to be completely soluble in water. Polyacrylic-acid and protamine is a weak acid and base, respectively, and change the pH-value of the aqueous solution due to dissociation.
- The particles are assumed to be amorphous. In contrast to crystalline particles, the integration step of growth is assumed to be instantaneous. Consequently the growth is assumed to be limited by the transport of molecules to the surface of the particle.
- The product particles are a solid mixture of polyacrylic-acid and protamine. The composition of the solid particles is non-stoichiometric, i.e., it depends on the charge numbers of the molecules involved in their formation. However, the charge numbers depend on the pH-value of the solution, which is determined by the ratio of polyacrylic-acid and protamine. Hence, it is not possible to define a net chemical reaction equation of the particle formation process, and also the kinetics are unclear. However, they can be expected to be fast (i.e., in the order of 10^{-5} s). Consequently, all reactions have been modeled to occur instantaneously.
- The remaining process steps, which need to be modeled, are mixing, nucleation, growth and aggregation.
- Nucleation is assumed to be homogeneous. The size of the nuclei is assumed to be the critical nuclei radius.
- Collisions of particles are assumed to be caused by Brownian motion, which is the dominant mechanism for nanoparticles. Furthermore only binary collisions are considered.
- The protamine molecules are assumed to determine the surface charge of the particles. It was shown in preliminary experiments, that the excess of protamine prevents the nanoparticle suspension from further aggregation by influencing the zeta-potential.

- The dissociation number ν_D, which is a parameter required for the nucleation rate (Eqn. 2-14), is defined as number of ions in one crystal unit (e.g. for $BaSO_4$ ν_D=2). For the substances considered in this work, the composition of the solid material depends on the pH-value, hence the definition of the dissociation number cannot be applied. Thus, the dissociation number was set to the number of components in the solid material, which is 2.
- In contrast to the complete solubility of the pure precursors, the mixture of them is hardly soluble. That is a precondition to obtain solid particles. To apply the classical nucleation theory, the definition of a supersaturation is required, which is defined as the ratio of the actual concentration to the equilibrium concentration (see. Eqn. 2-4). As explained above, it was not possible to define a chemical reaction with a hardly soluble product, as in the case of inorganic salts. Hence the solid mixture of the precursors was assumed to be hardly soluble itself. The supersaturation had to be defined by the concentrations of the dissolved components polyacrylic-acid and protamine. Although there is no physical meaning in this case, this was done analogous to the solubility product of inorganic salts:

$$S = \frac{c}{c^*} = \sqrt{\frac{c_1 \cdot c_2}{K_S}} \qquad \textbf{(4-1)}$$

The square root of the solubility product is the equilibrium concentration:

$$c^* = \sqrt{K_S} \qquad \textbf{(4-2)}$$

4.2 Nucleation Model

Based on the model assumptions explained above, the homogeneous nucleation theory is applied to the polyacrylic-acid/protamine system. The parameters of the nucleation rate (Eqn. 2-14), the required interfacial energy (Eqn. 2-13), and the critical nuclei radius (Eqn. 2-17) are known (see Chapter 3). The only unknowns are the interfacial energy constant K and the equilibrium concentration c*. To find a range for their values and to investigate

their influence on the nucleation rate, a parameter study was performed with MATLAB R2008a (for the code see Appendix B/II).

It is interesting to notice, that although the temperature appears in the nucleation rate (Eqn. 2-14), it cancels out after the substitution of the interfacial energy σ by Eqn. 2-13. That does not mean there is no temperature dependency of the nucleation rate. It is only hidden in the equilibrium concentration c* and the diffusion coefficient D, which are usually temperature dependent.

4.2.1 Comparison of the Nucleation Rate with Literature Data

Molecular weight	233.40 g/mol
Dissociation number	2
Hydrodynamic radius	0.44482 nm
Diffusion coefficient in H_2O at 22°C	$4.86 \cdot 10^{-10}$ m²/s
Solid density	4500 kg/m³
Solubility product	$1.01 \cdot 10^{-10}$ mol²/l²

Table 4-1: Parameters for the precipitation of $BaSO_4$.

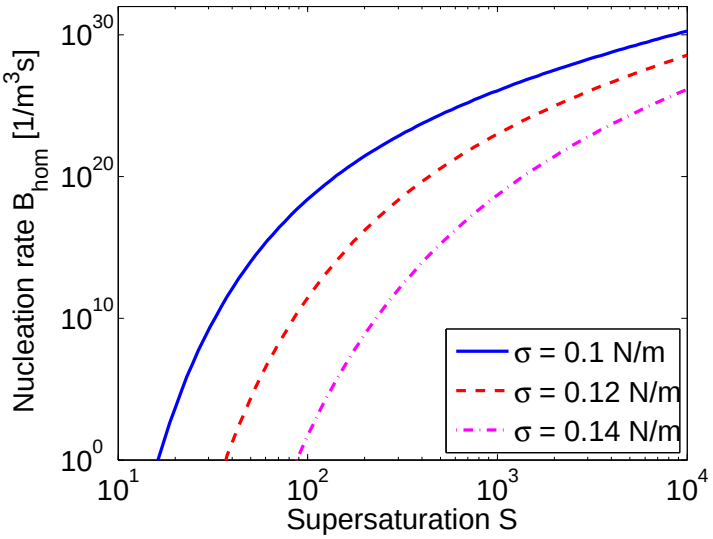


Figure 4-1: Dependency of the nucleation rate on the supersaturation for the precipitation of $BaSO_4$.

To verify the calculation of the nucleation rate, a comparison with the results of Schwarzer and Peukert [6], who investigated the precipitation of $BaSO_4$, was performed. The parameters for the precipitation of $BaSO_4$ are shown in Table 4-1. The used value for the

interfacial energy constant K was 0.414. The results shown in Figure 4-1 are identical to that reported in Schwarzer and Peukert [6].

4.2.2 Parameter Study of the Nucleation Rate

First of all, the influence of the unknown parameters K and c* on the homogeneous nucleation rate B_{hom} was investigated. According to Mersmann et al. [14], the interfacial energy constant K should be in the range of 0.310 to 0.414, hence the variation was done within this range. The equilibrium concentration c* must be lower than the initial concentration, therefore its variation was considered in the interval 10^{-12} to 10^{-6} mol/l.

As shown in Figure 4-2, the sensitivity of the nucleation rate to the variation of the equilibrium concentration is very high at equilibrium concentrations near the actual concentration in the system (the concentration after mixing is $3.6 \cdot 10^{-5}$ mol/l, see Eqn. 4-1). The lower the equilibrium concentration, the weaker is its influence on the nucleation rate. That means for the determination of c*, c* below 10^{-10} mol/l does not significantly change the nucleation rate, values of c* much larger than 10^{-10} mol/l influence the nucleation rate B_{hom} considerably.

It is also obvious, that at higher interfacial energy constants K the nucleation rate is lower, because higher values of the interfacial energy σ (see Eqn. 2-13) lead to a larger cluster size required for a stable nucleus. The formation of such larger clusters has a lower probability, and consequently the nucleation rate is lower.

From preliminary experiments the entire process time is known to be in the order of seconds or smaller. Together with the expected particle number density of $1.56 \cdot 10^{17}$ $1/m^3$ (see Chapter 3) a minimum value for the nucleation rate can be determined. A value in the order of 10^{17} $1/m^3s$ is required to obtain a desired number of particles with a size of around 100 nm. In order to get sufficiently high values for the nucleation rate, the equilibrium concentration c* must be below 10^{-10} mol/l for K in the range of $0.31<K<0.414$ (see Figure 4-2). Hence the value of 10^{-10} mol/l was chosen for the equilibrium concentration in lack of more precise experimental data.

In Figure 4-3 the sensitivity of the nucleation rate with respect to the parameter K for different values of the supersaturation is shown (using the value of 10^{-10} mol/l for c*). The variation of K between 0.31 and 0.414 causes a change of the nucleation rate by more than 3 orders of magnitude. This high sensitivity can be used to adjust the simulation results by

fitting the interfacial energy constant K to yield the experimentally determined mean particle size.

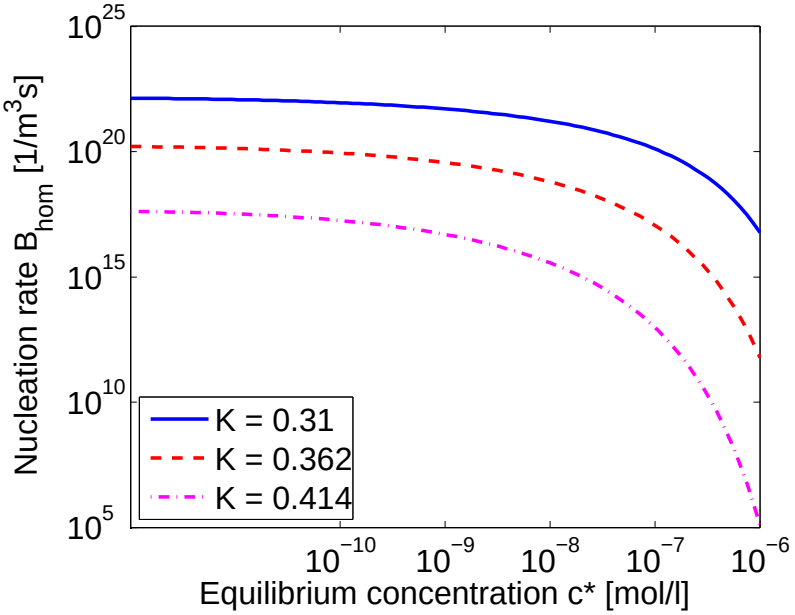


Figure 4-2: Dependency of the nucleation rate on the equilibrium concentration.

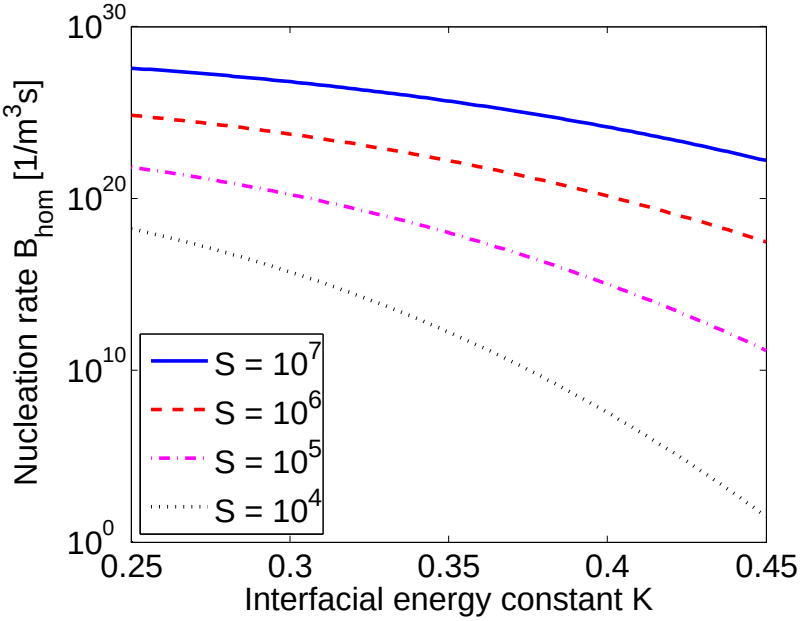


Figure 4-3: Sensitivity of the nucleation rate to variations of the interfacial energy constant for different values of the supersaturation.

Moreover, it is interesting to know the size of the critical nuclei. In Figure 4-4 the dependency of the critical nuclei size on the supersaturation is shown. The comparison of

these results to the hydrodynamic radius r_h of the molecules (approx. 1.4 nm) shows, that for supersaturations between 10^3 and 10^5 (depending on the value of K) the critical nuclei radius is smaller than r_h. Thus, at sufficiently high supersaturations the large molecules of polyacrylic-acid and protamine are nuclei themselves and precipitate spontaneously. The interfacial energy constant K influences the interfacial energy proportionally (Eqn. 2-13), therefore at higher values of K the nuclei get a higher surface energy and for stability a higher nuclei radius is required.

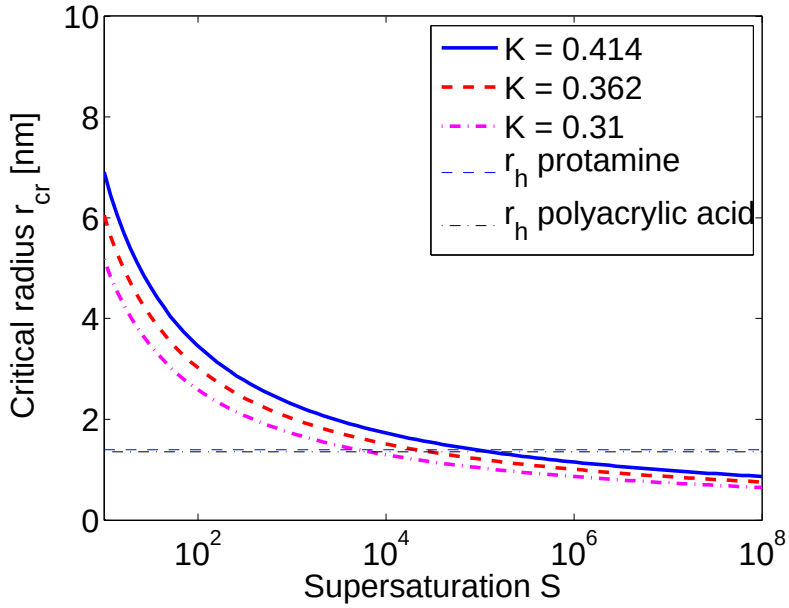


Figure 4-4: Critical nuclei radius as function of the supersaturation for different values of the interfacial energy constant.

In the case of polyacrylic-acid and protamine, the molar mass and hence the mean volume of a single molecule ($5.5 \cdot 10^{-27}$ m³) is much larger than in the case of barium sulfate ($8.6 \cdot 10^{-29}$ m³) or similar inorganics. Therefore, it is interesting, to investigate the influence of the molecule size on the nucleation rate. For this calculation the molecular volume and the solid concentration (which is directly related to the molecular volume by Eqn. 3-33) have been varied, while the other parameters have kept constant. For K a value of 0.414 was used.

As shown in Figure 4-5, the nucleation rate changes by more than 3 orders of magnitude for a ten-fold increase of the molecular volume. That means, the nucleation is considerably faster for larger molecules. The probability for the generation of stable clusters is higher

for larger molecules, caused by the lower number of molecules required for one stable cluster.

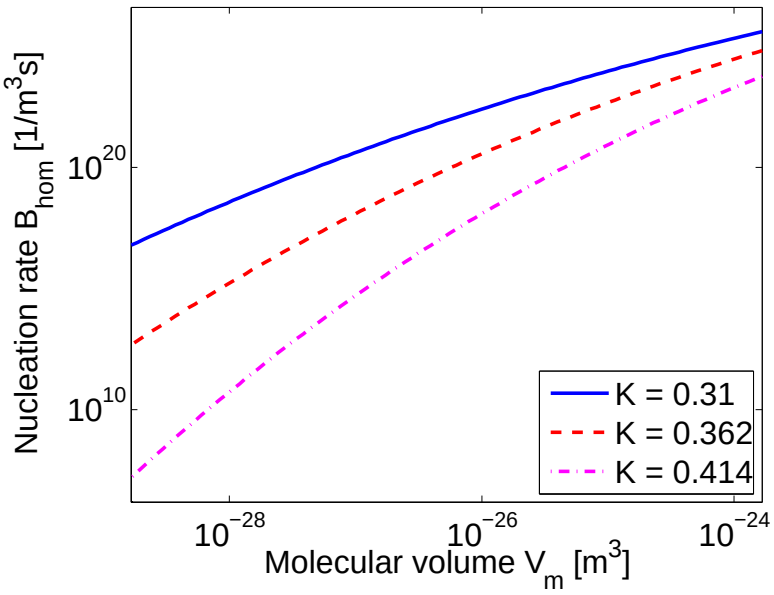


Figure 4-5: Dependency of the nucleation rate on the molecular volume.

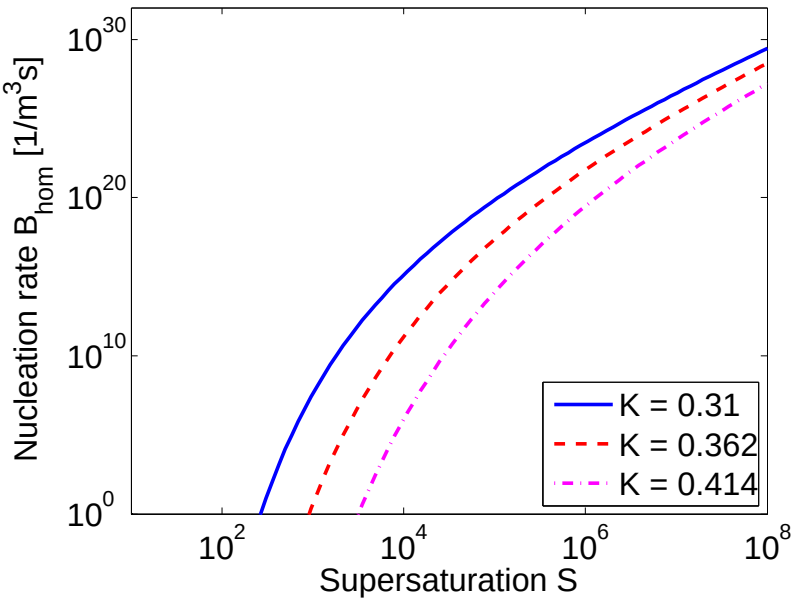


Figure 4-6: Dependency of the nucleation rate on the supersaturation.

Finally, the dependency of the nucleation rate on the supersaturation, which controls the evolution of the nucleation rate during the process, is shown in Figure 4-6. Clearly, for a nucleation rate of 10^{17} 1/m³s a minimum initial supersaturation of about 10^4 to 10^5 (depending on K) is required.

For small supersaturations the nucleation rate decreases strongly. Thus, during the mixing of the precursors the supersaturation has to overcome a threshold in order to yield a significant nucleation rate. At supersaturations below this threshold, there is no significant particle generation, although a certain supersaturation is present in the system. After overcoming the threshold, a significant amount of nuclei is generated, enabling growth and aggregation. By the consumption of dissolved precursor molecules, the supersaturation decreases after reaching a maximum value. The critical threshold for (significant) nucleation is then passed, nucleation is essentially stopped and the remaining supersaturation will be decreased via mass transport to the surface of the particles (i.e., particle growth). Finally the equilibrium is reached, where the supersaturation is unity.

4.3 Aggregation Model

The kinetics of aggregation are described by an aggregation kernel (Eqn. 2-20). It consists of the collision kernel (Eqn. 2-21) and the aggregation efficiency (Eqn. 2-22), including sub-models for particle interaction forces (Eqn. 2-23 – Eqn. 2-31). Most of the required parameters are known, only the Hamaker constant A, describing the Van der Waals forces, and the concentration of potential determining ions (PDI) at the point of zero charge (pzc) c_{PDI}^{pzc}, determining the electrostatic forces, are unknown. To investigate the influence of the unknown parameters and to get a deeper understanding of the model, a parameter study was performed.

4.3.1 Collision Kernel

The collision kernel accounting for Brownian motion β_{coll} (Eqn. 2-21) can be written as a product of a size-independent, constant prefactor and a dimensionless part β_{coll}^{*}. The latter depends only on the ratio of the two particle sizes λ:

$$\beta_{coll}(L_1, L_2) = \frac{2kT}{3\eta} \cdot \beta_{coll}^{*} \qquad \textbf{(4-3)}$$

$$\beta_{coll}^{*}(\lambda) = 2 + \lambda + \frac{1}{\lambda} \qquad \textbf{(4-4)}$$

with $\lambda = \frac{L_2}{L_1}$ **(4-5)**

A plot of the dimensionless kernel, i.e., β_{coll}^*, is shown in Figure 4-7. Clearly, the frequency of particle collisions increases with increasing particle size ratio λ. This is due to the different mobility (i.e., diffusion coefficients) of differently sized particles. A surface plot of the collision kernel as function of the size of the colliding particles is shown in Figure 4-8.

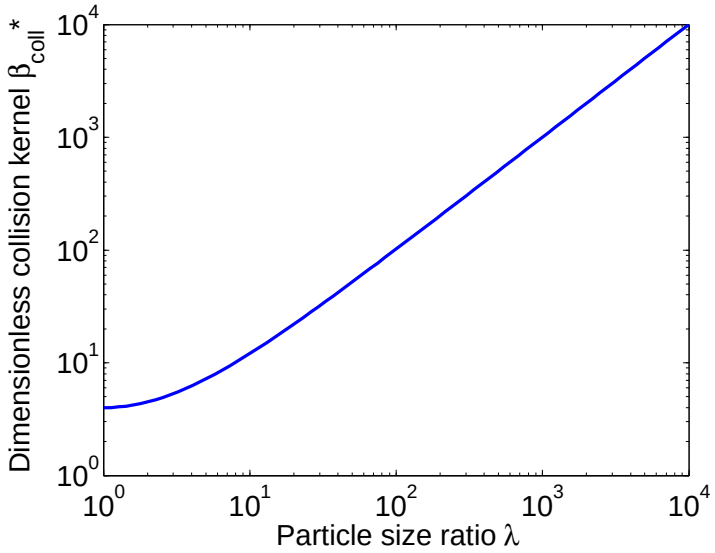


Figure 4-7: Dimensionless collision kernel as a function of the particle size ratio.

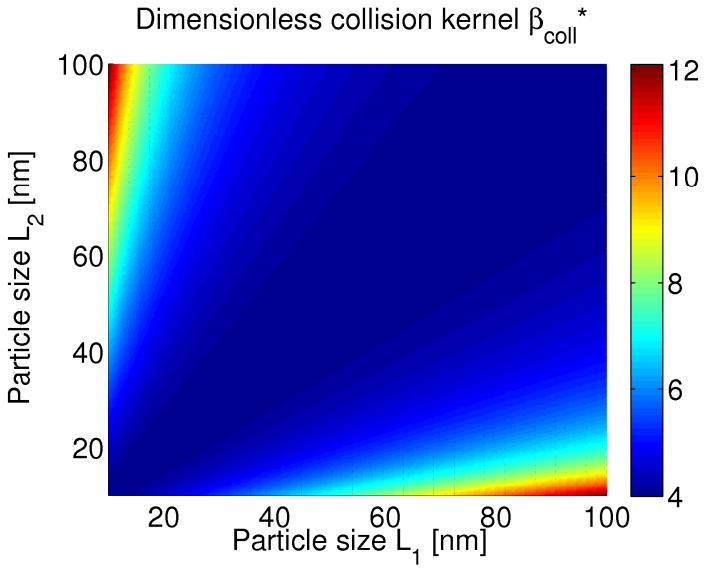


Figure 4-8: Dimensionless collision kernel as a function of two particle sizes.

4.3.2 Hamaker Potential

The attractive Van der Waals interaction forces between two spherical particles with the radii r_1 and r_2 are described by the Hamaker theory. The Hamaker potential ϕ_{VdW}, which is a function of the surface-to-surface distance "a" between the particles, can be transformed into a dimensionless formulation. Thus, the dimensionless surface-to-surface distance ξ is introduced:

$$\phi_{VdW}(\xi) = -\frac{A}{6}\left(\frac{2\lambda}{\xi^2+2\xi(1+\lambda)} + \frac{2\lambda}{\xi^2+2\xi(1+\lambda)+4\lambda} + \ln\left(\frac{\xi^2+2\xi(1+\lambda)}{\xi^2+2\xi(1+\lambda)+4\lambda}\right)\right) \quad \textbf{(4-6)}$$

$$\text{with } \xi = \frac{a}{r_1} \quad \textbf{(4-7)}$$

$$\text{with } \lambda = \frac{r_2}{r_1} \quad \textbf{(4-8)}$$

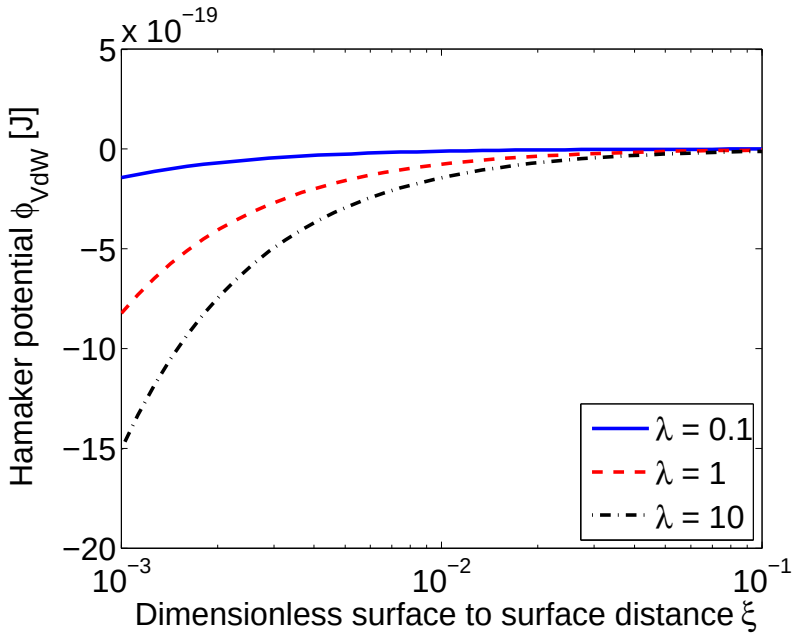


Figure 4-9: Hamaker potential for different particle size ratios λ.

Typical values for the Hamaker constant A for polymers dissolved in water are in the order of 10^{-20} J. According to Visser [35] e.g., the Hamaker constant for polytetrafluorethylene in

water is $0.64 \cdot 10^{-20}$ J, for polyethylene in water $6.4 \cdot 10^{-20}$ J and for polyvinyl-acetate in water $8.7 \cdot 10^{-20}$ J. The value for the polyacrylic-acid/protamine system is unknown, hence a value of $1 \cdot 10^{-20}$ J was chosen.

In Figure 4-9 a plot of the Hamaker potential for three different size ratios λ is shown. The Hamaker potential is symmetrical with respective to the particle radii r_1 and r_2 of course, the different curves for λ=0.1 and λ=10 are only caused by the reference for the dimensionless surface-to-surface distance ξ, which changes when r_1 and r_2 are interchanged. For a constant particle size r_1, an increase in r_2 (and consequently λ) results in a decrease in the Hamaker potential, i.e., Van der Waals attraction forces increase (the force is the gradient of the potential).

4.3.3 Electrostatic Potential

The electrostatic potential, calculated by the Gouy-Chapman model, describes the repulsive electrostatic forces due to particle surface potentials. The relative permittivity ε_r of water at 22°C has been assumed to be equal to 80 [33]. The surface potential of the particles is influenced by the concentration of the PDI. As assumed in Section 4.1, protamine determines the surface potential, hence it is the PDI. The only unknown parameter in this model is the concentration c_{PDI}^{pzc} of the PDI at the pzc.

The initial concentration of PDI is ca. $7 \cdot 10^{-5}$ mol/l, during the process the concentration of PDI decreases to the final value of ca. $2 \cdot 10^{-5}$ mol/l (respectively 0.087 g/l as shown in Chapter 3). To enable aggregation at the beginning of the process and to prevent aggregation of the final particles, the concentration of PDI at pzc is chosen to be $1 \cdot 10^{-4}$ mol/l, which is near the initial concentration of PDI. To get a precise value for c_{PDI}^{pzc}, experimental data of the zeta potential of the particles as function of the excess-concentration of protamine are required.

In Figure 4-10 the electrostatic potential for different combinations of particle sizes is shown. The concentrations in the solution are kept constant at the values of the finished process (i.e., the conversion X=1), where polyacrylic-acid is completely consumed and only the protamine excess is remaining. The larger the particles, the larger is the electrostatic potential and also the electrostatic repulsion forces (which are the gradient of the potential). If at least on particle involved in a collision is small (e.g., L = 10 nm), the

electrostatic potential is reduced. That means, large particles repulse each other much stronger than small particles, or a combination of a large and a small particle.

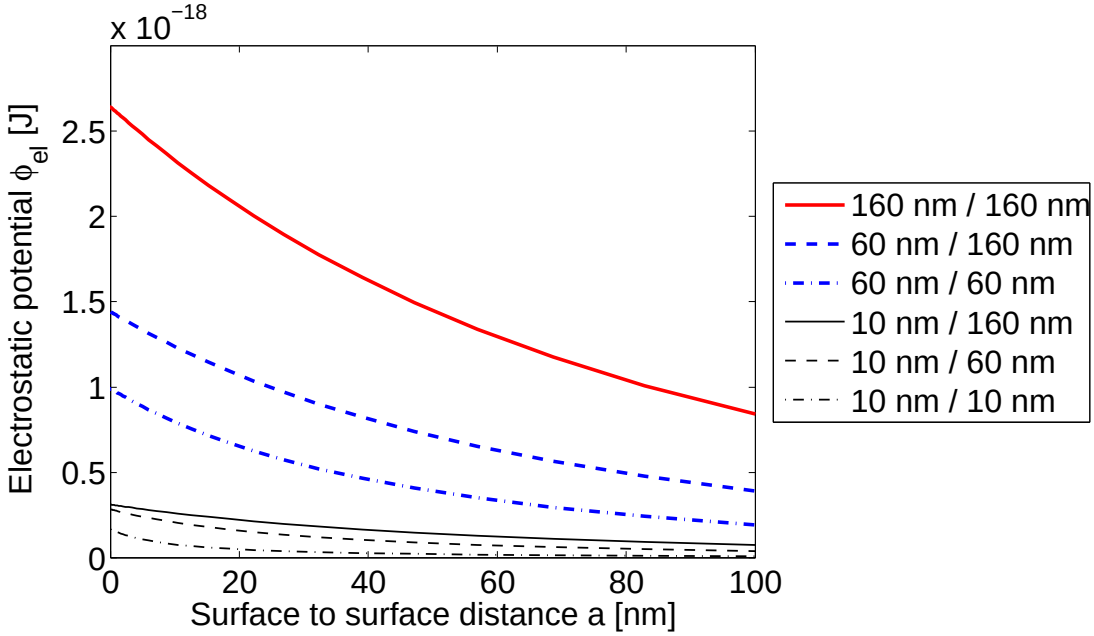


Figure 4-10: Electrostatic potential for different particle sizes (conversion X = 1).

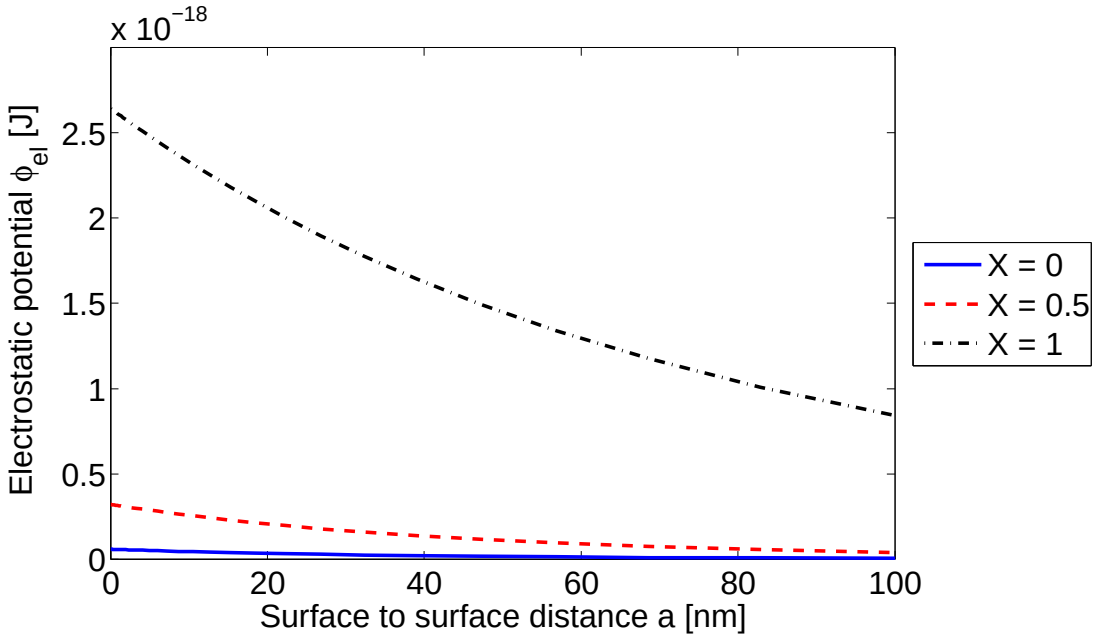


Figure 4-11: Electrostatic potential for different conversions (particles 160nm/160nm).

The influence of the concentrations in the solution is illustrated in Figure 4-11, where the electrostatic potential is shown for three different conversions. The value of c_{PDI}^{pzc} ($1 \cdot 10^{-4}$ mol/l) was chosen in order to prevent aggregation at the end of the process by electrostatic

repulsion. Hence, the electrostatic potential for the conversion X=1 is the largest, while for X=0 it is vanishing, meaning that at the beginning of the process no significant repulsion is existing and aggregation can take place. For the conversion X=0.5 the electrostatic potential is near X=0. Thus, the change of the repulsion forces during the first half of the process is relatively low.

4.3.4 Total Interaction Potential

The total interaction potential is the superposition of the Hamaker and electrostatic potential. Figure 4-12 is basically the same as Figure 4-10, but includes the Hamaker potential. Only in the case of a surface-to-surface distance below 10 nm a significant contribution of the Van der Waals potential is apparent. This nicely illustrates the difference in the range of Van der Waals and electrostatic forces. Van der Waals forces are short-range forces, influencing the interaction potential only at small distances (typically in the range of nm and below), while the long-range electrostatic forces interact over more than 100 nm. The influence of different particle sizes is the same than discussed in Section 4.3.3.

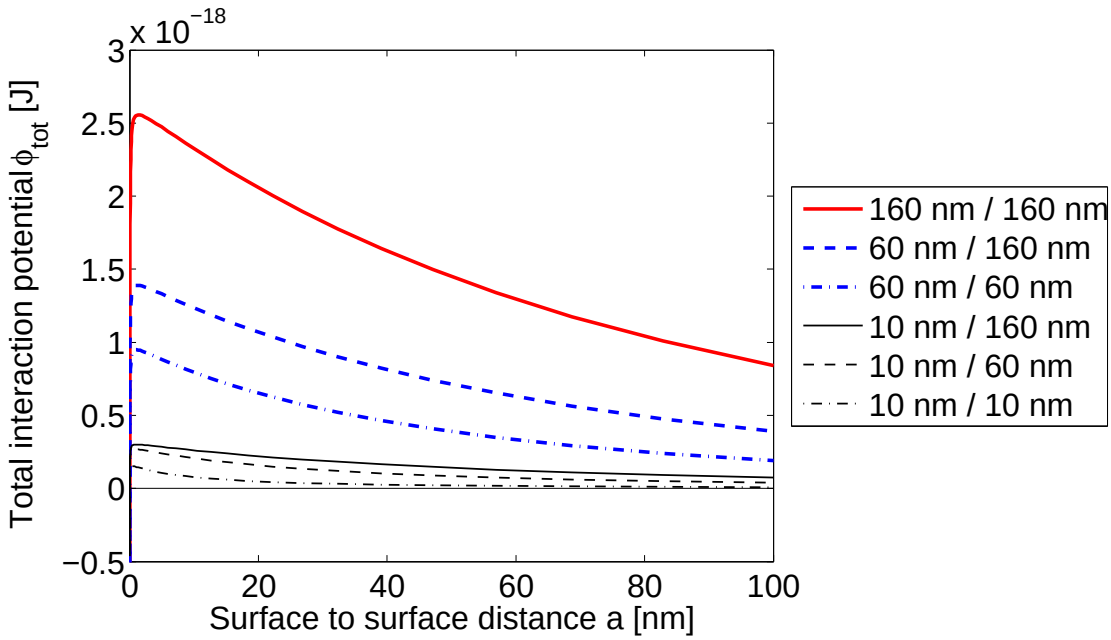


Figure 4-12: Total interaction potential for different particle sizes (conversion X=1).

4.3.5 Hydrodynamic Correction

The hydrodynamic correction (Eqn. 2-24) accounts for the viscous resistance of the fluid, which decreases the aggregation efficiency. The parameter r* is a mean value of the collision partner sizes r_1 and r_2. As shown in Figure 4-13, for specific surface-to-surface distances over 10, the hydrodynamic correction is nearly one, which means that its influence is negligible. Only for specific surface-to-surface distances below 10 it will impact the aggregation significantly.

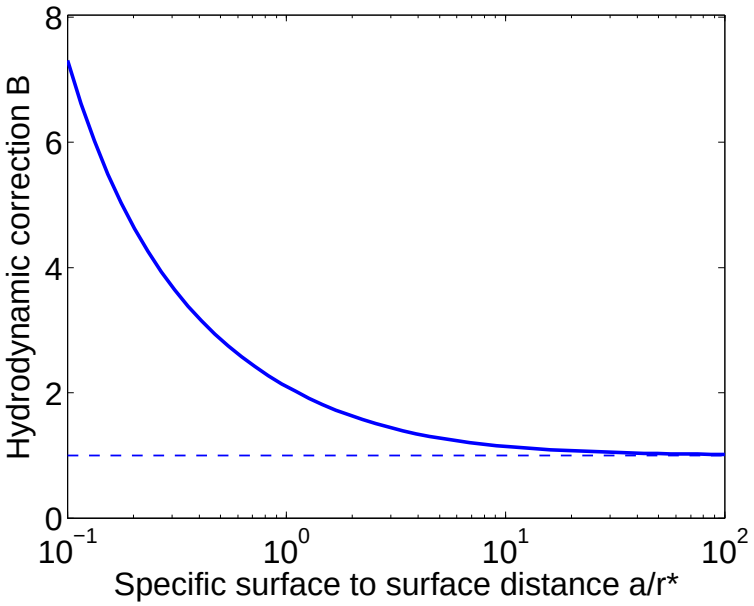


Figure 4-13: Hydrodynamic correction.

4.3.6 Aggregation Efficiency and Aggregation Kernel

The aggregation efficiency takes into account the effects of the interaction potentials and the hydrodynamic correction. These effects are combined by an integral over the particle separation, which is called stability ratio (Eqn. 2-23), and is the inverse of the aggregation efficiency. The integral was solved numerically by MATLAB R2008a (for the code see the Appendix B/III).

Most of the parameters influencing the aggregation efficiency are known, only the Hamaker constant A and the concentration of PDI at point of zero charge have been estimated. In the following, the influence of their variation on the aggregation efficiency was studied.

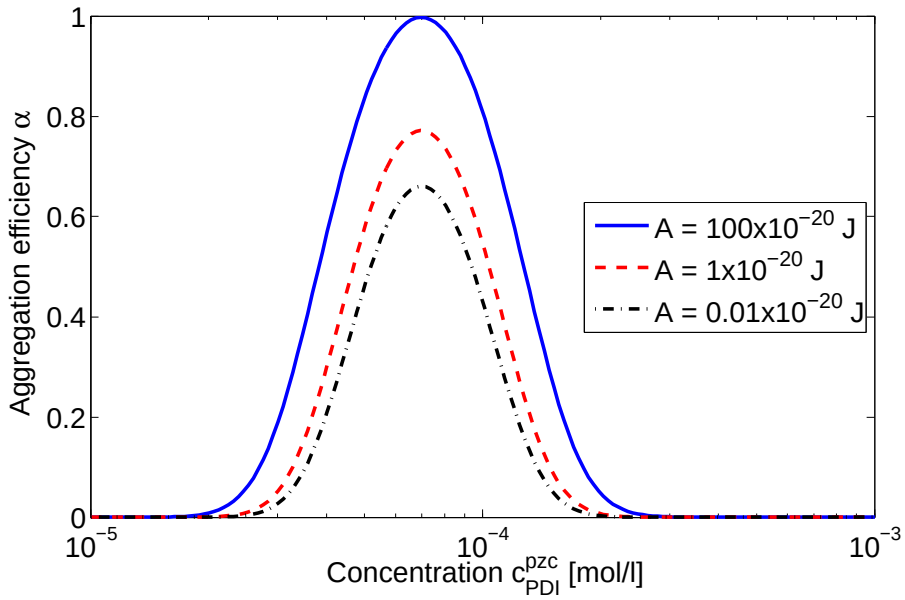


Figure 4-14: Aggregation efficiency as function of the concentration of PDI at pzc for different Hamaker constants A (c_{PDI} is $6.98 \cdot 10^{-5}$ mol/l, particle sizes 5nm/100nm).

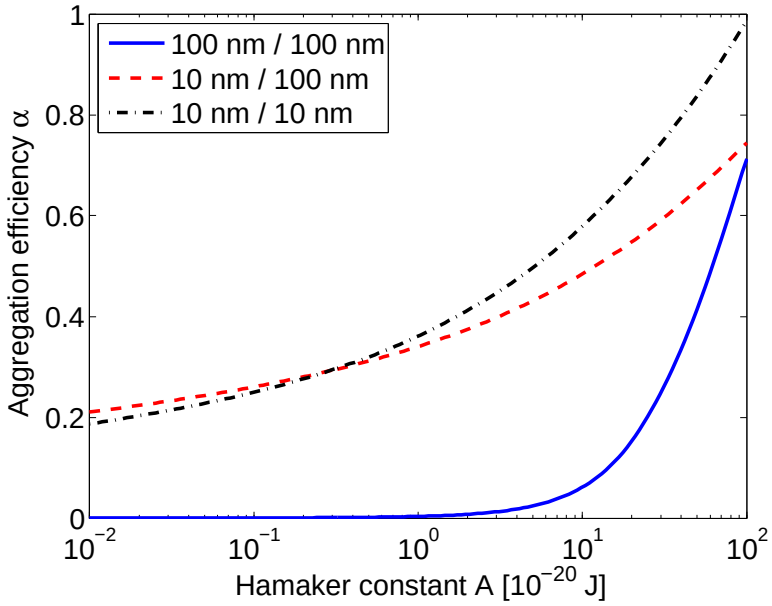


Figure 4-15: Aggregation efficiency as function of the Hamaker constant.

In Figure 4-14 the influence of the parameter c_{PDI}^{pzc} on the aggregation efficiency is shown for different values of the Hamaker constant A and for a particle collision scenario involving a 5 nm and 100 nm particle. For the concentrations the initial mixed concentrations of the reactants (conversion X=0) was used (see Chapter 3).

As shown in Figure 4-14 the aggregation efficiency reaches a maximum when the parameter c_{PDI}^{pzc} is equal to the concentration of the PDI c_{PDI} (i.e., the concentration of the

protamine, $6.98 \cdot 10^{-5}$ mol/l in this case). Then the system is at the point of zero charge, meaning the particles are uncharged and the electrostatic repulsion is zero. This makes aggregation most likely. The more c_{PDI}^{pzc} differs from c_{PDI}, the smaller gets the aggregation efficiency, caused by the increasing electrostatic repulsion due to adsorption of ions on the surface. The variation of the Hamaker constant A in the range of $0.01\text{-}100 \cdot 10^{-20}$ J causes only moderate changes of the aggregation efficiency.

In Figure 4-15 the influence of the Hamaker constant A on the aggregation efficiency for different particle sizes is shown. The initial concentrations (i.e., a conversion of X=0 was assumed) were used again, while the parameter c_{PDI}^{pzc} was set to the value of $1 \cdot 10^{-4}$ mol/l, i.e., the system is near the pzc. The Hamaker constant A was varied between 0.01 and $100 \cdot 10^{-20}$ J again.

The aggregation efficiency (i.e., the probability for aggregation in the case of a collision) shown in Figure 4-15 increases with an increasing Hamaker constant, due to increasing Van der Waals forces. For a collision of large particles (e.g., 100nm/100nm) the aggregation efficiency is much lower than for a collision of particles including at least one small particle (10 nm). The reason is the electrostatic potential. As discussed in Section 4.3.3, it causes a much stronger repulsion during a collision of large particles, compared to a collision including at least one small particle. The repulsion during a l00nm/100nm collision is as significant, that for Hamaker constants below $1 \cdot 10^{-20}$ J no aggregation is possible.

In Figure 4-16 the aggregation efficiency α is shown as function of two particle sizes, where again the initial concentrations have been used (i.e., a conversion of X = 0 was assumed). The Hamaker constant A was set to $1 \cdot 10^{-20}$ J and for the concentration of PDI, c_{PDI}^{pzc}, the value of $1 \cdot 10^{-4}$ mol/l was used again. As already discussed, for pairs of large particles (over 80 nm) aggregation is very unlikely for the used parameters. If at least one particle of the collision partners is below 20 nm, the aggregation efficiency is over 0.3, meaning over 30% of the collisions lead to aggregation. Aggregation efficiencies over 0.7 are only obtained if one collision partner is in the range of some nanometers. That means, the smaller a particle, the higher is the probability to aggregate in the case of a collision, but between too large particles aggregation is unlikely.

Finally, a surface plot of the aggregation kernel as a function of two particle sizes is shown in Figure 4-17, which represents the rate of aggregation. The aggregation kernel differs from the aggregation efficiency, because it takes into account also the collision frequency.

Between particles over 20 nm there is no significant aggregation. Either their aggregation efficiency is zero (for particles over 80 nm), caused by too high electrostatic repulsion, or their collision kernel is too low (in case of particles with a similar size). Only if the size ratio of the collision partners is sufficiently high (over 10) a significant aggregation rate can be observed.

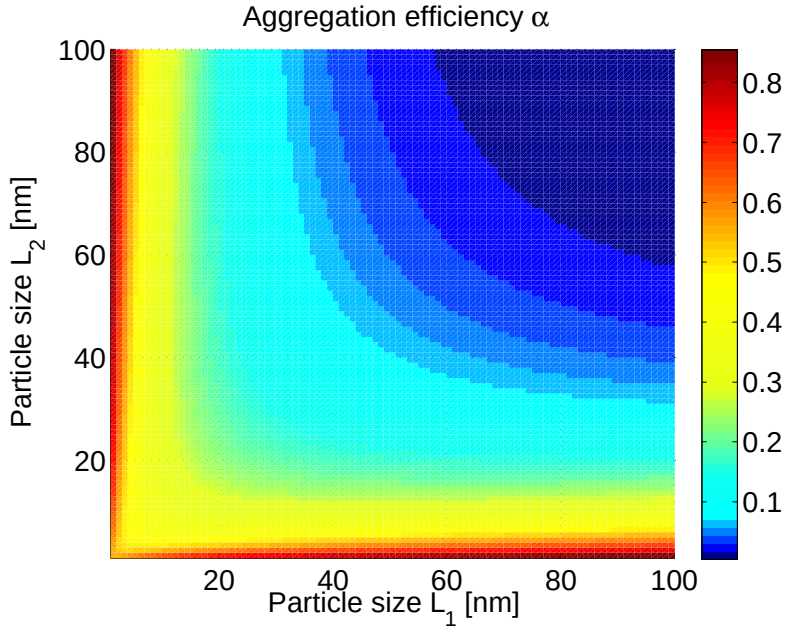


Figure 4-16: Aggregation efficiency as function of two particle sizes.

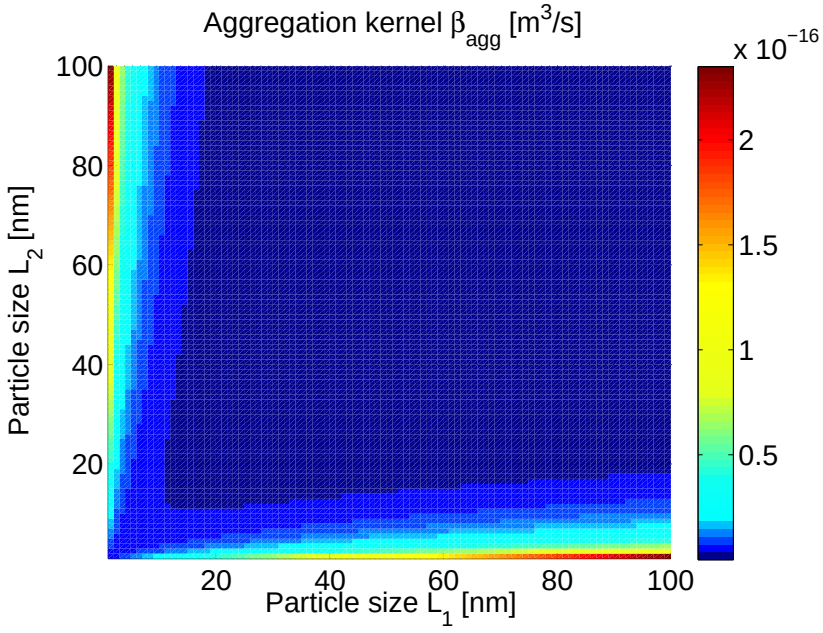


Figure 4-17: Aggregation kernel as function of two particle sizes.

5 Solution for a Well-Mixed System

The first step on the way to the simulation of the precipitation process is the solution of the PBE for the case of a well-mixed system. As solution method for the numerical solution of the PBE, i.e., the classes method was used (see Chapter 2).

5.1 Governing Equations

Basically, the precipitation process is described by the PBE (Eqn. 2-1). In the case of a well-mixed system the spatial derivatives are zero. This yields:

$$\frac{\partial n(L,t)}{\partial t} = J(L,S(t)) - \frac{\partial (G(L,S(t)) \cdot n(L,t))}{\partial L} + B_{agg}(L,t) - D_{agg}(L,t) \qquad \textbf{(5-1)}$$

Here, the nucleation source term J is defined by Eqn. 2-16, the growth rate G by Eqn. 2-18 and the aggregation rates B_{agg} and D_{agg} by Eqn. 2-2 and Eqn. 2-3. The supersaturation S is defined by Eqn. 4-1. To close the system, the balance equations for the species concentrations c_1 and c_2 in the liquid phase are required. The concentration decrease of species i due to nucleation is calculated as the amount of species i in one nucleus, $\nu_{i,Nucleus}$, multiplied by the nucleation rate B_{hom} (Eqn. 2-14), which is the number of generated nuclei per volume and time:

$$\left(\frac{dc_i}{dt}\right)_{Nucleation} = -B_{\hom}(S(t)) \cdot \nu_{i,Nucleus} \qquad \textbf{(5-2)}$$

The amount of species i in one nucleus is calculated by the concentration of species i in the solid phase (Eqn. 3-30, 3-31) and the volume of the nucleus, determined by its radius r_{cr}:

$$\nu_{i,Nucleus} = c_{Si} \cdot \frac{4}{3} \cdot r_{cr}^{\,3} \cdot \pi \qquad \textbf{(5-3)}$$

The molar flow rate of species i consumed by the growth of a single, spherical particle with the diameter L, is due to the definition of the Growth rate G = ∂L/∂t as time derivative of the diameter L:

$$\dot{\nu}_{i,Growth}(L) = \frac{G(L,S(t))}{2} \cdot L^2 \cdot \pi \cdot c_{Si} \qquad \textbf{(5-4)}$$

The concentration decrease of species i due to growth follows by integration over all particle classes:

$$\left(\frac{dc_i}{dt}\right)_{Growth} = -\int_{L=0}^{\infty} \dot{\nu}_{i,Growth}(L) \cdot n \cdot dL \qquad \textbf{(5-5)}$$

The total concentration decrease dc_i/dt of species i due to nucleation and growth is:

$$\frac{dc_i}{dt} = \left(\frac{dc_i}{dt}\right)_{Nucleation} + \left(\frac{dc_i}{dt}\right)_{Growth} \qquad \textbf{(5-6)}$$

Substitution of Eqn. 5-3 in 5-2, Eqn. 5-4 in 5-5, and finally in Eqn. 5-6 gives the changes of the concentrations c_1 and c_2 due to nucleation and growth:

$$\frac{dc_1}{dt} = -B_{\text{hom}}(S(t)) \cdot \frac{4}{3} \cdot r_{cr}{}^3 \cdot \pi \cdot c_{S1} - \frac{\pi \cdot c_{S1}}{2} \cdot \int_{L=0}^{\infty} G(L,S(t)) \cdot L^2 \cdot n \cdot dL \qquad \textbf{(5-7)}$$

$$\frac{dc_2}{dt} = -B_{\text{hom}}(S(t)) \cdot \frac{4}{3} \cdot r_{cr}{}^3 \cdot \pi \cdot c_{S2} - \frac{\pi \cdot c_{S2}}{2} \cdot \int_{L=0}^{\infty} G(L,S(t)) \cdot L^2 \cdot n \cdot dL \qquad \textbf{(5-8)}$$

Here, the solid concentrations c_{S1} and c_{S2} are calculated by Eqn. 3-30 and Eqn. 3-31, the critical nuclei radius r_{cr} by Eqn. 2-17, the homogeneous nucleation rate B_{hom} by Eqn. 2-14 and the growth rate G by Eqn. 2-18.

5.2 Discretization

The internal coordinate of the PBE (i.e., the particle size L) was discretized non-equidistant in order to get a finer resolution for smaller particle sizes. The minimum size L_{min} was chosen to be 2 nm, because the critical nuclei radius is at least 1 nm for supersaturations below 10^6 (see Chapter 4). The maximum size L_{max} was determined to be 300 nm. The following rule was used for the lower bound of class i (i.e. L_i), where the total number of classes is called M:

$$L_i = L_{\min} + \left(L_{\max} - L_{\min}\right) \cdot \left(\frac{i-1}{M}\right)^2 \qquad \textbf{(5-9)}$$

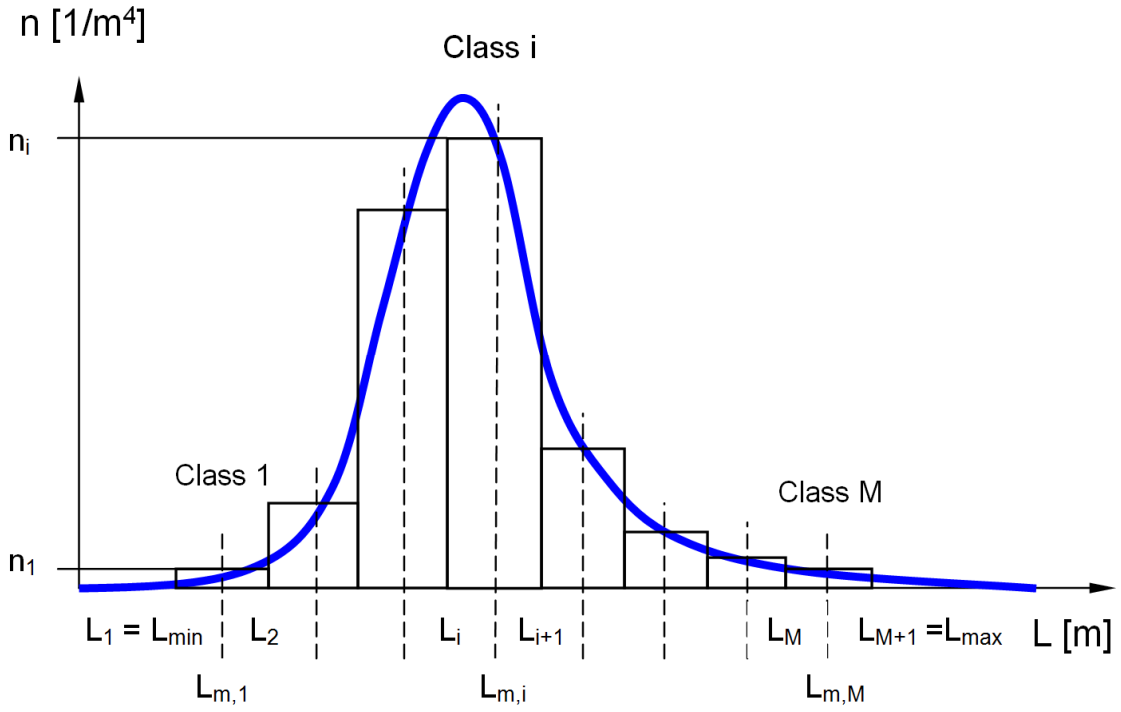


Figure 5-1: Discretization of the particle size with an exemplary size distribution.

The mean size of class i is calculated as the arithmetic mean of upper and lower bound:

$$L_{m,i} = \frac{L_{i+1} + L_i}{2} \qquad \textbf{(5-10)}$$

By the discretization the particle number density n(L,t) is split into a number of M values $n(L_{mi},t)$, called $n_i(t)$. The discretization of the particle size with an exemplary size distribution is illustrated in Figure 5-1.

The growth term in the PBE (Eqn. 5-1) was discretized by the first order upwind scheme, while the integrals in Eqn. 5-7 and 5-8 have been approximated by sums:

$$\frac{dn_i}{dt} = J(S(t)) - \frac{G(L_{m,i}, S(t)) \cdot n_i - G(L_{m,i-1}, S(t)) \cdot n_{i-1}}{L_{m,i} - L_{m,i-1}} + B_{agg,i} - D_{agg,i} \qquad \textbf{(5-11)}$$

$$\frac{dc_1}{dt} = -B_{\text{hom}}(S(t)) \cdot \frac{4}{3} \cdot r_{cr}{}^3 \cdot \pi \cdot c_{S1} - \frac{\pi \cdot c_{S1}}{2} \cdot \sum_{k=1}^{M} G(L_k, S) \cdot L_k{}^2 \cdot n_k \cdot (L_{k+1} - L_k) \qquad \textbf{(5-12)}$$

$$\frac{dc_2}{dt} = -B_{\text{hom}}(S(t)) \cdot \frac{4}{3} \cdot r_{cr}{}^3 \cdot \pi \cdot c_{S2} - \frac{\pi \cdot c_{S2}}{2} \cdot \sum_{k=1}^{M} G(L_k, S) \cdot L_k{}^2 \cdot n_k \cdot (L_{k+1} - L_k) \qquad \textbf{(5-13)}$$

According to Ramkrishna [36] the change of the discrete number density N_i due to aggregation can be described as (where the class j is related to the class k by Eqn. 5-17):

$$\frac{dN_i}{dt} = \frac{1}{2} \cdot \sum_{k=1}^{i-1} \beta_{agg,k,j} \cdot N_k \cdot N_j - N_i \cdot \sum_{k=1}^{\infty} \beta_{agg,k,i} \cdot N_k \qquad \textbf{(5-14)}$$

The number density N_i of a class i is related to the number density distribution by $N_i = n_i \cdot (L_{i+1} - L_i)$. This, and Eqn. 5-14 yields for the aggregation birth rate $B_{agg,i}$ and death rate $D_{agg,i}$:

$$D_{agg,i} = n_i(t) \cdot \sum_{k=1}^{M} \beta_{agg}(L_{m,k}, L_{m,i}) \cdot n_k(t) \cdot (L_{k+1} - L_k) \qquad \textbf{(5-15)}$$

$$B_{agg,i} = \frac{1}{2} \frac{\sum_{k=1}^{i-1} \beta_{agg}(L_{m,k}, L_{m,j}) \cdot n_k(t) \cdot (L_{k+1} - L_k) \cdot n_j(t) \cdot (L_{j+1} - L_j)}{(L_{i+1} - L_i)} \qquad \textbf{(5-16)}$$

In the aggregation birth rate $B_{agg,i}$ of the class i, the size $L_{m,j}$ of the collision partner j is related to $L_{m,k}$ via a simple mass conservation consideration and assuming constant particle densities:

$$L_{m,j} = \sqrt[3]{L_{m,i}^{\ 3} - L_{m,k}^{\ 3}} \qquad \textbf{(5-17)}$$

5.3 Implementation

The system of equations (i.e., Eqns. 5-11 to 5-13, 5-15 and 5-16) was implemented in MATLAB R2008a. MATLAB has also been used in other works for population balance models, e.g. Ward and Yu [37], who implemented the PBE in MATLAB/Simulink. For the solution of the ordinary differential equations (i.e., Eqn. 5-11 to 5-13) the MATLAB built-in solver “ode45”, which is based on the Runge-Kutta algorithm, was used.

The calculation of the aggregation birth rate B_{agg} according to Eqn. 5-16 would produce a mass conservation error, because due to the discretization of the particle size there is no collision partner with the exact size $L_{m,j}$ (see Eqn. 5-17). In order to calculate the aggregation birth rate B_{agg} mass conservative, the algorithm described in the following has to be applied for every combination of collision partners i and j.

The resulting particle size L_{res} after the successful collision of two particles with the sizes $L_{m,i}$ and $L_{m,j}$ is:

$$L_{res} = \sqrt[3]{L_{m,i}^{\ 3} + L_{m,j}^{\ 3}} \qquad \textbf{(5-18)}$$

Two adjacent mean sizes $L_{m,p}$ and $L_{m,p+1}$ have to be located, where the resulting size L_{res} is between:

$$L_{m,p} < L_{res} < L_{m,p+1} \qquad \textbf{(5-19)}$$

The particle, resulting of the collision, has to be partitioned between the two sizes $L_{m,p}$ and $L_{m,p+1}$, under the condition of mass conservation:

$$L_{res}{}^3 = L_{m,p}{}^3 \cdot w_1 + L_{m,p+1}{}^3 \cdot w_2 \quad \textbf{(5-20)}$$

$$w_1 + w_2 = 1 \quad \textbf{(5-21)}$$

Eqn. 5-20 and 5-21 can be solved to obtain the weights w_1 and w_2:

$$w_1 = \frac{L_{m,p+1}{}^3 - L_{res}{}^3}{L_{m,p+1}{}^3 - L_{m,p}{}^3} \quad \textbf{(5-22)}$$

$$w_2 = \frac{L_{res}{}^3 - L_{m,p}{}^3}{L_{m,p+1}{}^3 - L_{m,p}{}^3} \quad \textbf{(5-23)}$$

The aggregation birth rates $B_{agg,p}$ and $B_{agg,p+1}$ are increased by the contributions due to the collision of particles of classes i and j $\Delta B_{agg,i,j,p}$ and $\Delta B_{agg,i,j,p+1}$ (analogous to Eqn. 5-16):

$$\Delta B_{agg,i,j,p} = \frac{w_1}{2} \cdot \frac{\beta_{agg}\left(L_{m,i}, L_{m,j}\right) \cdot n_i(t) \cdot n_j(t) \cdot \left(L_{i+1} - L_i\right) \cdot \left(L_{j+1} - L_j\right)}{\left(L_{p+1} - L_p\right)} \quad \textbf{(5-24)}$$

$$\Delta B_{agg,i,j,p+1} = \frac{w_2}{2} \cdot \frac{\beta_{agg}\left(L_{m,i}, L_{m,j}\right) \cdot n_i(t) \cdot n_j(t) \cdot \left(L_{i+1} - L_i\right) \cdot \left(L_{j+1} - L_j\right)}{\left(L_{p+2} - L_{p+1}\right)} \quad \textbf{(5-25)}$$

The total aggregation birth rate of a class p $B_{agg,p}$ is the sum of the contributions of collision partners i and j smaller than the particles in class p, while the classes i and j are related to each other by Eqn. 5-18:

$$B_{agg,p} = \sum_{i=1}^{p-1} \Delta B_{agg,i,j,p} \quad \textbf{(5-26)}$$

The calculation of the aggregation efficiency α is relatively time-consuming, because the stability ratio involves a numerical integral, which has to be solved for every time step and for each pair of size classes. To keep the calculation time acceptable, a look-up table for

the aggregation efficiency as function of two discretized particle sizes L_1 and L_2 (each M discrete values) and 11 discrete values of the conversion X (0, 0.1, … 1) was precalculated. During the solution of the ODE system, the values for the aggregation efficiency are interpolated from this look-up table. A structogram of the implementation is included in Appendix A and the code is included in Appendix B/IV.

5.4 Results

5.4.1 Base Case

In total there are four parameters unknown for the well-mixed system, namely the interfacial energy constant K, the equilibrium constant c*, the Hamaker constant A and the concentration c_{PDI}^{pzc} of the PDI at pzc. In the parameter studies, shown in Chapter 4, the influence of their values was investigated. Consequently, the equilibrium concentration c* was set to 10^{-10} mol/l, while for the Hamaker constant A the value of $1 \cdot 10^{-20}$ J was chosen. In order to prevent aggregation of the final particles, and to reproduce an aggregation influence at the beginning of the process, the concentration of PDI at pzc c_{PDI}^{pzc} was set to $7 \cdot 10^{-5}$ mol/l, which is approximately the concentration of protamine in the initial mixture. The value of the interfacial energy constant K, which influences the nucleation rate strongly, determines the mean size of the product particles. It was set to 0.39 in order to get particles in the range of 140 nm. For the discretization of the particle size a number of 40 classes was used.

The resulting concentration and supersaturation profiles are shown in Figure 5-2. Starting from the values of the initial mixture, the curves descend with increasing time. After ca. 310 ms the concentration of polyacrylic-acid c_1 is zero, meaning that polyacrylic-acid is totally consumed, while the protamine concentration c_2 remains at a certain value greater than zero, representing the excess of protamine. It is obvious, that the tangents to the concentrations curves at the beginning (t=0) are nearly horizontal. At this point of time growth and aggregation cannot occur, because there are no particles existing, thus nucleation is the only process step which consumes dissolved species at t=0. Due to the well-mixed assumption, the supersaturation has its maximum value at the beginning, hence, also the nucleation rate has its maximum value at t=0. Thus, the (nearly) horizontal tangents imply, that the species consumption by nucleation is negligible, due to the vanishingly small size of the nuclei.

In Figure 5-3 the time dependent number density distribution of the particles is shown. Cross sections of this function for certain times are shown in Figure 5-4. Clearly, the mean size of the particles increases with increasing time (caused by growth and aggregation). Also the total particle number, represented by the area below the curve, grows with increasing time, which is caused by nucleation and slightly counteractive aggregation. After 300 ms the number density distribution does not change significantly, i.e., after the consumption of the precursors the process is finished and no further aggregation takes place. This is due to the choice of the concentration of PDI at pzc c_{PDI}^{pzc}.

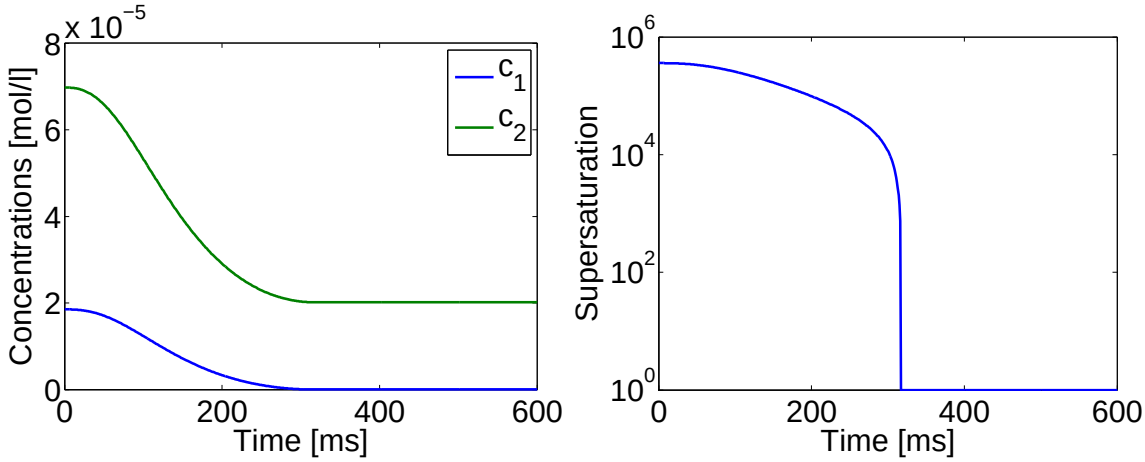


Figure 5-2: Concentrations and supersaturation over time, well-mixed base case.

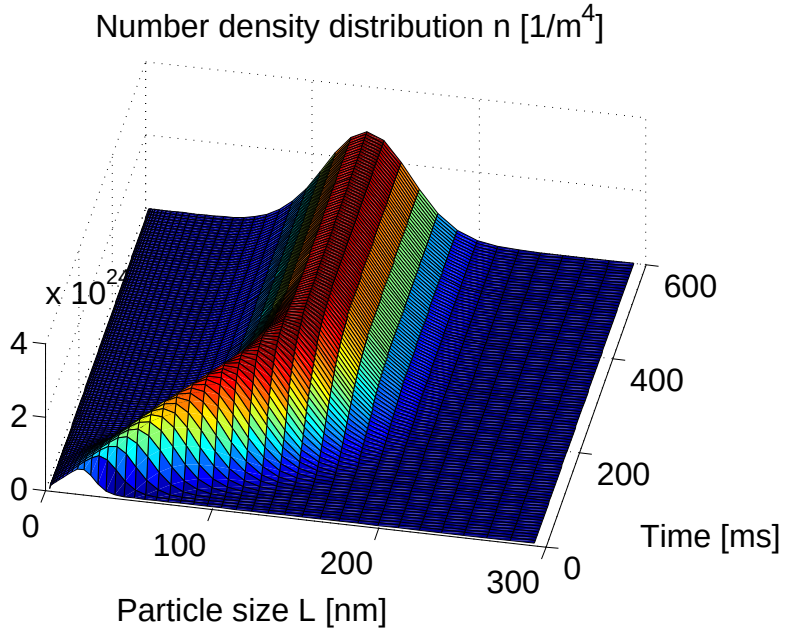


Figure 5-3: Number density distribution as function of time, well-mixed base case.

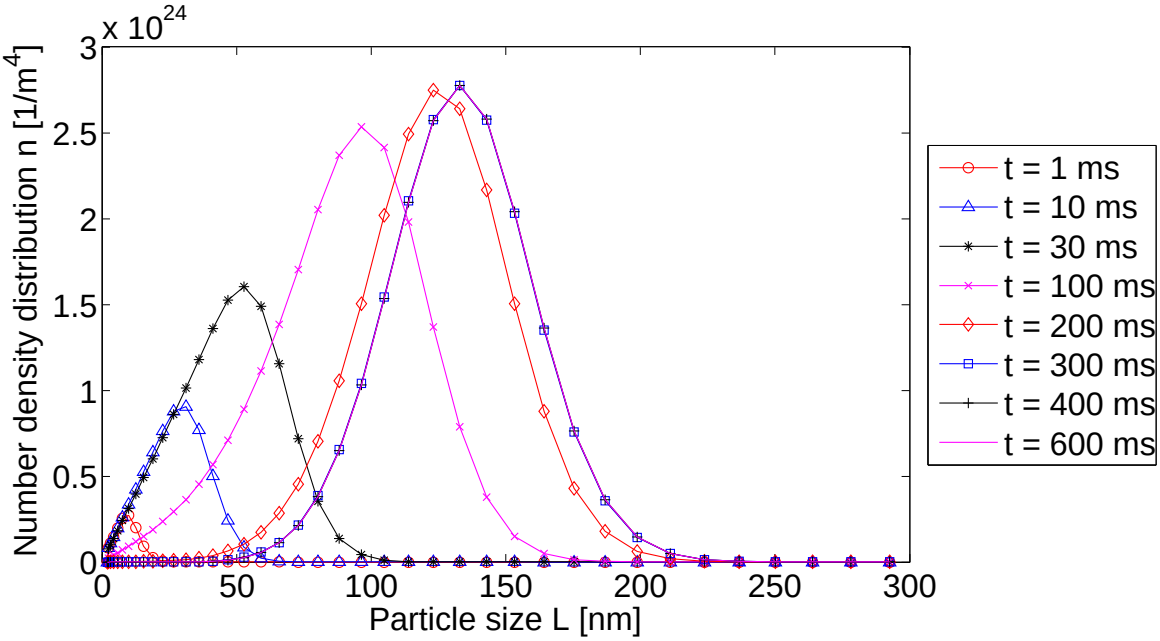


Figure 5-4: Number density distribution for different times, well-mixed base case.

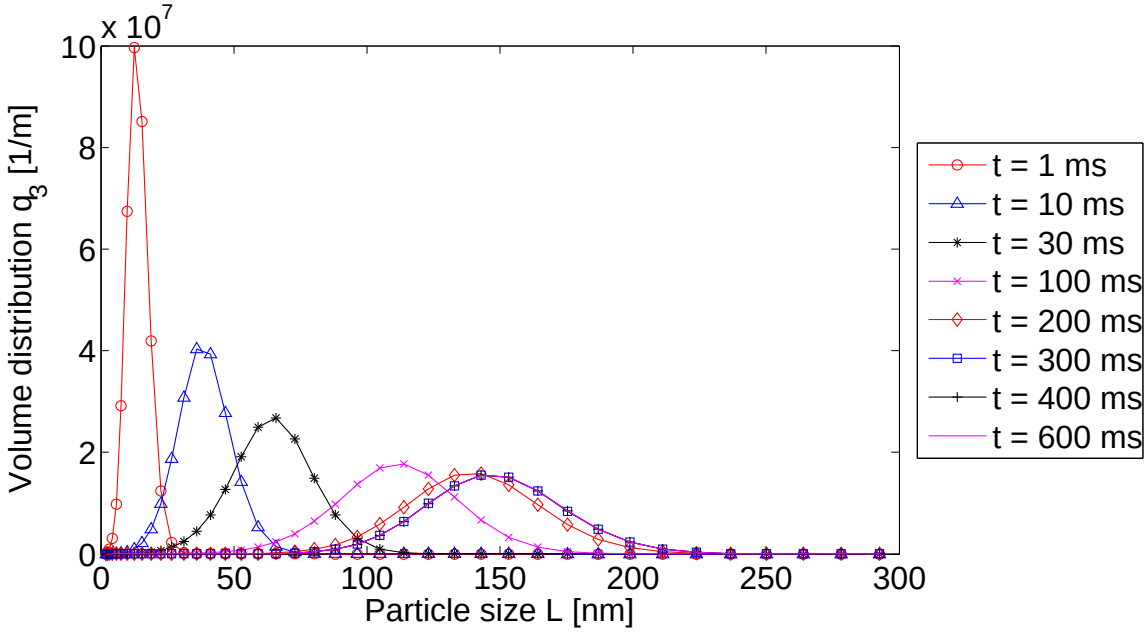


Figure 5-5: Volume distribution for different times, well-mixed base case.

The total number of the particles is represented by the area below the number density distribution. For the final particles, a number of approximately $1.5 \cdot 10^{17}$ 1/m³ with a mean size of 148.4 nm (volume based, see Figure 5-6) is obtained. This value is close to the estimation of $1.56 \cdot 10^{17}$ 1/m³ (for a particle size of 140 nm), which was obtained in Section 3.2, thus, the resulting particle number is plausible.

The volume distribution might have a greater importance than the number density distribution. This is because in the latter (which is a non-normalized number distribution) fine particles contribute proportional to their number, although their volumetric amount is low. Thus, the normalized volume distributions for different times are shown in Figure 5-5. The volume distribution of the final particles is shown in Figure 5-6. The mean particle size based on volume is 148.4 nm, while the standard deviation of the volume distribution is 25.9 nm.

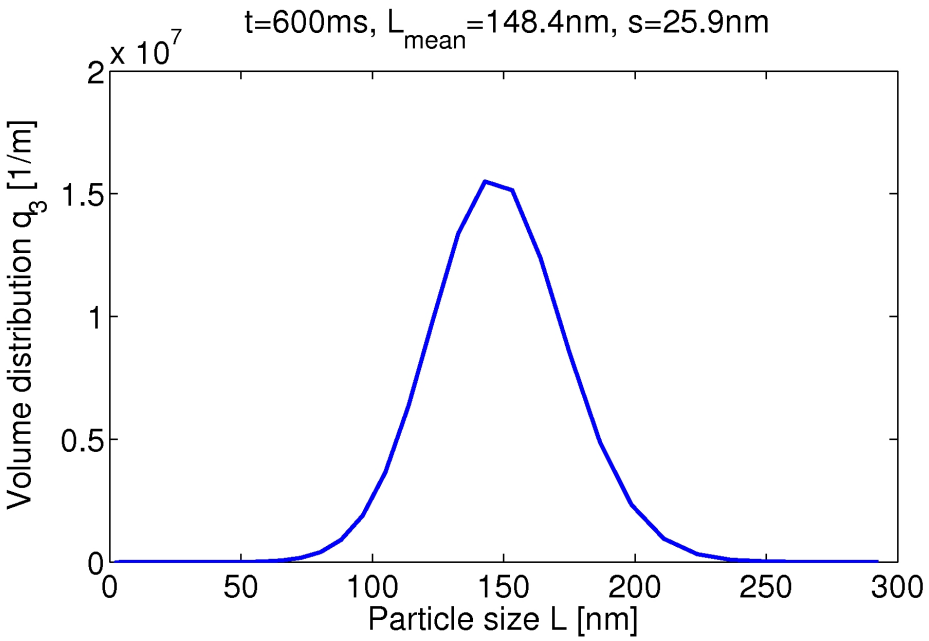


Figure 5-6: Volume distribution of the product particles, well-mixed base case.

5.4.2 Influence of Aggregation

In order to get an insight into the contributions of growth and aggregation, the simulation was run with an aggregation efficiency α=0, i.e., without aggregation. Also, a case with an aggregation efficiency of unity, meaning every collision is successful, was studied. The other parameters are equal to the base case (Section 5.4.1).

The results for the volume distribution of the final particles and for the supersaturation decay are shown in Figure 5-7 and Figure 5-8. The base case ($0<\alpha<1$) does not significantly differ from the case α=0. Hence, the aggregation has only little influence in the base case. Here, the system is at the point of zero charge in the beginning of the process, where no particles are existing. The more particles are generated, the more differs

the actual concentration from the point of zero charge. Thus, the little influence of aggregation is plausible.

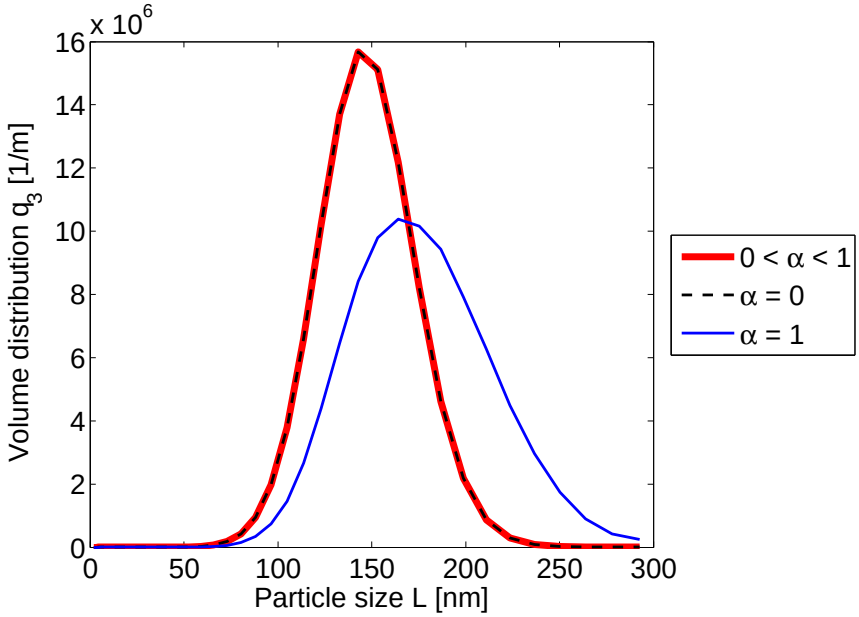


Figure 5-7: Volume distribution of the final particles (resp. t=600ms for α=1) with different aggregation efficiencies α.

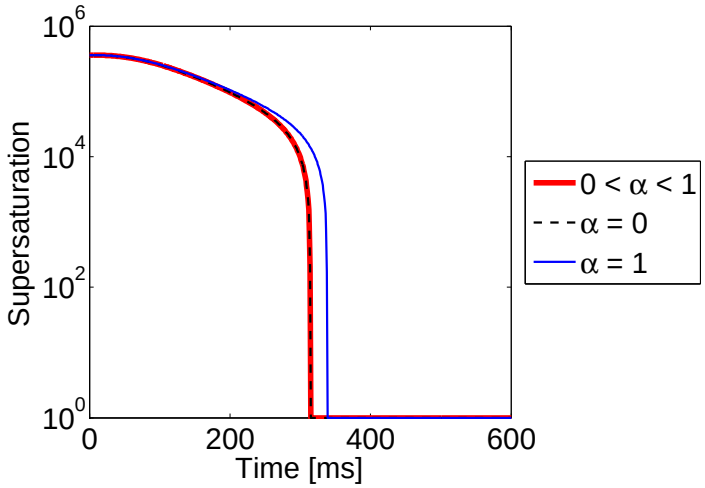


Figure 5-8: Supersaturation over time with different aggregation efficiencies α.

The case α=1 deviates much more from the base case. As shown in Figure 5-9, it is not possible to reach a finished product here, because aggregation does not stop after the supersaturation has reached its final value of 1 (after about 330 ms). The width of the number density distribution is increased with increasing time then. The area below the number density distribution (i.e., the total number of particles) decreases with increasing

time after 300 ms, because aggregation reduces the number of particles and nucleation does not occur at this time.

As the profile for the supersaturation for α=1 shows (see Figure 5-8), the time required for the consumption of the supersaturation is slightly longer here than for the cases with α=0 and 0<α<1. This is caused by the additional enlargement of the particles (due to the strong aggregation), which reduces the total surface and slows down particle growth.

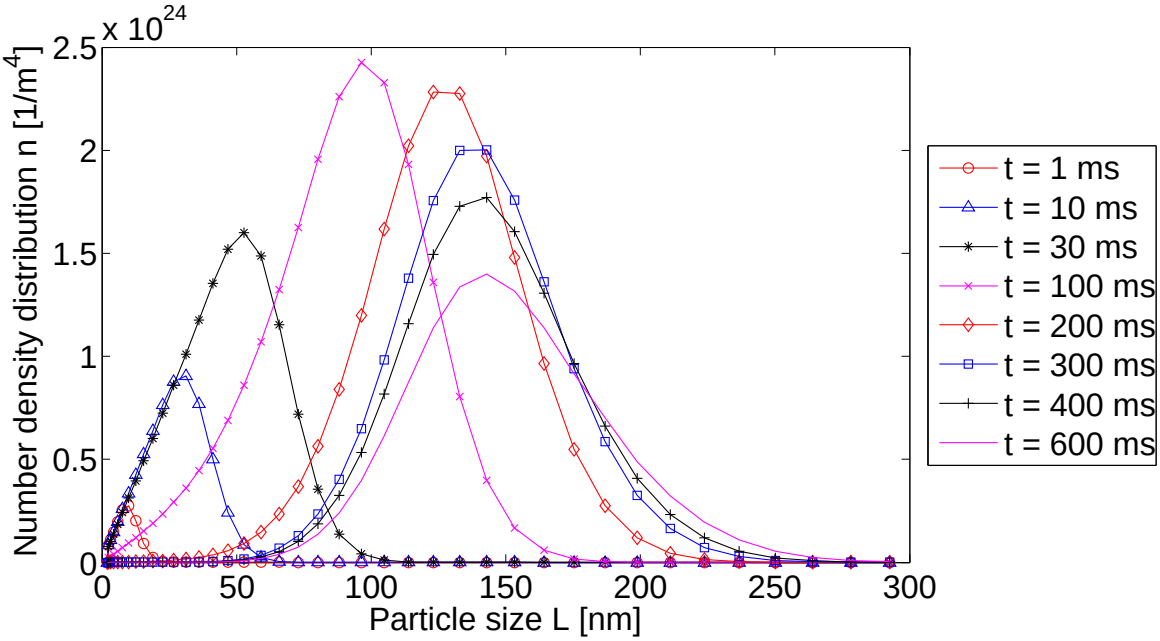


Figure 5-9: Number density distribution for different times (case α=1).

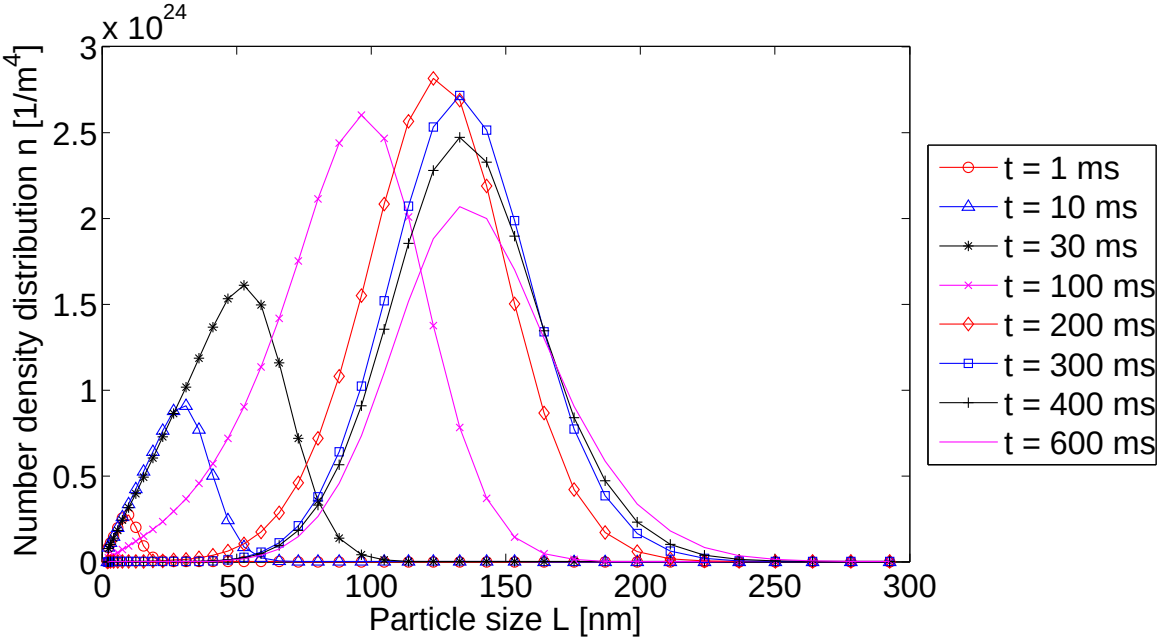


Figure 5-10: Number density distribution for different times (c_{PDI}^{pzc} = 2·10^{-5} mol/l)

To show the influence of the point of zero charge, another case was calculated with a concentration of PDI at pzc c_{PDI}^{pzc} of $2 \cdot 10^{-5}$ mol/l. This is approximately the excess concentration of protamine in the product. As shown in Figure 5-10, the number density distribution does not become constant, which is again caused by aggregation at the end of the process. Compared to the case of α=1 (Figure 5-9), the distributions are more narrow and the mean particle size is lower. This means that a stronger aggregation (in the case of α=1) leads to larger particle sizes and to a wider PSD, which was expected.

5.4.3 Influence of the Interfacial Energy Constant

As shown in Chapter 4, the influence of the interfacial energy constant K on the nucleation rate is high. A variation of K was done in order to investigate the influence on the results of the well-mixed system.

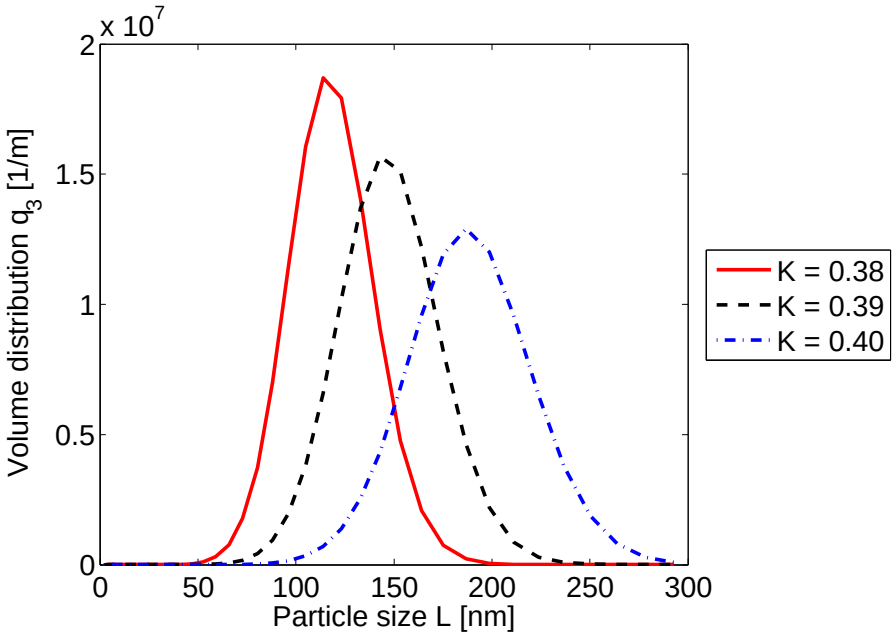


Figure 5-11: Volume distribution of the final particles for different values of K.

In Figure 5-11 and Figure 5-12 the volume distributions of the final particles and the supersaturation decay is shown for three different values of K, respectively. Clearly, both the mean size of the particles and the time required for the consumption of the supersaturation increase with increasing K. A slight increase of K significantly decreases the nucleation rate, which leads to a reduced total number of particles. This causes an increase of the mean particle size and a slowed growth, i.e., a longer process time.

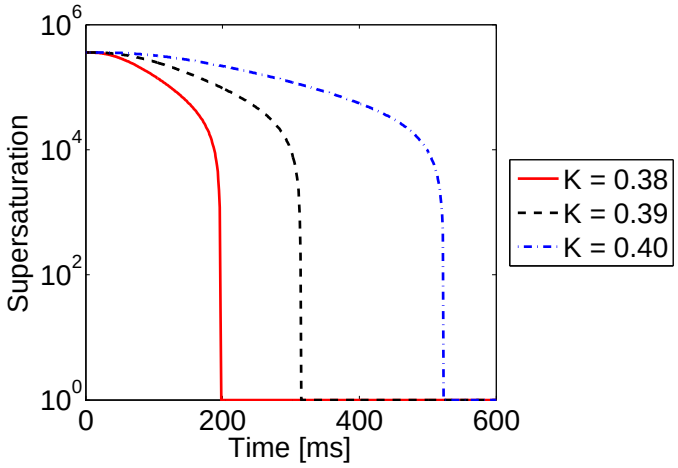


Figure 5-12: Supersaturation over time for different values of K.

5.4.4 Influence of the Equilibrium Concentration

A similar dependency was found by the variation of the equilibrium constant c*. As shown in Figure 5-13 and Figure 5-14, with increasing c* the mean particle size and the process time both increase, whereas the initial supersaturation decreases. Due to the lower supersaturation level, the nucleation rate decreases, which leads to a lower total number of particles, and a longer process time (analogous to Section 5.4.3).

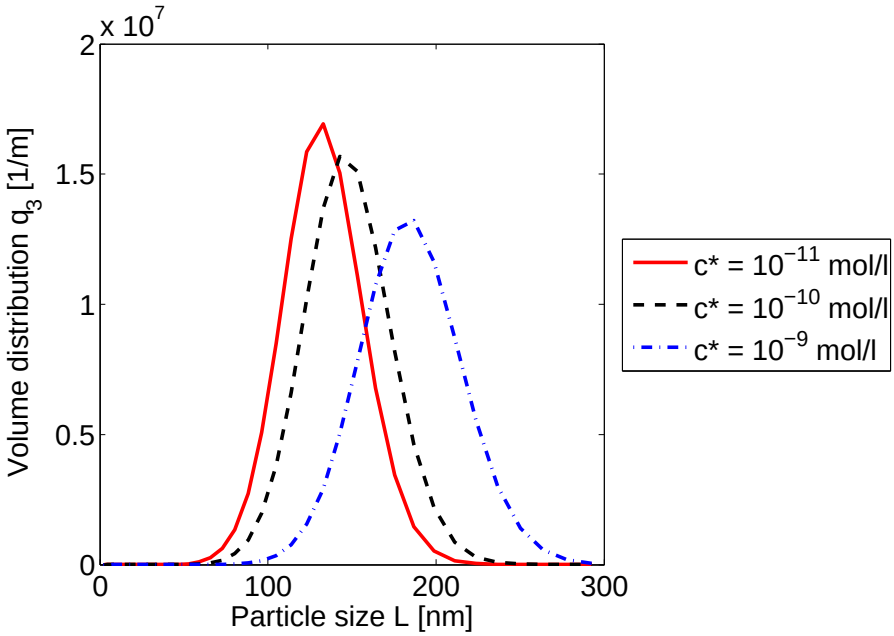


Figure 5-13: Volume distribution of the final particles for different values of c*.

On the contrary to the interfacial energy constant K, the equilibrium concentration c* not only influences the nucleation rate B_{hom} (Eqn. 2-14), but also the growth rate G (Eqn. 2-18). However, the influence of c* on the growth rate G is small, because G is proportional to the diffusion driving force c – c*. During the whole process (except the end) c is much larger than c*. Thus, the growth rate is not significantly influenced by variations of c*. Considering this, a variation of c* has the similar effect as a variation of K. However, for the adjustment of the simulation results, the interfacial energy constant is better suited than the equilibrium constant c*, because the increase of the nucleation rate by variations of c* is limited (see Figure 4-2). Moreover the value of c* can be measured with acceptable accuracy, which is not easily possible for K.

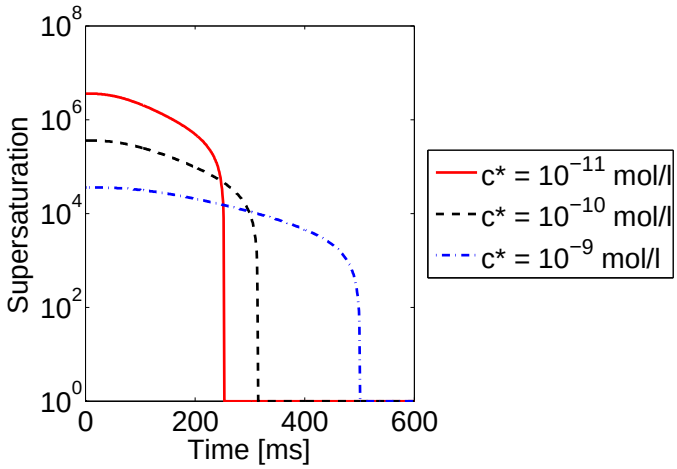


Figure 5-14: Supersaturation over time for different values of c*.

5.4.5 Comparison with Experimental Data

The interfacial energy constant K was adjusted in order to reproduce a mean particle size of 138.8 nm, which was obtained in preliminary experiments (Figure 5-15), performed by the Institute of Pharmaceutical Sciences, University of Graz [38], using the dynamic light scattering equipment "Zetasizer Nano ZS (Green badge)" (Malvern Instruments Ltd.). Originally, an intensity distribution was obtained, which has been transformed into the shown volume distribution by the Zetasizer software [39]. In the calculation, the other unknown parameters have been set to the values used in Section 5.4.1 (i.e., $c^* = 10^{-10}$ mol/l; $A = 10^{-20}$ J; $c_{PDI}^{pzc} = 7 \cdot 10^{-5}$ mol/l). With the value of K = 0.3871 a mean particle size of the volume distribution of 138.8 nm was obtained (see Figure 5-16).

It is obvious, that the width of the simulated PSD (s = 24.5 nm) is significantly lower than the experimental one (s = 59.07 nm). As the simulations were based on the well-mixed assumption, this is plausible, since incomplete mixing will cause a wide distribution of nucleation rates. Clearly, the mixing influence is expected to result in a wider PSD. This expectation was investigated in Chapter 6.

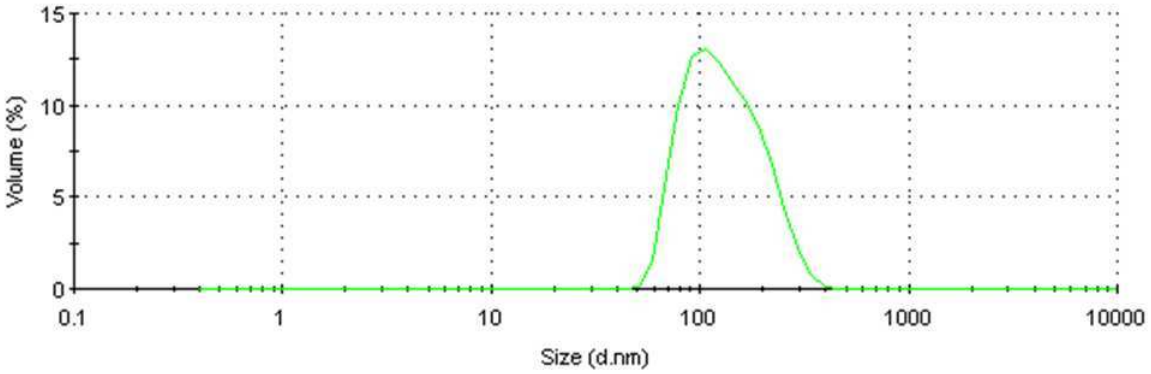


Figure 5-15: Experimental volume distribution (L_{mean}=138.8nm, s=59.07nm) Provided by the Institute of Pharmaceutical Sciences, University of Graz.

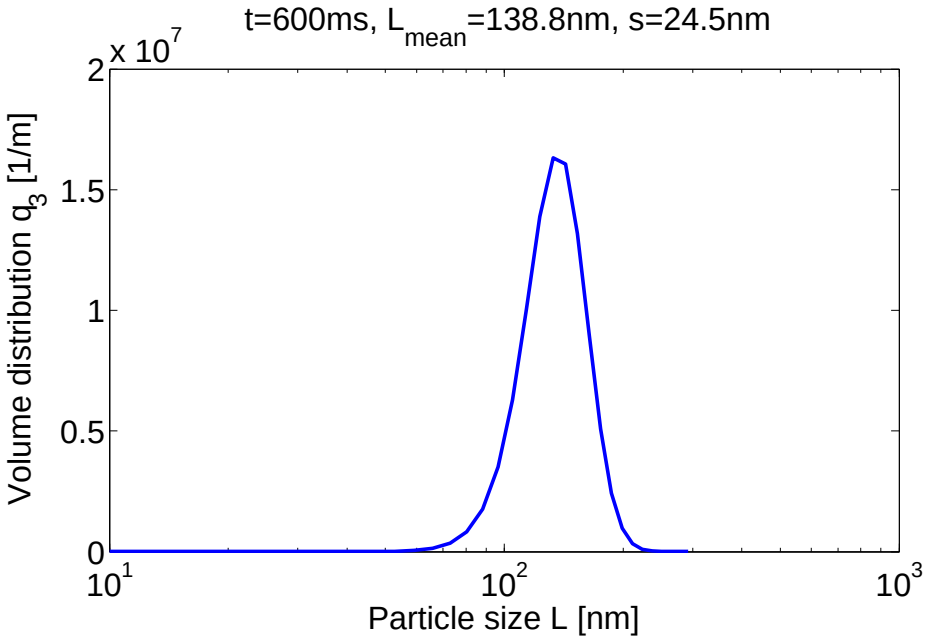


Figure 5-16: Volume distribution of the final particles for K=0.3871.

6 Coupling with a Mixing Model

6.1 Analytical Solution for the Engulfment Model

The engulfment model, mathematically described by Eqn. 2-43 to Eqn. 2-45, was solved analytically. Eqn. 2-44 only contains the unknown X_{me}, hence it can be solved alone by separation of variables and integration. The initial condition $X_{me}(t=0) = X_{me}^{0}$ determines the integration constant C_1:

$$\frac{dX_{me}}{(1-X_{me})X_{me}} = \frac{dt}{t_{me}} \tag{6-1}$$

$$\int\left(\frac{1}{(1-X_{me})}+\frac{1}{X_{me}}\right)dX_{me} = \int\frac{dt}{t_{me}} \tag{6-2}$$

$$\ln\left(\frac{X_{me}}{1-X_{me}}\right) = \frac{t}{t_{me}} + C_1 \tag{6-3}$$

$$X_{me}(t) = \frac{\exp\left(\frac{t}{t_{me}}\right)}{\exp\left(\frac{t}{t_{me}}\right)-1+\frac{1}{X_{me}^{0}}} \tag{6-4}$$

Substituting $X_{me}(t)$ (Eqn. 6-4) in Eqn. 2-43 gives an equation for $X_{mi}(t)$:

$$\frac{dX_{mi}}{dt} = EX_{mi} - EX_{mi}^{2}\left[1+\frac{1-X_{me}^{0}}{X_{me}^{0}}\exp\left(-\frac{t}{t_{me}}\right)\right] \tag{6-5}$$

This is a nonlinear ordinary differential equation of Bernoulli's type. Using the transformation shown in Eqn. 6-6, the ODE gets linear and inhomogeneous (Eqn. 6-7).

$$z(t) = \frac{1}{X_{mi}(t)} \quad \Rightarrow \quad \frac{dX_{mi}}{dt} = -\frac{1}{z(t)^2}\frac{dz(t)}{dt} \tag{6-6}$$

$$\frac{dz}{dt} + Ez = E\left[1 + \frac{1 - X_{me}^{0}}{X_{me}^{0}} \exp\left(-\frac{t}{t_{me}}\right)\right] \tag{6-7}$$

This equation can be solved by an Ansatz (Eqn. 6-8) for the homogeneous solution, and Eqn. 6-9 for the particular solution:

$$z_{\text{hom}} = A \cdot \exp(\lambda t) \tag{6-8}$$

$$z_p = B \cdot \exp(-t / t_{me}) + C \tag{6-9}$$

After solution and backward transformation to $X_{mi} = 1/z$ the solution for $X_{mi}(t)$ is achieved with the use of the initial condition $X_{mi}(t=0)=X_{mi}^{0}$:

$$X_{mi}(t) = \frac{1}{\frac{E\left(1 - X_{me}^{0}\right)}{\left(E - \frac{1}{t_{me}}\right)X_{me}^{0}}\left(\exp\left(-\frac{t}{t_{me}}\right) - \exp(-Et)\right) + \left(\frac{1}{X_{mi}^{0}} - 1\right)\cdot\exp(-Et) + 1} \tag{6-10}$$

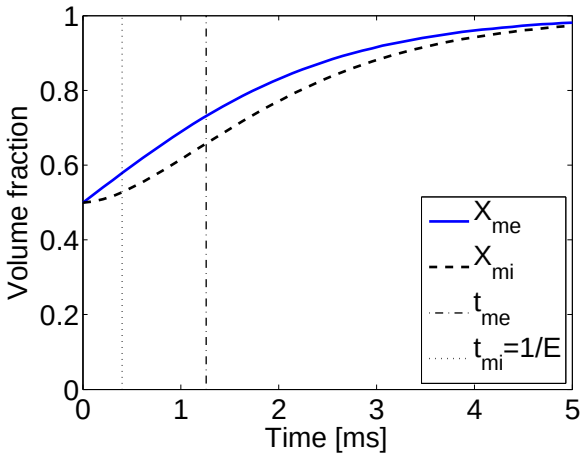


Figure 6-1: Analytical solution for X_{me} and X_{mi} (vertical lines represent t_{mi} and t_{me}).

Exemplary plots of the solutions for $X_{mi}(t)$ and $X_{me}(t)$ with the parameters $\nu = 10^{-6}$ m²/s, ε = 1000 W/kg, d = 0.5 mm and $X_{mi}^0 = X_{mi}^0 = 0.5$ are shown in Figure 6-1. The obtained time scales are t_{mi} = 0.40 ms and t_{me} = 1.26 ms (by Eqn. 2-46 and Eqn. 2-47).

The solution for V(t) (Eqn. 2-45) is easily obtained by dividing Eqn. 2-45 with Eqn. 2-43 and integration with the initial conditions $X_{mi}(t=0) = X_{mi}^0$ and $V(t=0) = V_0$:

$$\frac{1}{V}\frac{dV}{dt} = \frac{1}{X_{mi}}\frac{dX_{mi}}{dt} \tag{6-11}$$

$$\frac{V}{V_0} = \frac{X_{mi}}{X_{mi}^0} \tag{6-12}$$

$$V(t) = \frac{\frac{V_0}{X_{mi}^0}}{\frac{E\left(1-X_{me}^0\right)}{\left(E-\frac{1}{t_{me}}\right)X_{me}^0}\left(\exp\left(-\frac{t}{t_{me}}\right)-\exp(-Et)\right)+\left(\frac{1}{X_{mi}^0}-1\right)\cdot\exp(-Et)+1} \tag{6-13}$$

The limit of V(t) for $t\to\infty$ is V_0/X_{mi}^0, which is the total volume V_{total} (Eqn. 2-49), i.e., after a sufficiently high mixing time the total volume is micromixed.

6.2 Governing Equations for the Coupled Model

The engulfment model describes the time dependencies of the mixromixed and mesomixed volume fraction, i.e. it models mixing as a batch process. Also the used form of the PBE describes only the time dependency of the precipitation process, thus the coupled Model can only describe a batch process. However, the results of these batch calculations can be transferred to a continuous process by a Lagrangian flow consideration, assuming a plug flow and negligible axial dispersion. Then the time coordinate of the batch process is related to the axial coordinate of the continuous process.

The governing equations for the coupled model are analogous to the well-mixed case. However, the equations for the concentrations and the particle number density have to be extended by adequate terms, accounting for the transfer from the non-micromixed volume and for the dilution of the growing, micromixed volume, analogous to Eqn. 2-48:

$$\frac{\partial n(L,t)}{\partial t} = J(L,S(t)) - \frac{\partial(G(L,S(t))\cdot n(L,t))}{\partial L} + B_{agg}(L,t) - D_{agg}(L,t) + E\left(1-\frac{X_{mi}}{X_{me}}\right)\cdot\left(\langle n(L,t)\rangle - n(L,t)\right) \quad \textbf{(6-14)}$$

$$\frac{dc_1}{dt} = -B_{\text{hom}}(S(t))\cdot\frac{4}{3}\cdot {r_{cr}}^3\cdot\pi\cdot c_{S1} - \frac{\pi\cdot c_{S1}}{2}\cdot\int_{L=0}^{\infty} G(L,S(t))\cdot L^2\cdot n\cdot dL + E\left(1-\frac{X_{mi}}{X_{me}}\right)\cdot\left(\langle c_1\rangle - c_1\right) \quad \textbf{(6-15)}$$

$$\frac{dc_2}{dt} = -B_{\text{hom}}(S(t))\cdot\frac{4}{3}\cdot {r_{cr}}^3\cdot\pi\cdot c_{S2} - \frac{\pi\cdot c_{S2}}{2}\cdot\int_{L=0}^{\infty} G(L,S(t))\cdot L^2\cdot n\cdot dL + E\left(1-\frac{X_{mi}}{X_{me}}\right)\cdot\left(\langle c_2\rangle - c_2\right) \quad \textbf{(6-16)}$$

Here, ⟨n(L,t)⟩ is zero, because no particles are existing before mixing takes place. The discretization is identical to the well-mixed case (Section 5.2), because the additional algebraic mixing terms (i.e., the terms that involve the engulfment rate E) do not require a special discretization procedure.

6.3 Implementation

The implementation is similar to the well-mixed case (Section 5.3). There is no additional numerical procedure to solve the mixing model, because the mixing model was solved already analytically. The additional algebraic mixing terms were added to the well-mixed implementation. The code (for MATLAB R2008a) is also shown in Appendix B/VI.

6.4 Results

For the calculations the same parameters have been used as for the well-mixed case (see Section 5.4.1). In addition, two mixing parameters had to be determined, the engulfment constant E (which is the inverse micromixing time t_{mi}) and the mesomixing time t_{me} (Eqn. 2-46 and 2-47). They were calculated from the reactor inlet diameter d and a mean specific

power input ε. The reactor geometry chosen for the time scale considerations in Section 3.3.1 has an inlet diameter of 0.5 mm (which is a typical value for microreactors).

To investigate the mixing influence, a case with a relatively high mean specific power input ε of 100 W/kg (fast mixing) and a case with a relatively low mean specific power input ε of 0.1 W/kg (slow mixing) has been calculated. The parameters are summarized in Table 6-1, where also the mixing-times t_{mi} and t_{me} and the Damköhler number Da is shown. The latter is defined as ratio of a characteristic mixing time to a characteristic reaction time. As the characteristic mixing time the sum of micromixing and mesomixing time was taken, while for the characteristic reaction time the characteristic growth time of nano particles (equal to 60 ms) was used (see Section 3.3.3). This is motivated by the fact, that not the (very fast) protonation reactions are critical for particle formation, but particle growth. The characteristic nucleation time could also be used as the characteristic reaction time. However, the precise estimation of the nucleation time is more complicated, as detailed in Section 3.3.2.

$$Da = \frac{t_{mi} + t_{me}}{t_{Growth}} \qquad \textbf{(6-17)}$$

	ε [W/kg]	d [m]	t_{mi} [ms]	t_{me} [ms]	Da
Well-mixed	∞	-	0	0	0
Fast-mixed	100	0.0005	1	3	0.07
Slow-mixed	0.1	0.0005	40	27	1.12

Table 6-1: Mixing parameters for the three different considered conditions.

The resulting time profiles for the concentrations and the supersaturation are shown in Figure 6-2 and Figure 6-3. Clearly, the fast-mixed case does not much differ from the well-mixed case, only the first milliseconds are influenced by mixing. On the contrary, the slow mixed case is strongly influenced by mixing and differs from the well-mixed case over the total process time.

The volume distributions of the final particles for the three considered cases are shown in Figure 6-4, while the related data are shown in Table 6-2. The difference between the well-mixed and the fast-mixed case is little, as expected. However, also the volume distribution of the slow-mixed case is relatively similar to the well-mixed case. The main difference is,

that the mean particle size in the slow-mixed case is recognizably larger than in the well-mixed case. That is caused by a lower maximum supersaturation, i.e., a lower nucleation rate in the slow-mixed case, as shown in Figure 6-3.

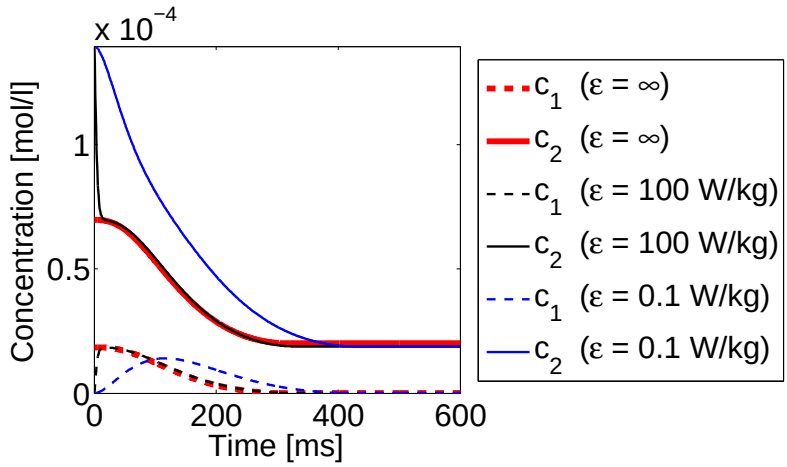


Figure 6-2: Concentrations in the micro mixed compartment over time for different mixing conditions.

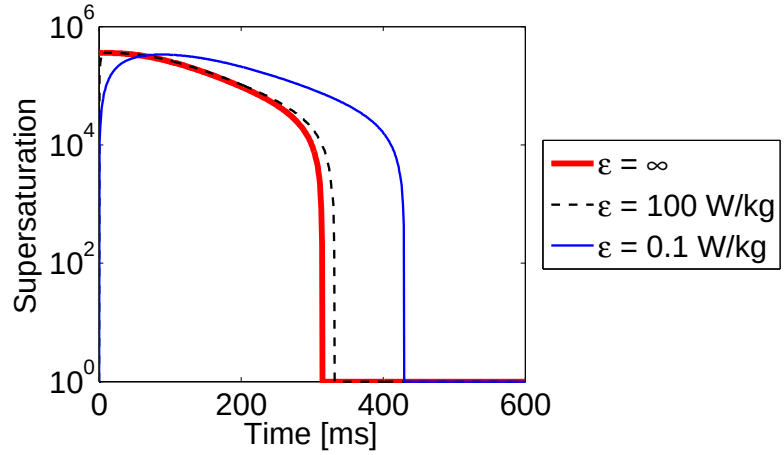


Figure 6-3: Supersaturation over time for different mixing conditions.

	ε [W/kg]	d [m]	L_{mean} [nm]	s [nm]	$s_{relative}$ [%]
Well-mixed	∞	-	148.4	25.9	17.5
Fast-mixed	100	0.0005	150.0	26.4	17.6
Slow-mixed	0.1	0.0005	159.1	28.9	18.2

Table 6-2: Results for the mean particle size and the standard deviation for different mixing conditions.

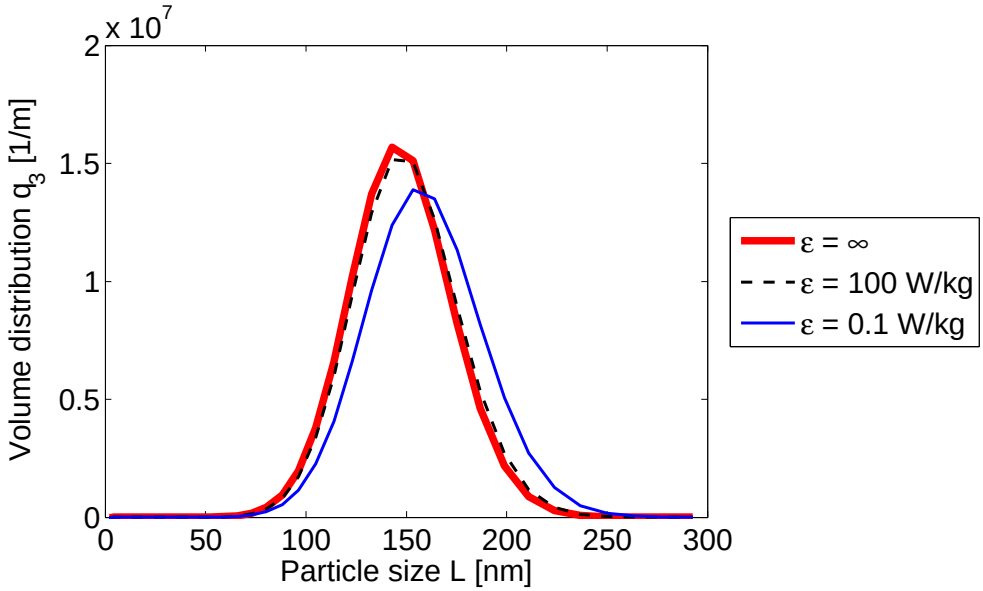


Figure 6-4: Volume distribution of the final particles for different mixing conditions.

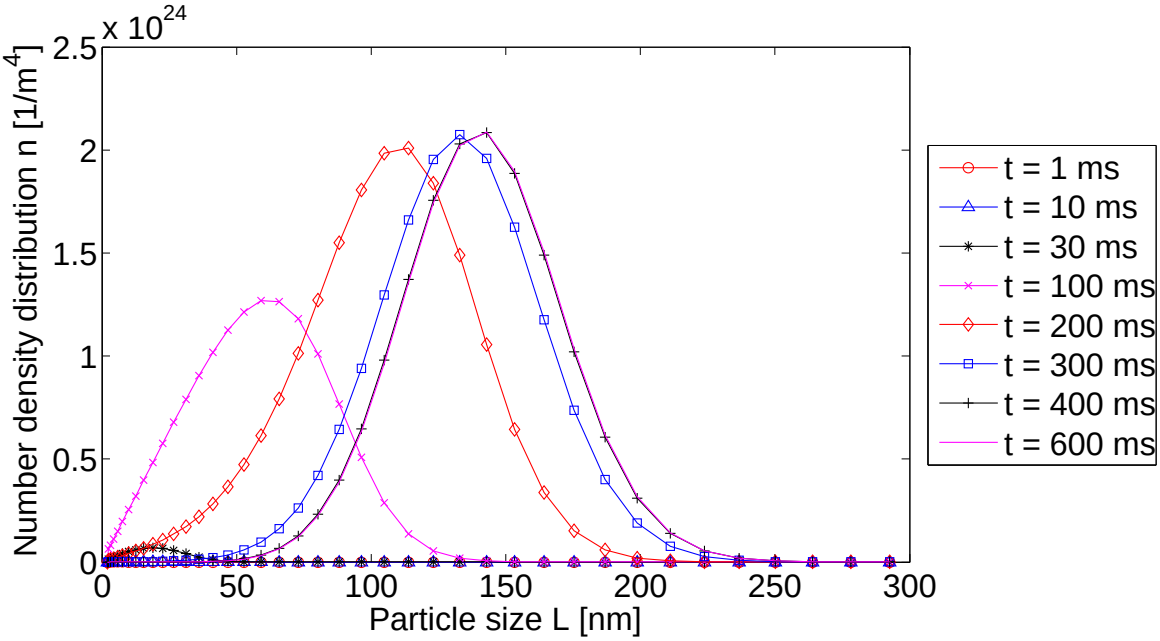


Figure 6-5: Number density distribution for different times for the slow-mixed case.

Unexpectedly, the relative standard deviation $s_{relativ}$ of the slow-mixed case is nearly equal to the well-mixed case. This can be explained by the high sensitivity of the nucleation rate with respect to the supersaturation (for details see Chapter 4). As obvious in Figure 6-3, the maximum supersaturation in the slow-mixed case is reached after about 100 ms. During a certain amount of these first 100 ms, the supersaturation (and also the nucleation rate) is significantly lower than the maximum value (notice the logarithmic scaling). Thus, during this first stage the generated number of particles is negligibly small, hence also growth and

aggregation cannot set in. This is also confirmed by Figure 6-5, where the number density until t = 30 ms s is nearly zero. Thus, the process starts as soon as the supersaturation has overrun a threshold value, over which significant nucleation occurs. The remaining process after the first 100 ms has a similar supersaturation decay than the well-mixed case, only the total time is longer, caused by the lower maximum supersaturation level. Thus, the basic difference in the supersaturation profiles of the well-mixed and the slow-mixed case (i.e., the difference in mixing times during the first stage where the supersaturation is relatively low) does not much influence the product.

As shown in Section 5.4.5 the size distribution obtained in a preliminary experiment is significantly wider than the simulated size distribution of the well-mixed case. It is not possible to reproduce the width of the experimental distribution with the mixing model, because the latter does not change the relative standard deviation of the PSD significantly.

7 Scale-up

There are different approaches to investigate scale-up. In the simplest case an analysis of the characteristic time scale for each process step can already provide valuable information for scale-up. Thus, the limiting process steps are identified, and their ratio (i.e., a dimensionless number) is held constant during scale-up. The precondition for such an approach is, that relations for all characteristic time scales are available. A more sophisticated approach is, to perform simulations at different length scales and to derive scale-up relations from the results. The quality of these results depends on the simulation model used.

In this work we have concentrated on scale-up relevant time scales for the precipitation process. Note, the well-mixed precipitation model (see Chapter 5) does not depend on a length scale of the reactor, hence its scale-up analysis is trivial. The precipitation model coupled with the engulfment model depends on the length scale of the reactor. Thus, this model is able to provide a basic understanding of scale effects on product properties, i.e., the product particle size distribution. However, the engulfment model for mixing is a relatively simple description of the mixing process, and only requires a single parameter for the reactor geometry (i.e., the inlet diameter d). Thus, more sophisticated scale-up considerations have to use improved mixing models, e.g., based on CFD, coupled with the dynamics of the precipitation process. We have not performed such sophisticated computations, and addressed this extremely challenging task to future work.

7.1 Scale-up via Characteristic Time Scales

The characteristic time scales for the process steps have been analyzed in Section 3.3. The time scales for nucleation, growth and aggregation are length scale independent. The only time scale which depends on the length scale of the reactor is the mixing time (see Eqn. 3-35). Theoretical and experimental investigations on mixing in relevant reactor geometries (i.e., CIJRs) were performed by Johnson and Prud'homme [34]. To ensure equal process conditions during scale-up, the characteristic mixing time has to be constant (Eqn. 7-1), as the dynamics of the precipitation process are scale invariant. The results of Johnson and Prud'homme [34] suggest the following relation between the inlet diameter d (i.e., the length scale) and the inlet velocity u:

$$\tau_m = 5400 \cdot \sqrt{\frac{\nu \cdot d}{u^3}} = const., \text{ thus} \quad \textbf{(7-1)}$$

$$u \propto d^{1/3} \quad \textbf{(7-2)}$$

Furthermore, scale-up relations for the flow rate F_V (Eqn. 7-3), Reynolds number Re (Eqn. 7-4) and residence time τ (Eqn. 7-5) can be derived from this result:

$$F_V \propto u \cdot d^2 \propto d^{7/3} \quad \textbf{(7-3)}$$

$$Re = \frac{u \cdot d}{\nu} \propto d^{4/3} \quad \textbf{(7-4)}$$

$$\tau = \frac{V_R}{F_V} \propto \frac{d^3}{u \cdot d^2} \propto d^{2/3} \quad \textbf{(7-5)}$$

As already noticed in Section 3.3.1, the relation for the characteristic mixing time was developed for Reynolds number of 150<Re<3,000 and a Schmidt number of Sc=1,000. In our work the Schmidt number is about 6,000. However, due to missing data for Sc>1,000, we have to assume that the principal functional dependency shown in Eqn. 7-1 is also valid for our system. This is supported by the statement of Johnson and Prud'homme [34], that mixing in this regime of high Sc has only little dependency on the diffusion coefficient in the system. Also, Sc is constant during scale up, hence its relative impact on mixing performance between scales can be expected to be small.

Plots of the variations of the inlet velocity, Reynolds number and residence time (Eqn. 7-2 to Eqn. 7-5) during scale-up are shown in Figure 7-1. Clearly, the enlargement of the reactor by factor of 10 corresponds to an increase of the inlet velocity by a factor of approximately 2. This results in an increase of the flow rate by a factor of ca. 200. The Reynolds number increases by factor 20, which may cause a change of the flow regime. This is critical, since the used relation for the characteristic mixing time is only valid in the Reynolds number range of 150 to 3,000. The mean residence time for the length scale

factor 10 is about 5 times longer. This may be critical for precipitation processes, in which the product is not stabilized, and aggregation occurs after full conversion of the precursors.

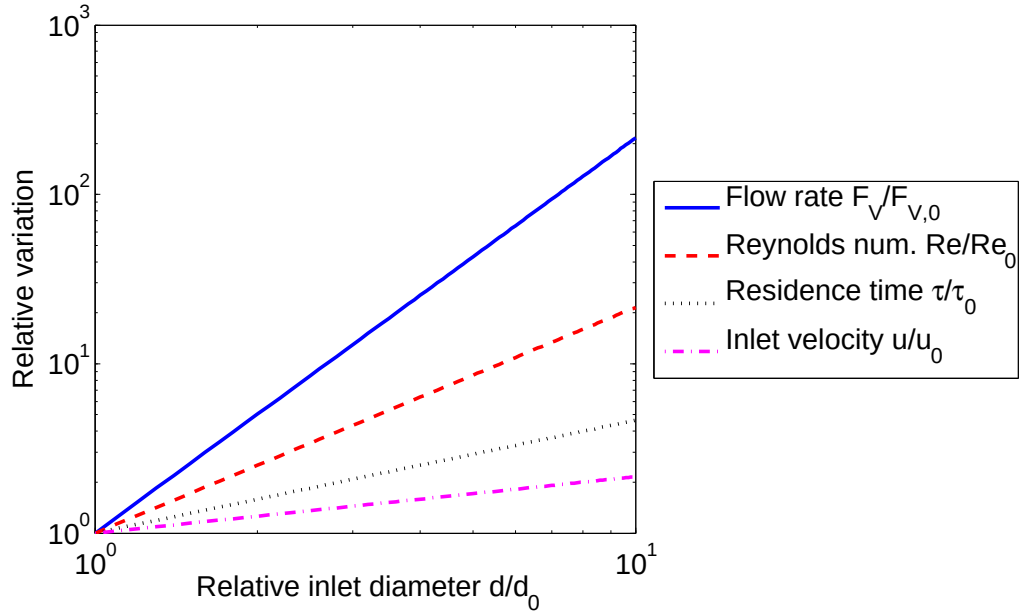


Figure 7-1: Variation of various process parameters during scale-up based on Eqn. 7-1.

Johnson and Prud'homme [34] showed in their work that various mixing time scales in the case of engulfment limited micromixing have the same scaling as momentum diffusion (i.e., $\tau_m \sim (\nu/\varepsilon)^{1/2}$, for 150<Re<4,000). Hence, the condition ε = const. keeps the mixing time constant, which yields:

$$\varepsilon = \frac{P}{m} = \frac{\Delta p \cdot F_V}{V_R \cdot \rho} \propto \frac{\zeta \cdot \rho \cdot u^2 \cdot d^2 \cdot u}{d^3 \cdot \rho} = \zeta \cdot \frac{u^3}{d} = const. \qquad \textbf{(7-6)}$$

Here, P is the power input, m the mass of the reactor content, Δp the pressure loss in the reactor, V the reactor volume, ρ the density of the reactor content and ζ the friction factor of the reactor. Commonly the friction factor ζ depends on the Reynolds number Re. For sufficiently high Reynolds numbers, i.e., in the fully turbulent regime, it typically reaches a constant value. For this case the condition of ε = const. leads to the same dependency than Eqn. 7-2.

7.2 Scale-up based on the Engulfment Model

In addition to the analysis of time scales (see Chapter 7.1), we calculated scale-up relevant information using the engulfment model coupled with the precipitation model. The parameters for the PBE are the same as the ones used in Chapter 5 and Chapter 6 (i.e., K = 0.39; c* = 10^{-10} mol/l; A = 10^{-20} J; c_2^{pzc} = 7.10^{-5} mol/l).

In the engulfment model, mixing is determined by two parameters, the mean specific power input ε and the inlet diameter d. For four different values of the mean specific power input ε (i.e., 1, 10, 100 W/kg and ∞ = well-mixed) the relative inlet diameter d/d_0 was varied from 1 to 100 (with d_0=0.5 mm) and the product particle size distribution has been calculated. The influence of these conditions on the mean product particle size is shown in Table 7-1 and Figure 7-2.

d/d_0	ε = 1	10	100	∞ [W/kg]
1	151.8	150.3	150.0	148.4
3	154.2	151.0	150.2	148.4
10	161.4	153.8	151.0	148.4
30	174.7	160.6	153.6	148.4
100	199.1	175.4	161.1	148.4

Table 7-1: Mean particle size L_{mean} [nm] for different length scales d/d_0 and specific power inputs ε.

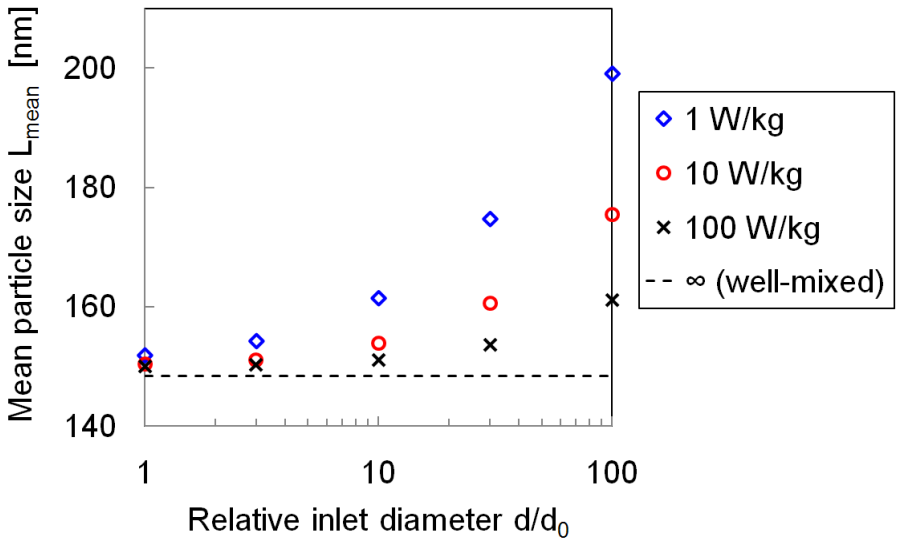


Figure 7-2: Scale-up dependency of the mean particle size for different mean specific power inputs ε.

Clearly, in the well-mixed case the mean particle size is independent on the length scale. The higher the mean specific power input ε, i.e., the higher the turbulence of the flow field, the smaller is the impact on the particle size distribution during scale-up.

For micro reactors (i.e., reactors with length scales in the range of 1 mm and below) the results are close to the well-mixed case. In the case of larger length scales the results differ more from the well-mixed results, meaning, mixing is more critical in these reactors. The larger the reactor, the higher is the required mean specific power input (i.e., the turbulence) to obtain a product similar to the well-mixed product.

The simulation results obtained with the engulfment-model show, that for a constant specific power input ε the mean particle size is not scale-up independent. The reason for the scale-up dependency of the mean particle size for ε = const. is the definition of characteristic time scales. The time scales in the engulfment model (Eqn. 2-46 and Eqn. 2-47) depend on the viscosity ν, the mean specific power input ε, and on the inlet diameter d. Clearly, for constant values of ν and ε the mean particle size of the product must be length scale dependent in the case of the engulfment model.

However, for sufficiently fast mixing conditions, i.e., a sufficiently high mean specific power input ε (e.g., ε ~ 10 W/kg) and moderate changes in the length scales (i.e., d/d_0 ~ 10), the scale-up dependency of the mean particle size is small. Thus, for fast mixing conditions the results agree approximately with the scale-up relation ε = const. shown in Section 7.1.

8 Estimation of the Discretization Uncertainty

A generally accepted procedure for the estimation of the discretization uncertainty in CFD-simulations is the grid convergence index (GCI, see Celik et al. [40]). This index is applicable to a spatial discretization necessary in CFD simulations. The discretization used in this work concerns the particle size, i.e., the recommended concept is not strictly valid here. However, the general concept was adapted.

The recommended procedure by Celik et al. uses three different grids to quantify the discretization error. That is useful for CFD, where the duration of one simulation run is typically relatively long (i.e., several hours to days). Compared to a 3-dimensional CFD-simulation, the 1-dimensional well-mixed PBE solved in this work is much less time consuming. Therefore, it was possible to use a larger number of different grid sizes and analyze the trends via a curve fit. The data generated by seven simulations are shown in Table 8-1, where the number of grid nodes (size classes) was varied from 30 to 100. Also, the relative error for different grids was calculated (with respect to the extrapolated values) and the relative calculation time has been computed. Notice, that for a decrease of 50% in the discretization error the calculation time has to be increased by a factor of 6.

M	1/M	L_{mean} [nm]	rel. error [%]	≈ rel. calc. time
30	0.033	154.2	20.0	0.5
40	**0.025**	**150.0**	**16.7**	**1.0**
50	0.020	147.5	14.8	2.0
60	0.017	140.9	9.6	3.5
70	0.014	139.7	8.7	5.5
80	0.013	138.8	8.0	8.0
100	0.010	135.8	5.7	16.0
∞	0	128.5		

Table 8-1: Dependency of the mean particle size L_{mean} on the number of classes M.

In Figure 8-1 the data points are plotted and a linear trend line was fitted. The extrapolated value of L_{mean} for an infinite number of classes is shown in the last line of Table 8-1.

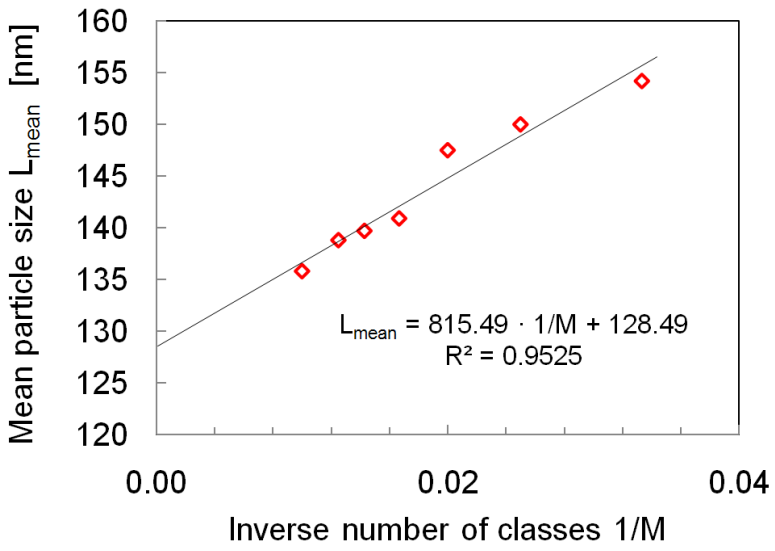


Figure 8-1: Dependency of the mean particle size on the number of classes.

9 Conclusions and Outlook

9.1 Conclusions

The aim of this work was to develop a numerical tool to study the precipitation of polyacrylic-acid/protamine nanoparticles. Also, the scale-up behaviour of this complex process was studied.

For the precipitation of inorganic salts, state-of-the-art simulation approaches based on the population balance equation (PBE) exist in literature. Thus, numerical investigations using these approaches have been recently used, e.g., for the prediction of $BaSO_4$ precipitation. In combination with the quadrature method of moments (QMOM) to solve the PBE, it is even possible to couple PBE with computational fluid dynamics (CFD). Such a sophisticated CFD simulation of has not been performed here. However, we were the first who studied precipitation of organic nanoparticles, as well as the impact of mixing, various other process parameters and scale-up. Our sensitivity analysis of the most critical parameters revealed, that the prediction of organic nano particle precipitation is significantly more challenging than that of, e.g., inorganic salts. This is due to the fact, that the structure of the molecules (polyacrylic-acid and protamine) is not exactly known. Assumptions had to be made in order to perform numerical simulations. Our model is robust in the sense, that the general trends of our predictions do not depend on the exact values of the parameters and fit experimental data reasonably well. A parameter study for the nucleation model was done, in order to find reasonable ranges for the values of the unknown parameters (interfacial energy constant K and equilibrium constant c*). The nucleation rate is highly sensitive to the values of K and c*. Especially the parameter K, which cannot be determined experimentally, is well suited to adjust the results of the simulation. However, the correct prediction of the width of the particle size distribution is difficult, and even the inclusion of a mixing model did not significantly improve the results. Moreover, the high sensitivity of the nucleation rate on the supersaturation was shown, which causes a supersaturation threshold. Below this threshold, no significant nucleation is possible, because of the properties of the precursor molecules.

Furthermore, a parameter study of the aggregation model was performed. We found, that the aggregation behaviour depends strongly on the particle sizes and on the deviation of the actual concentration from the point of zero charge (pzc). E.g., setting the pzc near the final

concentration, the particles continue to grow after the supersaturation approaches unity, because of the vanishing electrostatic repulsion. Aggregation is only possible if the actual concentration is in a certain range around the pzc. Otherwise, the electrostatic repulsion forces prevent aggregation. Aggregation occurs mostly between pairs of collision partners with different sizes. This is caused by the different mobility of different sized particles, leading to higher collision rates. Aggregation between large particles (i.e., particles in the range of 100 nm) is unlikely, compared to relatively small particles with a size around 10 nm. This is due to the increased electrostatic potentials between large particles.

We included the engulfment model in order to investigate the effect of incomplete mixing on the product particle size distribution. It was shown, that the mixing model influences the time profiles of the concentrations and the supersaturation significantly. Also, the mean size of the particles is influenced by the mixing model. However, the expected increase of the PSD width due to the mixing influence (compared to the well-mixed solution) was not observed. The reason is assumed to be the high sensitivity of the nucleation rate on the supersaturation. Clearly, the PBE coupled with the engulfment-model was not able to reproduce the width of the experimentally determined PSD.

Our predictions of product properties were checked with mass balances, in order to prove the plausibility of the simulation results. Also, an estimation of characteristic time scales for the process steps was performed. Clearly, the time scales for nucleation, growth and aggregation are in the same order of magnitude, while the time scale for mixing depends strongly on the length scale of the reactor and the energy dissipation in the reactor. It was also shown, that a microreactor, i.e., a reactor with dimensions of a few millimetres maximum, is well suited to ensure the required short mixing time for this precipitation process.

9.2 Future Directions

It is currently unclear what causes the experimentally observed relative wide spread in the PSD. In the preliminary experimental investigations, the mixing conditions were not quantified, and it may be anticipated, that the wide PSD was caused by this undefined mixing conditions. However, it is also possible, that the engulfment model (which assumes well-mixed conditions in the micromixed volume) is not fully appropriate to reconstruct the experimental mixing conditions. If the mixing conditions are not the reason for the experimentally observed width of the PSD, other phenomena have to be considered, e.g.,

growth rate dispersion, which is frequently used in PBE modeling of precipitation processes involving crystalline substances (see, e.g. Stahl et al. [14]). Also, a dependency of the surface tension on the actual liquid concentrations could lead to the spread in the PSD. Such an approach seems more realistic for the amorphous substance studies in our work, since growth rate dispersion is caused by differences in the crystalline structure of product particles.

The analysis of characteristic time scales shows, that the mean inlet velocity should scale with $u \sim d^{1/3}$ in a CIJR. In the case of a constant reactor friction factor this is equal to ε = const. The result $u \sim d^{1/3}$ is only valid for Reynolds numbers between 150 and 3,000. The relation for the characteristic mixing time, see Eqn. 7-1, depends on the geometrical details of the reactor. For other reactor geometries, an adequate relationship for the mixing time has to be found. A way to determine such relationships would be, to analyze mixing in new reactors via the simulation of a tracer experiment. Thus, at t=0 a concentration jump of a non-reacting scalar at the inlet of such a reactor could be imposed. Measuring the residence time distribution, and fitting this distribution with a simple model, e.g., a plug-flow reactor with axial dispersion, could be an easy way to quantify the (overall) mixing time (Note, $\tau_{mix} = l^2/D_{ax}$ for such a simple axial dispersion model, where l is the length of the reactor, and D_{ax} the axial dispersion coefficient). However, a micro mixing time could not be deducted from such an estimate of the overall mixing time. Micro mixing should be faster, and Johnson and Prud'homme [34] showed, that a single mixing time is sufficient to characterize the performance of a reactor. The key issue is, however, to find a correlation for the mixing time for low and intermediate Reynolds numbers (i.e., below 150), as well as for Re > 3,000. This is because the Reynolds numbers increases during scale up when using $u \sim d^{1/3}$ and a constant fluid viscosity. Thus, a too massive increase in the geometrical size of the reactor cannot be done, as there are no correlations for the mixing time available for a wide range of Re.

Another approach would be to start from the condition of a constant mean specific power input ε and to simulate the flow field via CFD (not the total precipitation process). For different reactor length scales the corresponding volumetric flow rates (and the mean inlet velocities) can be found by holding ε = const. However, it is unclear if this approach is also valid for low Re numbers, since there is a change in the flow regime. Anyhow, single-phase fluid flow simulations are standard nowadays, and could be done with relatively high precision at comparably low costs.

For sufficiently fast mixing conditions, i.e., sufficiently high values of the mean specific power input ε, predictions with the coupled PBE/mixing model showed only a low sensitivity of the particle size distribution with the scale of the reactor. However, the predicted trends are plausible only for the mean value of the particle size (i.e., increasing particle size with increasing scale of the reactor). The (relative) width of the PSD does not well agree with experimental results. Improvement is needed here in future work. Other mixing models, e.g. the segregated-feed-model (described in Section 2.5.2), which calculate the precipitation process within two well-mixed compartments with different conditions in parallel, could be used to investigate this problem. However, other authors (e.g., Alvarez and Myerson [41]) found, that neither a plug flow model, nor an axial dispersion model could describe the width of the product PSD accurately. Spatially resolved simulations, i.e., a CFD-simulation of the precipitation process, could lead to a deeper insight here. However,

"Turbulent precipitation still poses a challenge for comprehensive models integrating fluid dynamics and PBE modelling."

as noticed by Rigopoulos [21].

10 References

[1] M. Aoun, E. Plasari, R. David, J. Villermaux. A Simultaneous Determination of Nucleation and Growth Rates from Batch Spontaneous Precipitation. Chemical Engineering Science 54 (1999), 1161 – 1180.

[2] A. R. Paschedag. CFD in der Verfahrenstechnik. 1st Edition. Wiley-VCH, Berlin (2004).

[3] C. Beck, S. V. Dalvi, R. N. Dave. Controlled Liquid Antisolvent Precipitation using a Rapid Mixing Device. Chemical Engineering Science 65 (2010), 5669 – 5675.

[4] E. Gavi, L. Rivautella, D. L. Marchisio, M. Vanni, A. A. Barresi, G. Baldi. CFD Modelling of Nano-Particle Precipitation in Confined Impinging Jet Reactors. Chemical Engineering Research and Design 85 (2007), 735 – 744.

[5] E. Gavi, D. L. Marchisio, A. A. Barresi, M. G. Olsen, R. O. Fox. Turbulent Precipitation in Micromixers: CFD Simulation and Flow Field Validation. Chemical Engineering Research and Design 88 (2010), 1182 – 1193.

[6] H. C. Schwarzer, W. Peukert. Combined Experimental/Numerical Study on the Precipitation of Nanoparticles. AICHE Journal 50 (2004), 3234–3247.

[7] J. Baldyga, W. Orciuch. Barium Sulphate Precipitation in a Pipe – an Experimental Study and CFD Modelling. Chemical Engineering Science 56 (2001), 2435 – 2444.

[8] A. S. Myerson. Handbook of Industrial Crystallisation. 2nd Edition. Butterworth-Heinemann, Oxford (2002).

[9] D. L. Marchisio, A. A. Barresi. CFD Simulation of Mixing and Reaction: the Relevance of the Micro-Mixing Model. Chemical Engineering Science 58 (2003), 3579 – 3587.

[10] H. C. Schwarzer, F. Schwertfirm, M. Manhart, H.-J. Schmid, W. Peukert. Predictive Simulation of Nanoparticle Precipitation based on the Population Balance Equation. Chemical Engineering Science 61 (2006), 167 – 181.

[11] A. Mersmann, K. Bartosch, B. Braun, A. Eble, C. Heyer. Approaches to the Predictive Estimation of Crystallisation Kinetics. Chemie Ingenieur Technik 72 (2000), 17–30.

[12] A. Mersmann, B. Braun, M. Löffelmann. Prediction of Crystallization Coefficients of the Population Balance. Chemical Engineering Science 57 (2002), 4267 – 4275.

[13] A. Mersmann. Calculation of Interfacial Tensions. Journal of Crystal Growth 102 (1990), 841 – 847.

[14] M. Ståhl, Å. C. Rasmuson. Towards Predictive Simulation of Single Feed Semibatch Reaction Crystallization. Chemical Engineering Science 64 (2009), 1559 – 1576.

[15] M. Smoluchowski. Versuch einer mathematischen Theorie der Koagulationskinetik kolloider Lösungen. Zeitschrift für physikalische Chemie (1917), 129–168.

[16] S. Rollié. Heteroaggregation Processes in Colloidal Particle and Cell Systems. PhD Thesis, Fakultät für Verfahrens- und Systemtechnik der Otto-von-Guericke-Universität Magdeburg (2010).

[17] H. C. Schwarzer, W. Peukert. Prediction of Aggregation Kinetics based on Surface Properties of Nanoparticles. Chemical Engineering Science 60 (2005), 11 – 25.

[18] D. L. Marchisio, R. D. Vigil, R. O. Fox. Quadrature Method of Moments for Aggregation-Breakage Processes. Journal of Colloid and Interface Science 258 (2003), 322 – 334.

[19] D. L. Marchisio. Large Eddy Simulation of Mixing and Reaction in a Confined Impinging Jets Reactor. Computers & Chemical Engineering 33 (2009), 408 – 420.

[20] S. Radl, D. Suzzi, J. G. Khinast. Fast Reactions in Bubbly Flows: Film Model and Micro-Mixing Effects. Industrial & Engineering Chemistry Research, in Press.

[21] S. Rigopoulos. Population Balance Modelling of Polydispersed Particles in Reactive Flows. Progress in Energy and Combustion Science 36 (2010), 412 – 443.

[22] J. Baldyga, W. Podgórska, R. Pohorecki. Mixing-Precipitation Model with Application to Double Feed Semibatch Precipitation. Chemical Engineering Science 50 (1995), 1281 – 1300.

[23] Z. Ulbert, B. G. Lakatos. Modelling and Simulation of Crystallisers under Non-Perfect Micromixing Conditions. Chemical Engineering Science 60 (2005), 3525 – 3536.

[24] B. G. Lakatos. Population Balance Model for Mixing in Continuous Flow Systems. Chemical Engineering Science 63 (2008), 404 – 423.

[25] J. Baldyga, L. Makowski, W. Orciuch. Interaction between Mixing, Chemical Reactions, and Precipitation. Industrial & Engineering Chemistry Research 14 (2005), 5342–5352.

[26] R. Zauner, A. G. Jones. Mixing Effects on Product Particle Characteristics from Semi-Batch Crystal Precipitation. Chemical Engineering Research and Design 78 (2000), 894 – 902.

[27] J. Baldyga, J. R. Bourne, S. J. Hearn. Interaction between Chemical Reactions and Mixing on various Scales. Chemical Engineering Science 52 (1997), 457 – 466.

[28] N. L. Allinger, M. P. Cava, D. C. de Jongh, C. R. Johnson, N. A. Lebel, C. L. Stevens. Organische Chemie. Walter de Gruyter, Berlin (1980).

[29] K. P. C. Vollhardt, N. E. Schore. Organische Chemie. Wiley-VCH, Weinheim (2005).

[30] T. Ando, M. Yamasaki, K. Suzuki. Protamines. Springer-Verlag, New York (1973).

[31] D. Lochmann, J. Weyermann, C. Georgens, R. Prassl, A. Zimmer. Albumin-Protamine-Oligonucleotide Nanoparticles as a new Antisense Delivery System. Part 1: Physicochemical Characterization. European Journal of Pharmaceutics and Biopharmaceutics 59 (2005), 419 – 429.

[32] R. H. Perry, D. W. Green, J. O. Maloney. Perry's Chemical Engineers' Handbook. McGraw-Hill Companies, Inc., New York (1999).

[33] P. W. Atkins. Physical Chemistry. Oxford University Press, Oxford (1998).

[34] B. K. Johnson, R. K. Prud'homme. Chemical Processing and Micromixing in Confined Impinging Jets. AIChE Journal 49 (2003), 2264–2282.

[35] J. Visser. On Hamaker Constants: A Comparison between Hamaker Constants and Lifshitz-Van der Waals Constants. Advances in Colloid and Interface Science 3 (1972), 331 – 363.

[36] D. Ramkrishna. Population Balances. Academic Press, London (2000).

[37] J. D. Ward and C. Yu. Population Balance Modeling in Simulink: PCSS. Computers & Chemical Engineering 32 (2008), 2233 – 2242.

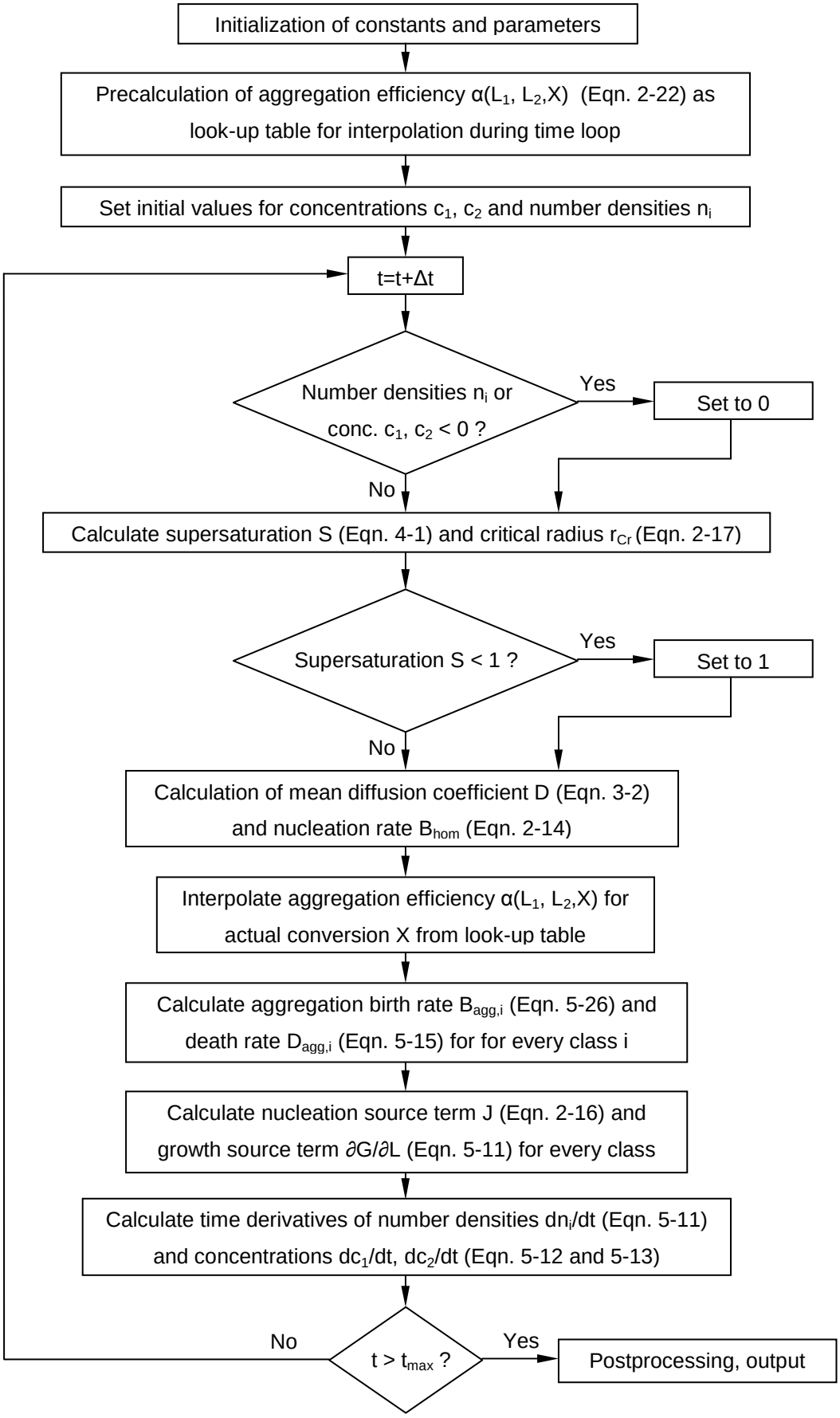
[38] C. Petschacher. Email "AW: Zetasizer + Info über Verteilungen" (2010-09-07).

[39] Malvern Instruments Ltd. Technical note MRK1357-01 (2009).

[40] I. B. Celik, U. Ghia, P. J. Roache, C. J. Freitas, H. Coleman, P. E. Raad. Procedure for Estimation and Reporting of Uncertainty due to Discretization in CFD Applications. Journal of Fluids Engineering 130 (2008), 078001.

[41] A. J. Alvarez, A. S. Myerson. Continuous Plug Flow Crystallization of Pharmaceutical Compounds. Crystal Growth & Design 10 (2010), 2219–2228.

Appendix A: Structogram of the PBE Implementation



Appendix B: Code (MATLAB R2008a)

I. Calculation of the Molecular Charge Numbers

MoleculeChargeNumbers.m

```
%% Molecule Charge Numbers
% (c) by Andreas Eitzlmayr

clear


%% Input:

% pKs-Values and numbers of acidic/basic groups per molecule:
    % Polyacrylic acid:
pKs1=1.9;  % Cystein-COOH
n1=3;
pKs2=8.4;  % Cystein-SH
n2=3;
pKs3=4.26; % Acrylic acid-COOH
n3=51;
    % Protamine:
pKs4=12.1; % Arginine-NH
n4=22;

% Ionic product of water:
Kw=10^-14;  % [molÂ²/lÂ²]

% Initial mass concentrations before mixing:
cm10=0.2;     %[g/l] Polyacrylic acid
cm20=0.6;     %[g/l] Protamine

% Mixing volume ratio V1/V2:
VolRatio=1;

% Molecular weight:
M1=5400; %[g/mol] Polyacrylic acid
M2=4300; %[g/mol] Protamine

% Graphic format:
figFontSize=18;
labelFontSize=18;
lineWidth=2;

%% Calculation:

% Initial molar concentrations:
c10ini=cm10/M1;
c20ini=cm20/M2;

% Molar concentrations in the mixture:
c10=c10ini*VolRatio/(1+VolRatio);
c20=c20ini*1/(1+VolRatio);

% pH-Range:
```

```
pH=linspace(0,14,14001);

% H3O+ Concentration:
cH3O=10.^-pH;

% Equilibrium constants:
K1=10^-pKs1;
K2=10^-pKs2;
K3=10^-pKs3;
K4=10^-pKs4;

% Conversion of Polyacrylic acid:
X=linspace(0,1,6);
m=size(X,2);

n=size(pH,2);
F=zeros(m,n);
pH_F0=zeros(1,m);

% Initial estimation of charge numbers (completely dissociated):
zPAAmean=n1+n2+n3;
zPROTmean=n4;

zPAAmeanOLD=0;
zPROTmeanOLD=0;

ItCount=0;

while (zPAAmean-zPAAmeanOLD)^2 + (zPROTmean-zPROTmeanOLD)^2 > 1e-6

    ItCount=ItCount+1;

    % Actual concentrations:
    c1=c10*(1-X);
    c2=c20-c10*X*zPAAmean/zPROTmean;

    zPAAmeanOLD=zPAAmean;
    zPROTmeanOLD=zPROTmean;

    zPAAmean=0;
    zPROTmean=0;

    for j=1:m
        for i=1:n
            F(j,i)=n1*K1*c1(j)/(cH3O(i)+K1) + n2*K2*c1(j)/ ...
                (cH3O(i)+K2) + n3*K3*c1(j)/(cH3O(i)+K3) + ...
                Kw/cH3O(i) - cH3O(i) - n4*c2(j)*cH3O(i)/(cH3O(i)+K4);
        end
        % Look for zero point of F (= Solution for pH):
        [F0,Idx(j)]=min(abs(F(j,:)));
        pH_F0(j)=pH(Idx(j));

        % Evaluate molecule charge numbers:
        zPAA(j)=n1*K1/(cH3O(Idx(j))+K1)+n2*K2/(cH3O(Idx(j))+K2) + ...
            n3*K3/(cH3O(Idx(j))+K3);
        zPROT(j)=n4*cH3O(Idx(j))/(cH3O(Idx(j))+K4);

        zPAAmean=zPAAmean+zPAA(j);
        zPROTmean=zPROTmean+zPROT(j);
    end
```

```
    % Mean molecule charge numbers:

    zPAAmean=zPAAmean/size(zPAA,2);
    zPROTmean=zPROTmean/size(zPROT,2);

end


% Evaluate number of charged groups at different pH-Values:
% (independent of conversion!!)

% for Cystein-COOH:

p1=zeros(1,n); % Amount of dissociated Cystein-COOH per Molecule

for i=1:n
    p1(i)=K1/(cH3O(i)+K1);
end


% for Cystein-SH:

p2=zeros(1,n); % Amount of dissociated Cystein-SH per Molecule

for i=1:n
    p2(i)=K2/(cH3O(i)+K2);
end


% Evaluate Molecule charge numbers at different pH-Values and conversions:

% for Polyacrylic acid-COOH:

p3=zeros(1,n); % Amount of dissociatedPolyacrylic acid-COOH Groups per
Molecule

for i=1:n
    p3(i)=K3/(cH3O(i)+K3);
end


% Evaluate Molecule charge numbers at different pH-Values and conversions:

% for Arginine-NH:

p4=zeros(1,n); % Amount of dissociated Arginine-NH Groups per Molecule

for i=1:n
    p4(i)=cH3O(i)/(cH3O(i)+K4);
end


% Evaluate Charge numbers per molecule for conversion 0 and 1:

% Cystein-COOH:

z10=p1(Idx(1))*n1;
z11=p1(Idx(6))*n1;

% Cystein-SH:
```

```
z20=p2(Idx(1))*n2;
z21=p2(Idx(6))*n2;

% PAA-COOH:

z30=p3(Idx(1))*n3;
z31=p3(Idx(6))*n3;

% Arginine-NH:

z40=p4(Idx(1))*n4;
z41=p4(Idx(6))*n4;

%% Output:

% Plot Function F:

plot(pH,F(1,:),'-b','LineWidth',2)
hold on
plot(pH,F(2,:),'-b','LineWidth',1)
plot(pH,F(3,:),'--r','LineWidth',2)
plot(pH,F(4,:),'--r','LineWidth',1)
plot(pH,F(5,:),'-.k','LineWidth',2)
plot(pH,F(6,:),'-.k','LineWidth',1)
plot([min(pH) max(pH)],[0 0],':k','LineWidth',1)
hold off

% Graphic format:
set(gca,'FontSize',figFontSize);
xlabel('pH');
ylabel('Function F [mol/l]');

%legend(['X = ',num2str(X(1))], ...
%    ['X = ',num2str(X(2))], ...
%    ['X = ',num2str(X(3))], ...
%    ['X = ',num2str(X(4))], ...
%    ['X = ',num2str(X(5))], ...
%    ['X = ',num2str(X(6))]);

axis([min(pH) max(pH) min(min(F)) max(max(F))])

% Save graphic:
fileSaveName='MoleculeChargeNumF.eps'
set(gcf,'PaperpositionMode','auto')
print(gcf,'-depsc','-r250',fileSaveName)

disp(['Saved as: ', fileSaveName])

% Plot zoomed graph of F:

axis([10.5 10.7 -1e-4 1e-4])
legend(['X = ',num2str(X(1))], ...
    ['X = ',num2str(X(2))], ...
    ['X = ',num2str(X(3))], ...
    ['X = ',num2str(X(4))], ...
    ['X = ',num2str(X(5))], ...
    ['X = ',num2str(X(6))],'Location','EastOutside');

% Save graphic:
fileSaveName='MoleculeChargeNumFzoom.eps'
set(gcf,'PaperpositionMode','auto')
```

```
print(gcf,'-depsc','-r250',fileSaveName)

disp(['Saved as: ', fileSaveName])


% Plot Amount of dissociated Groups per Molecule p1:


plot(pH,p1,'-b','LineWidth',2)
hold on
plot(pH,p3,'--r','LineWidth',2)
plot(pH,p2,'-.m','LineWidth',2)
plot(pH,p4,':k','LineWidth',2)
plot([pH_F0(1) pH_F0(1)],[0 1],'-k','LineWidth',0.5)
plot([pH_F0(6) pH_F0(6)],[0 1],'-k','LineWidth',0.5)
hold off

legend('Cysteine carboxy','Acrylic acid carboxy','Cysteine
sulfide','Guanidinium', ...
    'Location','EastOutside')
Pos = get(gcf,'Position');
set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.5 Pos(4)]);

% Graphic format:
set(gca,'FontSize',figFontSize);
xlabel('pH');
ylabel('Charged amount');

axis([min(pH) max(pH) min(min(p1)) max(max(p1))])

% Save graphic:
fileSaveName='MoleculeChargeNumGroups.eps'
set(gcf,'PaperpositionMode','auto')
print(gcf,'-depsc','-r250',fileSaveName)

disp(['Saved as: ', fileSaveName])

%% Text Output:

disp(['pH at Conversion X = ',num2str(X(1)),': ',num2str(pH_F0(1))])
disp(['pH at Conversion X = ',num2str(X(2)),': ',num2str(pH_F0(2))])
disp(['pH at Conversion X = ',num2str(X(3)),': ',num2str(pH_F0(3))])
disp(['pH at Conversion X = ',num2str(X(4)),': ',num2str(pH_F0(4))])
disp(['pH at Conversion X = ',num2str(X(5)),': ',num2str(pH_F0(5))])
disp(['pH at Conversion X = ',num2str(X(6)),': ',num2str(pH_F0(6))])
disp(['Charged Cystein-carboxy per molecule: ',num2str(z10),' - ', ...
    num2str(z11)])
disp(['Charged Cystein-sulfide per molecule: ',num2str(z20),' - ', ...
    num2str(z21)])
disp(['Charged PAA-carboxy per molecule: ',num2str(z30),' - ', ...
    num2str(z31)])
disp(['Charged Guanidinium per molecule: ',num2str(z40),' - ', ...
    num2str(z41)])

disp(['Total charge number PAA (negative): ',num2str(z10+z20+z30), ...
    ' - ',num2str(z11+z21+z31),', Mean: ', ...
    num2str((z10+z20+z30+z11+z21+z31)/2)])
disp(['Total charge number Protamine (positive): ',num2str(z40), ...
    ' - ',num2str(z41),', Mean: ',num2str((z40+z41)/2)])

disp(['Iteration Counter = ',num2str(ItCount)])
```

II. Nucleation Parameter Study

NucleationRate.m

```
%% Nucleation Rate (classical nucleation theory, Mersmann 2000)

function B=NucleationRate(D,c,cequ,sig,T,Vm,nd)


global NA
global k


% Nucleation rate:
B = 1.5*D*(c*NA*1000)^(7/3)*sqrt(sig/(k*T))*Vm * ...
    exp(-16*pi*sig^3*Vm^2/(3*(k*T)^3*(nd*log(c/cequ))^2));
```

Nucleation.m

```
%% Nucleation
% (c) by Andreas Eitzlmayr

clear

global z1
global z2
global cequ
global D1
global D2
global rh1
global rh2
global D
global cS1
global cS2
global c
global S

global k
global NA
global K
global c10
global c20
global T
global Vm
global sig
global cS
global nd
global Spezies

% Select Calculation:

ID1=3;

% ID1 =
% 0 ... Investigate Nucleation Rate
% 1 ... Calculate free energy vs. cluster size

Spezies=0;

% 0 ... Polyacrylic acid + Protamine
% 1 ... Bariumsulfate

%% Input:

% Constants:

k=1.380650424e-23;      % [J/K] Boltzmann's constant
NA=6.0221417930e23;       % [1/mol] Avogadro's constant

K=0.414;   % Interfacial energy constant
           % according to Mersmann 2000 between 0.310 and 0.414
Kad=0;     % free variable for adjusted K


% Spezies specific parameters:
switch Spezies
    case 0 % Polyacrylic acid + Protamine
```

```
        name1='Polyacrylic acid cysteine';
        name2='Protamine';
        M1=5400;                % [g/mol] Molar masses
        M2=4300;
        z1=-56.98;                 % [] molecular charge numbers
        z2=21.27;
        rh1=1.40;           % [nm] hydrodynamic radius
        rh2=1.35;           % (corresponding to diffusion coefficient)
        rhoS=1400;              % [kg/m³] Solid density
        etaW=0.001;             % [Pa s] dynamic viscosity

        cequ=1e-10;             % [mol/l] Equilibium concentration

        B_Desired=1e17;     % [1/m³s] Desired nucleation rate to adjust K

        % Initial conditions:
        cm10=0.2;     %[g/l] Polyacrylic acid
        cm20=0.6;     %[g/l] Protamine

        % Mixing volume ratio V1/V2:
        VolRatio=1;
    case 1 % Bariumsulfate

        name1='Barium';
        name2='Sulfate';
        M1=137.3;               % [g/mol] Molar masses
        M2=96.1;
        z1=2;                   % [] molecular charge numbers
        z2=-2;
        rh1=0.44482;            % [nm] hydrodynamic radius
        rh2=0.44482;            % (corresponding to diffusion coefficient)
        rhoS=4500;              % [kg/m³] Solid density
        etaW=0.001;             % [Pa s] dynamic viscosity

        cequ=sqrt(1.01e-4)*1e-3;  % [mol/l] Equilibium concentration

        B_Desired=1.18e11;     % [1/m³s] Desired nucleation rate to adjust
K

        % Initial conditions:
        cm10=0.2746;            %[g/l] Polyacrylic acid
        cm20=0.1922;            %[g/l] Protamine
        % Mixing volume ratio V1/V2:
        VolRatio=1;

end


% Integration parameters:
timeStep=2; % [s]
tmax=480;      % [s]

% Conditions:
T=295.15;                       % [K] Temperature

% Graphic format:
figFontSize=18;
labelFontSize=18;
lineWidth=2;
```

```
%% Calculation:

%Diffusion coefficients [m²/s]:
D1=k*T/(6*pi*etaW*rh1*1e-9);
D2=k*T/(6*pi*etaW*rh2*1e-9);

% Initial molar concentrations [mol/l]:
c10ini=cm10/M1;
c20ini=cm20/M2;

% Molar concentrations in the mixture:
c10=c10ini*VolRatio/(1+VolRatio);
c20=c20ini*1/(1+VolRatio);

% Solid concentrations [mol/l]:
% (due to electrical neutrality)
cS1=z2*rhoS/(z2*M1-z1*M2); % Polyacrylic acid (1)
cS2=-cS1*z1/z2;            % Protamine (2)
nd=2;%1-z1/z2;                % Dissociation number
cS=cS1+cS2;           % Total

% Mean solid molecular volume:
Vm=1/(1000*NA*cS);    %[m³]

% Interfacial energy:
sig=K*k*T*(1000*cS*NA)^(2/3)*log(cS/cequ); %[J/m²]

switch ID1
    case 0
        % Investigate Nucleation Rate:

        % Initial Supersaturation:
        c=sqrt(c10*c20);
        S=c/cequ;

        % Initial diffusion coefficient:
        D=(D1*c10+D2*c20)/(c10+c20);

        NucPlots
    case 1
        % Calculate Free energy vs. cluster size
        n=51;                % number of points
        r=linspace(0,5,n)*1e-9;  % [m] cluster sizes1

        Gsurf=zeros(1,n);
        Gvol=zeros(1,n);
        Gtot=zeros(1,n);

        S=1000; % Supersaturation
        DgSL=nd*k*T*log(S)/Vm;  % [J/m³] volume specific phase transf.
energy
        rc=2*sig/DgSL;   % [m] critical radius

        for i=1:n
            Gsurf(i)=4*pi*r(i)^2*sig;     % Surface free energy [J]
            Gvol(i)=-4/3*pi*r(i)^3*DgSL;   % Volume free energy [J]
            Gtot(i)=Gsurf(i)+Gvol(i);     % Total free energy [J]
        end

        % Print Results:
        r=r*1e9;
```

```
        rc=rc*1e9;
        plot(r,Gsurf,'-.r','LineWidth',lineWidth/2);
        hold on
        plot(r,Gtot,'-b','LineWidth',lineWidth);
        plot(r,Gvol,':k','LineWidth',lineWidth/2);
        plot([rc rc],[-1e-18 1e-18],'--k','LineWidth',lineWidth/2);
        hold off;

        % Graphic format:
        set(gca,'FontSize',labelFontSize);
        set(gca,'FontSize',figFontSize);
        xlabel('Cluster radius r [nm]');
        ylabel('Free energy \DeltaG [J]');
        axis([0 4 -1e-18 1e-18]);

        Pos=[560 530 560 420];
        set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.5 Pos(4)]);

        legend('Surface energy','Total energy', ...
            'Phase transf. energy','Critical radius','Location', ...
            'EastOutside');

        % Save graphic:
        fileSaveName='FreeEnergyClusterSize.eps';
        set(gcf,'PaperpositionMode','auto')
        print(gcf,'-depsc','-r250',fileSaveName)

        disp(['Saved as: ', fileSaveName])

    otherwise
end

%% Output:

% Text:

disp('NUCLEATION')
disp('****************************************************************')
disp(' ')

disp('INPUT:')

disp([name1,' (1):'])
disp(['Molar mass:                        ', num2str(M1), ' g/mol'])
disp(['Electrical charge number:          ', num2str(z1)])
disp(['hydrodynamic radius:               ', num2str(rh1), ' nm'])
disp(['Initial mass concentration:        ', num2str(cm10), ' g/l'])
disp(' ')

disp([name2,' (2):'])
disp(['Molar mass:                        ', num2str(M2), ' g/mol'])
disp(['Electrical charge number:          ', num2str(z2)])
disp(['hydrodynamic radius:               ', num2str(rh2), ' nm'])
disp(['Initial mass concentration:        ', num2str(cm20), ' g/l'])
disp(' ')

disp(['Interfacial energy constant:       ', num2str(K)])
disp(['Dissociation number:               ', num2str(nd)])
disp(['Solid density:                     ', num2str(rhoS), ' kg/m³'])
disp(['Dynamic fluid viscosity:           ', num2str(etaW), ' Pa s'])
disp(['Solubility product:                ', num2str(cequ^2), ...
```

```
    ' (mol/l)²'])
disp(['Temperature:                    ', num2str(T), ' K'])
disp(' ')
disp(' ')

disp('OUTPUT:')

disp(['Diffusion coefficient 1:        ', num2str(D1), ' m²/s'])
disp(['Diffusion coefficient 2:        ', num2str(D2), ' m²/s'])
disp(['Initial concentration 1:        ', num2str(c10), ' mol/l'])
disp(['Initial concentration 2:        ', num2str(c20), ' mol/l'])
disp(['Solid concentration 1:          ', num2str(cS1), ' mol/l'])
disp(['Solid concentration 2:          ', num2str(cS2), ' mol/l'])
disp(['Total solid concentration:      ', num2str(cS), ' mol/l'])
disp(['Mean solid molecular volume:    ', num2str(Vm), ' m³'])
disp(['Equilibrium concentration:      ', num2str(cequ), ' mol/l'])
disp(['Interfacial energy:             ', num2str(sig), ' J/m²'])

if Kad==0
else
    disp(['Adjusted interf. energy const. K: ', num2str(Kad)])
end
```

NucPlots.m

```
%% Nucleation Plots
% (c) by Andreas Eitzlmayr

global k
global NA
global c10
global c20

global c
global S
global D
global cequ
global K

global T
global Vm
global sig
global cS
global nd
global rh1
global rh2
global Spezies

% Graphic format:
figFontSize=18;
lineWidth=2;



%% Nucelation vs. equilibrium conc./K-Sigma

fileSaveName='FigNucEquK.eps';

n=100;  % Number of different equilibrium concentrations CEqu
m=3;    % Number of different Interfacial energy constants Ksig

CEqu=logspace(-12,-6,n);      % equilibrium concentrations
Ksig=linspace(0.310,0.414,m); % Interfacial energy constants

Sigma=zeros(m,n);   % Interfacial energy [J/m²]
B=zeros(m,n);       % Nucleation rate [1/m³s]
rc=zeros(m,n);      % Critical radius [nm]

for i=1:m   % Change Ksig
    for j=1:n      % change CEqu
       Sigma(i,j)=Ksig(i)*k*T*(1000*cS*NA)^(2/3)*log(cS/CEqu(j));
       B(i,j)=NucleationRate(D,c,CEqu(j),Sigma(i,j),T,Vm,nd);
       rc(i,j)=2*Sigma(i,j)*Vm/(nd*k*T*log(c/CEqu(j)))*1e9;
    end
end

% look for maximum nucleation rate:
[Bmax,I]=max(max(B));
CEquMax=CEqu(I);

% Print nucleation rate results:
loglog(CEqu,B(1,:),'-b',CEqu,B(2,:),'--r',CEqu,B(3,:),'-.m', ...
    'LineWidth',lineWidth);
```

```
% Graphic format:
legend(['K = ',num2str(Ksig(1))],['K = ',num2str(Ksig(2))],['K = ', ...
    num2str(Ksig(3))],'Location','SouthWest');
set(gca,'FontSize',figFontSize);
set(gca,'XTick',[1e-10 1e-9 1e-8 1e-7 1e-6 1e-5]);
xlabel('Equilibrium concentration c* [mol/l]');
ylabel('Nucleation rate B_{hom} [1/m^{3}s]');

% Save graphic:
set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)

% Print critical radius results:
fileSaveName='FigRCEquK.eps';
semilogx(CEqu,rc(3,:),'-b',CEqu,rc(2,:),'--r',CEqu,rc(1,:),'-.m', ...
    'LineWidth',lineWidth);

% Graphic format:
legend(['K = ',num2str(Ksig(3))],['K = ',num2str(Ksig(2))],['K = ', ...
    num2str(Ksig(1))],'Location','NorthWest');
set(gca,'FontSize',figFontSize);
set(gca,'XTick',[1e-10 1e-9 1e-8 1e-7 1e-6 1e-5]);
xlabel('Equilibrium concentration c* [mol/l]');
ylabel('Critical radius r_{cr} [nm]');

% Save graphic:
set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)

%Text Output:
disp(['Saved as: ', fileSaveName]);
disp(['Maximum Nucleationrate:           ', num2str(Bmax),' 1/m^{3}s'])
disp(['at equilibrium Concentration:     ', num2str(CEquMax),' mol/l'])


%% Nucleation and Critical nuclei radius vs. supersaturation:

fileSaveName='FigNucSup.eps';

m=100;              % Number of different Supersaturations

if Spezies==0;  % Polyacrylic acid & protamine
    Sup=logspace(1,8,m); % Supersaturations
elseif Spezies==1; % Bariumsulfate
    Sup=logspace(1,4,m); % Supersaturations
end

KSig1=[0.310 0.362 0.414]; % 3 different interfacial energies
Sigma1=KSig1*k*T*(1000*cS*NA)^(2/3)*log(cS/cequ);

if Spezies==1; % Bariumsulfate
    Sigma1=[0.1 0.12 0.14];   % [N/m]
end

n = size(Sigma1,2);

B=zeros(n,m);      % Nucleation rates [1/m³s]
rc=zeros(n,m);     % Critical radius [nm]

for i=1:n
```

```
    for j=1:m
        B(i,j)=NucleationRate(D,cequ*Sup(j),cequ,Sigma1(i),T,Vm,nd);
        rc(i,j)=2*Sigma1(i)*Vm/(nd*k*T*log(Sup(j)))*1e9;
    end
end

% Print nucleation rate results:
loglog(Sup,B(1,:),'-b',Sup,B(2,:),'--r',Sup,B(3,:),'-.m', ...
    'LineWidth',lineWidth);

% Graphic format:
axis([min(Sup) max(Sup) 1e0 1e32]);
set(gca,'FontSize',figFontSize);
xlabel('Supersaturation S');
ylabel('Nucleation rate B_{hom} [1/m^{3}s]');

if Spezies==0;  % Polyacrylic acid & protamine
    legend(['K = ',num2str(KSig1(1))],['K = ',num2str(KSig1(2))], ...
        ['K = ',num2str(KSig1(3))],'Location','SouthEast');
elseif Spezies==1; % Bariumsulfate
    legend(['\sigma = ',num2str(Sigma1(1)),' N/m'],['\sigma = ', ...
        num2str(Sigma1(2)),' N/m'],['\sigma = ',num2str(Sigma1(3)), ...
        ' N/m'],'Location','SouthEast');
end

% Save graphic:
set(gcf,'PaperpositionMode','auto')
print(gcf,'-depsc','-r250',fileSaveName)

disp(['Saved as: ', fileSaveName])


% Print critical radius results:
fileSaveName='FigRCSup.eps';
semilogx(Sup,rc(3,:),'-b',Sup,rc(2,:),'--r',Sup,rc(1,:),'-.m', ...
    'LineWidth',lineWidth);
hold on
semilogx([min(Sup) max(Sup)],[rh1 rh1],'--b',[min(Sup) max(Sup)], ...
    [rh2 rh2],'-.k','LineWidth',0.5);
hold off

% Graphic format:
set(gca,'FontSize',figFontSize);
axis([min(Sup) max(Sup) 0 10]);
xlabel('Supersaturation S');
ylabel('Critical radius r_{cr} [nm]');
legend(['K = ',num2str(KSig1(3))],['K = ',num2str(KSig1(2))], ...
    ['K = ',num2str(KSig1(1))],'r_{h} protamine', ...
    'r_{h} polyacrylic acid','Location','NorthEast');

% Save graphic:
set(gcf,'PaperpositionMode','auto')
print(gcf,'-depsc','-r250',fileSaveName)

disp(['Saved as: ', fileSaveName])


%% Nucleation rate vs. interfacial energy constant K

fileSaveName='FigNucIntE.eps';
```

```
n=100;                          % Number of different
Ksig=linspace(0.25,0.45,n);    % Interfacial energy constants

m=4;                           % Number of different supersaturations
Sup=[1e7 1e6 1e5 1e4];


B=zeros(m,n);                  % Nucleation Rates [1/m³s]
Sigma=zeros(1,n);              % Interfacial energies [J/m²s]

for j=1:m
    for i=1:n
        Sigma(i)=Ksig(i)*k*T*(1000*cS*NA)^(2/3)*log(cS/cequ);
        B(j,i)=NucleationRate(D,cequ*Sup(j),cequ,Sigma(i),T,Vm,nd);
    end
end

% Print Results:
semilogy(Ksig,B(1,:),'-b','LineWidth',lineWidth)
hold on
semilogy(Ksig,B(2,:),'--r','LineWidth',lineWidth)
semilogy(Ksig,B(3,:),'-.m','LineWidth',lineWidth)
semilogy(Ksig,B(4,:),':k','LineWidth',lineWidth)
hold off

% Graphic format:
set(gca,'FontSize',figFontSize);
xlabel('Interfacial energy constant K');
ylabel('Nucleation rate B_{hom} [1/m^{3}s]');
axis([min(Ksig),max(Ksig),10^0,10^30]);
legend(['S = 10^{',num2str(log10(Sup(1))),'}'], ...
    ['S = 10^{',num2str(log10(Sup(2))),'}'], ...
    ['S = 10^{',num2str(log10(Sup(3))),'}'], ...
    ['S = 10^{',num2str(log10(Sup(4))),'}'],'Location','SouthWest')

% Save graphic:
set(gcf,'PaperpositionMode','auto')
print(gcf,'-depsc','-r250',fileSaveName)

disp(['Saved as: ', fileSaveName])

%% Nucleation vs. Molecular volume (Solid concentration)

fileSaveName='FigNucv.eps';

n=100;                          % Number of different solid conc.
CS=logspace(-3,2,n);           % Solid concentrations [mol/]

m=3;                           % Number of different constants K
Ksig=[0.310 0.362 0.414];

V=zeros(1,n);                  % Molecular volumes [m³]
Sigma=zeros(m,n);              % Interfacial energies [J/m²]
B=zeros(m,n);                  % Nucleation Rates [1/m³s]

for j=1:m
    for i=1:n
        V(i)=1/(1000*NA*CS(i));
        Sigma(j,i)=Ksig(j)*k*T*(1000*CS(i)*NA)^(2/3)*log(CS(i)/cequ);
        B(j,i)=NucleationRate(D,c,cequ,Sigma(j,i),T,V(i),nd);
    end
```

```
end

% Print results:
loglog(V,B(1,:),'-b','LineWidth',lineWidth);
hold on
loglog(V,B(2,:),'--r','LineWidth',lineWidth);
loglog(V,B(3,:),'-.m','LineWidth',lineWidth);
hold off

% Graphic format:
axis([min(V),max(V),0.1*min(min(B)),10*max(max(B))]);
set(gca,'FontSize',figFontSize);
xlabel('Molecular volume V_{m} [m^{3}]');
ylabel('Nucleation rate B_{hom} [1/m^{3}s]');
legend(['K = ',num2str(Ksig(1))],['K = ',num2str(Ksig(2))], ...
    ['K = ',num2str(Ksig(3))],'Location','SouthEast');

% Save graphic:
set(gcf,'PaperpositionMode','auto')
print(gcf,'-depsc','-r250',fileSaveName)

disp(['Saved as: ', fileSaveName])


%% Nucleation vs. Temperature

fileSaveName='FigNucT.eps';

n=10;                           % Number of different Temperatures
Temp=linspace(273.15,323.15,n); % Temperatures [K]

Sigma=zeros(1,n);           % Interfacial energies [J/m²]
B=zeros(1,n);               % Nucleation Rates [1/m³s]

for i=1:n
    Sigma(i)=K*k*Temp(i)*(1000*cS*NA)^(2/3)*log(cS/cequ);
    B(i)=NucleationRate(D,c,cequ,Sigma(i),Temp(i),Vm,nd);
end

% Print results:
plot(Temp,B,'LineWidth',lineWidth);

% Graphic format:
set(gca,'FontSize',figFontSize);
xlabel('Temperature T [K]');
ylabel('Nucleation rate B_{hom} [1/m^{3}s]');
title(['cequ=',num2str(cequ),' mol/l, S=',num2str(S), ', K=', ...
    num2str(K)]);

% Save graphic:
set(gcf,'PaperpositionMode','auto')
print(gcf,'-depsc','-r250',fileSaveName)

disp(['Saved as: ', fileSaveName])
```

III. Aggreation Parameter Study

Collisionkernel.m

```
%% Collision kernel for brownian motion (Smoluchowski 1917)

function Beta=Collisionkernel(z)

% z = L1/L2 ... size ratio of 2 colliding particles

global k
global T
global etaW

Beta=2*k*T/(3*etaW)*(2+z+1/z); % [m³/s]
```

HydrodynCorr.m

```
%% Hydrodynamic correction

function BHyd=HydrodynCorr(a,L1,L2)

% a ... surface to surface distance
% L1, L2 ... Particle sizes

z=a*(L1+L2)/(L1*L2);

BHyd=(6*z.^2+13*z+2)./(6*z.^2+4*z);
```

HamakerPotential.m

```
%% Hamaker Potential

function PhiVDW=HamakerPotential(a,r1,r2)

% a ... Surface to surface distance [m]
% r1, r2 ... particle radii [m]

global A

x=a/r1;   % dimensionless surface to surface distance

z=r2/r1;  % particle size ratio

PhiVDW = -A/6 * ( 2*z./(x.^2+2*x*(1+z)) + 2*z./(x.^2+2*x*(1+z)+4*z) + ...
    log( (x.^2+2*x*(1+z)) ./ (x.^2+2*x*(1+z)+4*z) ) );
```

GouyChapman.m

```
%% Gouy-Chapman electrostatic potential

function PhiEl=GouyChapman(a,r1,r2,c1,c2)

% a ... Surface to surface distance [m]
% r1, r2 ... particel radii [m]
% c1, c2 ... component concentrations [mol/l]

global NA
global k
global T
global e
global c2pzc
global eps0
global epsr
global z1
global z2

I=0.5*(c1*z1^2+c2*z2^2);  % ionic strength [mol/l]

kappa=sqrt(2*e^2*NA*I/(eps0*epsr*k*T));  % reciprocal Debye length [m]

PsiP=log(c2/c2pzc);  % surface potential; factor kT/ze is canceled out!

PhiEl = 128*pi*1000*(c1+c2)*NA*k*T/kappa^2 * (tanh(PsiP/4))^2 * ...
    r1*r2./(r1+r2+a) .* exp(-kappa*a); %[J]
% Factor 1000 because c1 and c2 are in mol/l !
```

StabilityIntegrand.m

```
%% Stability ratio Integrand

function StIn=StabilityIntegrand(a)

% a ... Surface to surface distance [m]
% L1_SI, L2_SI ... particle sizes [m]
% c1_SI, c2_SI ... component concentrations [mol/l]

global k
global T

global L1_SI
global L2_SI
global c1_SI
global c2_SI
global ScaleFac

r1=L1_SI/2;
r2=L2_SI/2;
c1=c1_SI;
c2=c2_SI;


StIn = HydrodynCorr(a,L1_SI,L2_SI) .* exp((HamakerPotential(a,r1,r2) + ...
    GouyChapman(a,r1,r2,c1,c2)) / (k*T)) ./ (r1+r2+a).^2;
StIn = StIn/ScaleFac;
```

AggregationEfficiency.m

```
%% Aggregation efficiency (dimensionless)

function Alpha=AggEfficiency(L1,L2,c1,c2)

global L1_SI
global L2_SI
global c1_SI
global c2_SI
global ScaleFac

L1_SI=L1;
L2_SI=L2;
c1_SI=c1;
c2_SI=c2;

%Evaluate scaling factor (to avoid numerical problems with integration):

ScaleFac=1;   % Scaling factor for Stability integrand

n=20;
a=logspace(-11,-6,n);
StIn=zeros(1,n);

for i=1:n
    StIn(i)=StabilityIntegrand(a(i));
end

ScaleFac=max(StIn);

if isinf(ScaleFac)==1 || ScaleFac>1e100
    % Integral is infinity
    Integral=0;
    Alpha=0;
else

    % Integration:

    LowerB=1e-20;

    Integral=quad(@StabilityIntegrand,LowerB,1e-10);
    Integral=Integral+quad(@StabilityIntegrand,1e-10,1e-9);
    Integral=Integral+quad(@StabilityIntegrand,1e-9,1e-8);

    OldIntegral=0;
    Bound=1e-8;

    while (Integral-OldIntegral)/Integral > 0.01
        OldIntegral=Integral;
        Integral=Integral+quad(@StabilityIntegrand,Bound,10*Bound);
        Bound=Bound*10;
    end

    W=Integral*(L1+L2)/2*ScaleFac;

    % Complete kernel:
    Alpha = 1 / W;
end
end
```

PlotCollisionkernel.m

```
%% Plot Collisionkernel and Hydrodynamic Correction
% (c) by Andreas Eitzlmayr

clear

global z1
global z2
global D1
global D2

global k
global NA
global e
global c10
global c20
global T
global etaW
global A
global c2pzc
global eps0
global epsr

% variables *_SI are arguments for function
% StabilityIntegrand, they must be defined global, because
% for integration of StablityIntegrand only 1 argument is
% allowed


% Select Calculation:

ID1=1;

% ID1 =
% 0 ... Collision kernel depending on L1 and L2
% 1 ... Collision kernel depending on ration L1/L2
% 2 ... Hydrodynamic correction


Spezies=0;

% 0 ... Polyacrylic acid + Protamine
% 1 ... Bariumsulfate

%% Input:

% Constants:

k=1.381e-23;      % [J/K] Boltzmann's constant
NA=6.022e23;     % [1/mol] Avogadro's constant
e=1.602e-19;    % [As] elementary charge
eps0=8.854e-12;         % [As/Vm] electric constant
epsr=80;                 % [] relative permittivity

K=0.414;   % Interfacial energy constant
           % according to Mersmann 2000 between 0.310 and 0.414
Kad=0;     % free variable for adjusted K

A=1e-20;    % Hamaker constant [J]
```

```
% Spezies specific parameters:
switch Spezies
    case 0 % Polyacrylic acid + Protamine

        name1='Polyacrylic acid cysteine';
        name2='Protamine';
        M1=5400;                % [g/mol] Molar masses
        M2=4300;
        z1=-56.99;                 % [] molecular charge numbers
        z2=20.72;
        rh1=1.40;           % [nm] hydrodynamic radius
        rh2=1.35;           % (corresponding to diffusion coefficient)
        rhoS=1400;              % [kg/m³] Solid density
        etaW=0.001;             % [Pa s] dynamic viscosity

        % Initial conditions:
        cm10=0.2;     %[g/l] Polyacrylic acid
        cm20=0.6;     %[g/l] Protamine
        % Mixing volume ratio V1/V2:
        VolRatio=1;


        c2pzc=1e-4;       %[mol/l] concentr. of Potential determining ions

    case 1 % Bariumsulfate

        name1='Barium';
        name2='Sulfate';
        M1=137.3;               % [g/mol] Molar masses
        M2=96.1;
        z1=2;                   % [] molecular charge numbers
        z2=-2;
        rh1=0.44482;            % [nm] hydrodynamic radius
        rh2=0.44482;            % (corresponding to diffusion coefficient)
        rhoS=4500;              % [kg/m³] Solid density
        etaW=0.001;             % [Pa s] dynamic viscosity

        % Initial conditions:
        cm10=0.1373;            %[g/l] Polyacrylic acid
        cm20=0.0961;            %[g/l] Protamine

        % Mixing volume ratio V1/V2:
        VolRatio=1;
end


% Integration parameters:
timeStep=2; % [s]
tmax=480;      % [s]

% Conditions:
T=293.15;                       % [K] Temperature

% Graphic format:
figFontSize=18;
labelFontSize=14;
lineWidth=2;

%% Calculation:
```

```
% Diffusion coefficients [m²/s]:
D1=k*T/(6*pi*etaW*rh1*1e-9);
D2=k*T/(6*pi*etaW*rh2*1e-9);

% Initial molar concentrations [mol/l]:
c10ini=cm10/M1;
c20ini=cm20/M2;


% Molar concentrations in the mixture:
c10=c10ini*VolRatio/(1+VolRatio);
c20=c20ini*1/(1+VolRatio);

switch ID1
    case 0


        % Collision kernel (depending on L1 and L2):

        n=50; % number of points

        x=linspace(10,100,n); % particle sizes [nm]

        Beta=zeros(n,n);
        BetaDimless=Beta;

        for i=1:n
            for j=1:n
                % collision frequency for sizes x(i) and x(j):
                Beta(i,j)=Collisionkernel(x(i)/x(j));
                BetaDimless(i,j)=Beta(i,j)*3*etaW/(2*k*T);
            end
        end

    case 1
         % Collision kernel (depending on the ratio L1/L2):

         n=80;  % Number of points
         v=4;   % Maximum ratio L1/L2
         z=logspace(0,v,n);
         Beta=zeros(1,n);
         BetaDimless=Beta;

         for i=1:n
             Beta(i)=Collisionkernel(z(i));
             BetaDimless(i)=Beta(i)*3*etaW/(2*k*T);
         end

    case 2

        % Hydrodynamic correction:

        n=50;   % number of points BHyd(r/r*)

        r=logspace(-1,2,n); % surface to surface distance
        BHyd=zeros(1,n);    % hydrodynamic correction

        for i=1:n
            BHyd(i)=HydrodynCorr(r(i),2,2);
            % For L1=L2=2 is r*=1
```

```
        end

    otherwise
end

%% Grafic output

switch ID1
    case 0
        % Plot collision kernel (depending on L1 and L2)

        colormap(jet);
        surf(x,x,Beta);
        view([0,0,1]);
        colorbar('location','EastOutside');

        % Graphic format:
        set(gca,'FontSize',figFontSize);
        xlabel('Particle size L_{1} [nm]');
        ylabel('Particle size L_{2} [nm]');
        axis([min(x) max(x) min(x) max(x)]);
        title('Collision kernel \beta_{coll} [m^{3}/s]');

        % Save graphic:
        fileSaveName='Collisionkernel_Smol1.eps'
        set(gcf,'PaperpositionMode','auto')
        print(gcf,'-depsc','-r250',fileSaveName)

        disp(['Saved as: ', fileSaveName])

        surf(x,x,BetaDimless);
        view([0,0,1]);
        colorbar('location','EastOutside');

        % Graphic format:
        set(gca,'FontSize',figFontSize);
        xlabel('Particle size L_{1} [nm]');
        ylabel('Particle size L_{2} [nm]');
        axis([min(x) max(x) min(x) max(x)]);
        title('Dimensionless collision kernel \beta^{*}_{coll}');

        % Save graphic:
        fileSaveName='Collisionkernel_Smol1dimless.eps'
        set(gcf,'PaperpositionMode','auto')
        print(gcf,'-depsc','-r250',fileSaveName)

        disp(['Saved as: ', fileSaveName])

    case 1
        % Plot collision kernel (depending on ratio L1/L2)

        loglog(z,Beta,'LineWidth',lineWidth);

        % Graphic format:
        set(gca,'FontSize',figFontSize);
        xlabel('Particle size ratio L_{1}/L_{2}');
        ylabel('Collision kernel [m^{3}/s]');

        % Save graphic:
        fileSaveName='Collisionkernel_Smol2.eps'
        set(gcf,'PaperpositionMode','auto')
```

```
        print(gcf,'-depsc','-r250',fileSaveName)

        disp(['Saved as: ', fileSaveName])


        % Plot dimensionless collision kernel (depending on ratio L1/L2)

        loglog(z,BetaDimless,'LineWidth',lineWidth);

        % Graphic format:
        set(gca,'FontSize',figFontSize);
        xlabel('Particle size ratio L_{1}/L_{2}');
        ylabel('Dimensionless collision kernel');

        % Save graphic:
        fileSaveName='Collisionkernel_Smol2dimless.eps'
        set(gcf,'PaperpositionMode','auto')
        print(gcf,'-depsc','-r250',fileSaveName)

        disp(['Saved as: ', fileSaveName])

    case 2
        % Plot hydrodynamic correction
        semilogx(r,BHyd,'LineWidth',lineWidth);
        hold on
        semilogx([min(r) max(r)],[1 1],'--','LineWidth',lineWidth/2);
        hold off

        % Graphic format:
        set(gca,'FontSize',figFontSize);
        axis([min(r),max(r),0,max(BHyd)*1.1]);
        xlabel('Specific surface to surface distance r/r*');
        ylabel('Hydrodynamic correction');

        % Save graphic:
        fileSaveName='HydrodynCorr.eps'
        set(gcf,'PaperpositionMode','auto')
        print(gcf,'-depsc','-r250',fileSaveName)

        disp(['Saved as: ', fileSaveName])

    otherwise
end
```

PlotInteracPotentials.m

```
%% Plot Interaction Potentials
% (c) by Andreas Eitzlmayr

clear

global z1
global z2
global D1
global D2

global k
global NA
global e
global c10
global c20
global T
global etaW
global A
global c2pzc
global eps0
global epsr

% variables *_SI are arguments for function
% StabilityIntegrand, they must be defined global, because
% for integration of StablityIntegrand only 1 argument is
% allowed

% Select Calculation:

ID1=1;

% ID1 =
% 0 ... Hamaker potential
% 1 ... electrostatic potential + Total interaction potential
% 2 ... Surface potential


Spezies=0;

% 0 ... Polyacrylic acid + Protamine
% 1 ... Bariumsulfate

%% Input:

% Constants:

k=1.381e-23;      % [J/K] Boltzmann's constant
NA=6.022e23;     % [1/mol] Avogadro's constant
e=1.602e-19;    % [As] elementary charge
eps0=8.854e-12;         % [As/Vm] electric constant
epsr=80;                 % [] relative permittivity

K=0.414;   % Interfacial energy constant
           % according to Mersmann 2000 between 0.310 and 0.414
Kad=0;     % free variable for adjusted K

A=1e-20;    % Hamaker constant [J]
```

```
% Spezies specific parameters:
switch Spezies
    case 0 % Polyacrylic acid + Protamine

        name1='Polyacrylic acid cysteine';
        name2='Protamine';
        M1=5400;                % [g/mol] Molar masses
        M2=4300;
        z1=-56.99;                 % [] molecular charge numbers
        z2=20.72;
        rh1=1.40;           % [nm] hydrodynamic radius
        rh2=1.35;           % (corresponding to diffusion coefficient)
        rhoS=1400;              % [kg/m³] Solid density
        etaW=0.001;             % [Pa s] dynamic viscosity

        % Initial conditions:
        cm10=0.2;     %[g/l] Polyacrylic acid
        cm20=0.6;     %[g/l] Protamine
        c2pzc=1e-4;       %[mol/l] concentr. of Potential determining ions

        % Mixing volume ratio V1/V2:
        VolRatio=1;
    case 1 % Bariumsulfate

        name1='Barium';
        name2='Sulfate';
        M1=137.3;               % [g/mol] Molar masses
        M2=96.1;
        z1=2;                   % [] molecular charge numbers
        z2=-2;
        rh1=0.44482;            % [nm] hydrodynamic radius
        rh2=0.44482;            % (corresponding to diffusion coefficient)
        rhoS=4500;              % [kg/m³] Solid density
        etaW=0.001;             % [Pa s] dynamic viscosity

        % Initial conditions:
        cm10=0.1373;            %[g/l] Polyacrylic acid
        cm20=0.0961;            %[g/l] Protamine

        % Mixing volume ratio V1/V2:
        VolRatio=1;
end


% Integration parameters:
timeStep=2; % [s]
tmax=480;      % [s]

% Conditions:
T=293.15;                       % [K] Temperature

% Graphic format:
figFontSize=18;
labelFontSize=18;
lineWidth=2;

%% Calculation:

% Diffusion coefficients [m²/s]:
D1=k*T/(6*pi*etaW*rh1*1e-9);
D2=k*T/(6*pi*etaW*rh2*1e-9);
```

```
% Initial molar concentrations [mol/l]:
c10ini=cm10/M1;
c20ini=cm20/M2;


% Molar concentrations in the mixture:
c10=c10ini*VolRatio/(1+VolRatio);
c20=c20ini*1/(1+VolRatio);

switch ID1

    case 0

        % Hamaker potential (Van der Waals):

        n=40;   % number of points
        m=3;    % number of different size ratios;

        r=logspace(-3,-1,n); % different dimensionless distances (a/r1)
        z=logspace(-1,1,m); % different size ratios (r2/r1)

        PhiVDW=zeros(m,n);

        for j=1:m
            for i=1:n
                PhiVDW(j,i) = HamakerPotential(r(i),1,z(j));
            end
        end

    case 1
        % Gouy-Chapman electrostatic potential + total interaction pot.
        % for model I (nucleation + aggregation)

        n=50; % number of points

        a=logspace(-11,-7,n); % [m] different distances
        % [m] different combinations r1, r2:
        r=[80 80;30 80;30 30;5 80;5 30;5 5]*1e-9;
        X=[0 0.5 1];    % Reaction conversion (% consumed Polyacrylic acid)

        m=size(r,1); % number of different combinations of radii r1 and r2
        p=size(X,2); % number of different reaction conversions

        PhiEl=zeros(n,m,p);
        PhiTot=zeros(n,m,p);
        c1=zeros(1,p);
        c2=zeros(1,p);

        for l=1:p  % loop for different conversions
            c1(l)=c10*(1-X(l));
            c2(l)=c20-c10*X(l)*(-z1/z2);

            for j=1:m      % loop for different radii r2, r2
                for i=1:n  % loop for different distances a
                    PhiEl(i,j,l) = GouyChapman(a(i),r(j,1),r(j,2), ...
                        c1(l),c2(l));
                    PhiTot(i,j,l) =  PhiEl(i,j,l) + ...
                        HamakerPotential(a(i),r(j,1),r(j,2));
                end
            end
```

```
        end

     case 2
        %Surface potential

        n=20; % number of points
        cPDI=linspace(1e-5,1e-4,n);
        cPDIpzc=[1e-9 1e-8 1e-7 1e-6 1e-5 1e-4];
        m=size(cPDIpzc,2);

        PSI=zeros(n,m);

        for j=1:m % change cPDIpzc
            for i=1:n % change cPDI
                PSI(i,j)=k*T/(z2*e)*log(cPDI(i)/cPDIpzc(j));
            end
        end

    otherwise
end

%% Grafic output

switch ID1

    case 0
        % Plot Hamaker potential

        semilogx(r,PhiVDW(1,:),'-b','LineWidth',lineWidth);
        hold on
        semilogx(r,PhiVDW(2,:),'--r','LineWidth',lineWidth);
        semilogx(r,PhiVDW(3,:),'-.k','LineWidth',lineWidth);
        hold off


        % Graphic format:
        set(gca,'FontSize',figFontSize);
        xlabel('Dimensionless surface to surface distance \xi = a/r_{1}');
        ylabel('Hamaker potential \phi_{VdW} [J]');
        axis([min(r), max(r), -2e-18, 0.5e-18]);
        %title('Hamaker potential for two particles with size ratio \lambda
        %= r_{2}/r_{1} (r_{1}=const.)');

        legend(['\lambda = ',num2str(z(1))],['\lambda = ', ...
            num2str(z(2))],['\lambda = ',num2str(z(3))], ...
            'location','SouthEast')

        % Save graphic:
        fileSaveName='HamakerPotential.eps';
        set(gcf,'PaperpositionMode','auto')
        print(gcf,'-depsc','-r250',fileSaveName)

        disp(['Saved as: ', fileSaveName])

    case 1
        % Plot Gouy-Chapman electrostatic potential and total interac. pot.
        % for model I (nucleation + aggregation)

        a=a*1e9; % [m] -> [nm]
        L=r*2*1e9; % [m] -> [nm]
```

```
Xmin=0;
Xmax=100;
Ymin=-0.5e-18;
Ymax=3e-18;

% Plot Gouy-Chapman potential for different sizes

for l=1:p
    plot(a,PhiEl(:,1,l),'-r','LineWidth',lineWidth)
    hold on
    plot(a,PhiEl(:,2,l),'--b','LineWidth',lineWidth)
    plot(a,PhiEl(:,3,l),'-.b','LineWidth',lineWidth)
    plot(a,PhiEl(:,4,l),'-k','LineWidth',lineWidth/2)
    plot(a,PhiEl(:,5,l),'--k','LineWidth',lineWidth/2)
    plot(a,PhiEl(:,6,l),'-.k','LineWidth',lineWidth/2)
    hold off

    % Graphic format:
    set(gca,'FontSize',labelFontSize);
    set(gca,'FontSize',figFontSize);
    xlabel('Surface to surface distance a [nm]');
    ylabel('Electrostatic potential \phi_{el} [J]');
    axis([Xmin Xmax 0 Ymax]);

    legend([num2str(L(1,1)),' nm / ',num2str(L(1,2)),' nm' ],...
        [num2str(L(2,1)),' nm / ',num2str(L(2,2)),' nm' ], ...
        [num2str(L(3,1)),' nm / ',num2str(L(3,2)),' nm' ], ...
        [num2str(L(4,1)),' nm / ',num2str(L(4,2)),' nm' ], ...
        [num2str(L(5,1)),' nm / ',num2str(L(5,2)),' nm' ], ...
        [num2str(L(6,1)),' nm / ',num2str(L(6,2)),' nm' ], ...
        'location','EastOutside');

    Pos=[560 530 560 420];
    set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.5 Pos(4)]);

    % Save graphic:
    fileSaveName=['GouyChapmanPotential', ...
        num2str(round(X(l)*100)),'.eps'];
    set(gcf,'PaperpositionMode','auto')
    print(gcf,'-depsc','-r250',fileSaveName)

    disp(['Saved as: ', fileSaveName])
end

% Plot Gouy-Chapman potential for for different conversions

plot(a,PhiEl(:,1,1),'-b','LineWidth',lineWidth)
hold on
plot(a,PhiEl(:,1,2),'--r','LineWidth',lineWidth)
plot(a,PhiEl(:,1,3),'-.k','LineWidth',lineWidth)
hold off

% Graphic format:
set(gca,'FontSize',labelFontSize);
set(gca,'FontSize',figFontSize);
xlabel('Surface to surface distance a [nm]');
ylabel('Electrostatic potential \phi_{el} [J]');
axis([Xmin Xmax 0 Ymax]);

legend(['X = ',num2str(X(1))], ...
    ['X = ',num2str(X(2))], ...
```

```
    ['X = ',num2str(X(3))], ...
    'location','EastOutside');

Pos=[560 530 560 420];
set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.5 Pos(4)]);

% Save graphic:
fileSaveName=['GouyChapmanPotentialConversion.eps'];
set(gcf,'PaperpositionMode','auto')
print(gcf,'-depsc','-r250',fileSaveName)

disp(['Saved as: ', fileSaveName])


% Plot total interaction potential for different sizes

for l=1:p
    plot(a,PhiTot(:,1,l),'-r','LineWidth',lineWidth)
    hold on
    plot(a,PhiTot(:,2,l),'--b','LineWidth',lineWidth)
    plot(a,PhiTot(:,3,l),'-.b','LineWidth',lineWidth)
    plot(a,PhiTot(:,4,l),'-k','LineWidth',lineWidth/2)
    plot(a,PhiTot(:,5,l),'--k','LineWidth',lineWidth/2)
    plot(a,PhiTot(:,6,l),'-.k','LineWidth',lineWidth/2)
    plot([0 200], [0 0],'-k','LineWidth',0.25)
    hold off

    % Graphic format:
    set(gca,'FontSize',labelFontSize);
    set(gca,'FontSize',figFontSize);
    xlabel('Surface to surface distance a [nm]');
    ylabel('Total interaction potential \phi_{tot} [J]');
    axis([Xmin Xmax Ymin Ymax]);

    legend([num2str(L(1,1)),' nm / ',num2str(L(1,2)),' nm' ],...
        [num2str(L(2,1)),' nm / ',num2str(L(2,2)),' nm' ], ...
        [num2str(L(3,1)),' nm / ',num2str(L(3,2)),' nm' ], ...
        [num2str(L(4,1)),' nm / ',num2str(L(4,2)),' nm' ], ...
        [num2str(L(5,1)),' nm / ',num2str(L(5,2)),' nm' ], ...
        [num2str(L(6,1)),' nm / ',num2str(L(6,2)),' nm' ], ...
        'location','EastOutside');

    Pos=[560 530 560 420];
    set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.5 Pos(4)]);

    % Save graphic:
    fileSaveName=['TotalInteracPotential', ...
        num2str(round(X(l)*100)),'.eps'];
    set(gcf,'PaperpositionMode','auto')
    print(gcf,'-depsc','-r250',fileSaveName)

    disp(['Saved as: ', fileSaveName])
end

% Plot total interaction potential for for different conversions

plot(a,PhiTot(:,1,1),'-b','LineWidth',lineWidth)
hold on
plot(a,PhiTot(:,1,2),'--r','LineWidth',lineWidth)
plot(a,PhiTot(:,1,3),'-.k','LineWidth',lineWidth)
semilogx([0 200], [0 0],'-k','LineWidth',0.25)
```

```
        hold off

        % Graphic format:
        set(gca,'FontSize',labelFontSize);
        set(gca,'FontSize',figFontSize);
        xlabel('Surface to surface distance a [nm]');
        ylabel('Total interaction potential \phi_{tot} [J]');
        axis([Xmin Xmax Ymin Ymax]);

        legend(['X = ',num2str(X(1))], ...
            ['X = ',num2str(X(2))], ...
            ['X = ',num2str(X(3))], ...
            'location','EastOutside');

        Pos=[560 530 560 420];
        set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.5 Pos(4)]);

        % Save graphic:
        fileSaveName=['TotalInteracPotentialConversion.eps'];
        set(gcf,'PaperpositionMode','auto')
        print(gcf,'-depsc','-r250',fileSaveName)

        disp(['Saved as: ', fileSaveName])

    case 2
        % Plot surface potential

        PSI=PSI*1000; % [V -> mV]

        plot(cPDI,PSI(:,1),'-b','LineWidth',lineWidth);
        hold on
        plot(cPDI,PSI(:,2),'--r','LineWidth',lineWidth);
        plot(cPDI,PSI(:,3),'-.k','LineWidth',lineWidth);
        plot(cPDI,PSI(:,4),'-b','LineWidth',lineWidth/2);
        plot(cPDI,PSI(:,5),'--r','LineWidth',lineWidth/2);
        plot(cPDI,PSI(:,6),'-.k','LineWidth',lineWidth/2);

        % Graphic format:
        set(gca,'FontSize',labelFontSize);
        set(gca,'FontSize',figFontSize);
        xlabel('Concentration of PDI [mol/l]');
        ylabel('Particle surface potential [mV]');
        title('Surface potential');
        axis([min(cPDIpzc) max(cPDIpzc) -20 20]);
        legend(['c_{PDI}^{pzc} = ',num2str(cPDIpzc(1))], ...
            ['c_{PDI}^{pzc} = ',num2str(cPDIpzc(2))], ...
            ['c_{PDI}^{pzc} = ',num2str(cPDIpzc(3))], ...
            ['c_{PDI}^{pzc} = ',num2str(cPDIpzc(4))], ...
            ['c_{PDI}^{pzc} = ',num2str(cPDIpzc(5))], ...
            ['c_{PDI}^{pzc} = ',num2str(cPDIpzc(6))], ...
            'Location','SouthEast');

        % Save graphic:
        fileSaveName='SurfacePotential.eps'
        set(gcf,'PaperpositionMode','auto')
        print(gcf,'-depsc','-r250',fileSaveName)

        disp(['Saved as: ', fileSaveName])

    otherwise
end
```

PlotAggregationEfficiency.m

```
%% Plot Aggregation efficiency
% (c) by Andreas Eitzlmayr

clear

global z1
global z2
global D1
global D2

global k
global NA
global e
global c10
global c20
global T
global etaW
global A
global c2pzc
global eps0
global epsr

% Select Calculation:

ID1=2;

% ID1 =
% 0 ... Aggregation efficiency and -kernel as function of L1, L2
% 1 ... Aggregation efficiency as function of unknown parameter cPDIpcz
% 2 ... Aggregation efficiency as function of unknown parameter A


Spezies=0;

% 0 ... Polyacrylic acid + Protamine
% 1 ... Bariumsulfate

%% Input:

% Constants:

k=1.381e-23;      % [J/K] Boltzmann's constant
NA=6.022e23;     % [1/mol] Avogadro's constant
e=1.602e-19;    % [As] elementary charge
eps0=8.854e-12;         % [As/Vm] electric constant
epsr=80;                 % [] relative permittivity

K=0.414;   % Interfacial energy constant
           % according to Mersmann 2000 between 0.310 and 0.414
Kad=0;     % free variable for adjusted K

A=1e-20;    % Hamaker constant [J]


% Spezies specific parameters:
switch Spezies
    case 0 % Polyacrylic acid + Protamine
```

```
        name1='Polyacrylic acid cysteine';
        name2='Protamine';
        M1=5400;                % [g/mol] Molar masses
        M2=4300;
        z1=-56.98;                 % [] molecular charge numbers
        z2=21.27;
        rh1=1.4;           % [nm] hydrodynamic radius
        rh2=1.35;           % (corresponding to diffusion coefficient)
        rhoS=1400;              % [kg/m³] Solid density
        etaW=0.001;             % [Pa s] dynamic viscosity

        % Initial conditions:
        cm10=0.2;     %[g/l] Polyacrylic acid
        cm20=0.6;     %[g/l] Protamine
        c2pzc=1e-4;       %[mol/l] concentr. of Potential determining ions

        % Mixing volume ratio V1/V2:
        VolRatio=1;
    case 1 % Bariumsulfate

        name1='Barium';
        name2='Sulfate';
        M1=137.3;             % [g/mol] Molar masses
        M2=96.1;
        z1=2;                 % [] molecular charge numbers
        z2=-2;
        rh1=0.44482;          % [nm] hydrodynamic radius
        rh2=0.44482;          % (corresponding to diffusion coefficient)
        rhoS=4500;            % [kg/m³] Solid density
        etaW=0.001;           % [Pa s] dynamic viscosity

        % Initial conditions:
        cm10=0.1373;          %[g/l] Polyacrylic acid
        cm20=0.0961;          %[g/l] Protamine
        % Mixing volume ratio V1/V2:
        VolRatio=1;
end


% Integration parameters:
timeStep=2; % [s]
tmax=480;      % [s]

% Conditions:
T=293.15;                      % [K] Temperature

% Graphic format:
figFontSize=16;
labelFontSize=14;
lineWidth=2;

%% Calculation:

% Diffusion coefficients [m²/s]:
D1=k*T/(6*pi*etaW*rh1*1e-9);
D2=k*T/(6*pi*etaW*rh2*1e-9);

% Initial molar concentrations [mol/l]:
c10ini=cm10/M1;
c20ini=cm20/M2;
```

```
% Molar concentrations in the mixture:
c10=c10ini*VolRatio/(1+VolRatio);
c20=c20ini*1/(1+VolRatio);

switch ID1

     case 0
        % Calculate aggregation efficiency as function of sizes L1, L2

        X=[0 0.5 1];    % Reaction conversion (% consumed Polyacrylic acid)
        l=1;
        c1(l)=c10*(1-X(l));   % Concentrations due to determined conversion
        c2(l)=c20-c10*X(l)*(-z1/z2);

        m=100; % number of different particle sizes
        x=linspace(1,100,m)*1e-9; % particle sizes [m]

        W=zeros(m,m); % Stabilito ratio
        alpha=W;      % Aggregation efficiency
        betaAgg=W;    % Aggregation kernel

        n=100;  % number of points (distances)
        a=logspace(-11,-7,n); % [m] different distances

        for l=1:m      % change particle size L1
            for j=l:m  % change particle size L2

                % variables *_SI are arguments for function
                % StabilityIntegrand, they must be defined global, because
                % for integration of StablityIntegrand only 1 argument is
                % allowed

                L1=x(l);   % determine L1 and L2 due to index l
                L2=x(j);

                alpha(l,j)=AggEfficiency(L1,L2,c1(1),c2(1)); % Agg.
efficiency
                alpha(j,l)=alpha(l,j);


                %Aggregationkernel:
                betaAgg(l,j)=alpha(l,j)*Collisionkernel(x(l)/x(j));
                betaAgg(j,l)=betaAgg(l,j);
            end
        end

    case 1
        % Calculate aggregation efficiency as function of unknown cPDIpzc

        X=[0 0.5 1];    % Reaction conversion (% consumed Polyacrylic acid)
        l=1;
        c1(l)=c10*(1-X(l));
        c2(l)=c20-c10*X(l)*(-z1/z2);

        L1=5e-9;      % [nm ]constant particle sizes
        L2=100e-9;

        n=160; % number of points (different cPDIpzc concentrations)

        cPDIpzc=logspace(-5,-3,n); % different point of zero charge conc.
```

```
        AHam=[0.01 1 100]*1e-20;  % [J] different Hamaker constants
        %AHam=[3 10 20 30 100 200]*1e-20;  % [J] different Hamaker
        %constants
        m=size(AHam,2);  % number of different Hamaker constants

        W=zeros(n,m); % Stabilito ratio
        alpha=W;      % Aggregation efficiency

        p=100;  % number of points (surface/surface distances)
        a=logspace(-11,-6,p); % [m] different distances

        for l=1:m      % change Hamaker constant
            A=AHam(l);
            for j=1:n  % change concentration cPDIpzc

                c2pzc=cPDIpzc(j);

                alpha(l,j)=AggEfficiency(L1,L2,c1(1),c2(1)); % Agg.
efficiency
            end
        end

    case 2  % Aggregation efficiency as function of unknown parameter A

        X=[0 0.5 1];    % Reaction conversion (% consumed Polyacrylic acid)
        l=1;
        c1(l)=c10*(1-X(l));
        c2(l)=c20-c10*X(l)*(-z1/z2);

        L1=5e-9;      % [nm ]constant particle sizes
        L2=50e-9;
        L=[100 100;10 100;10 10]*1e-9; % [m] different size combinations
        m=size(L,1);  % number of different size combinations

        n=160; % number of points (different Hamaker constants A)

        cPDIpzc=1e-4; % [mol/l] point of zero charge conc.

        AHam=logspace(-22,-18,n);  % [J] different Hamaker constants

        W=zeros(n,m); % Stabilito ratio
        alpha=W;      % Aggregation efficiency

        p=100;  % number of points (surface/surface distances)
        a=logspace(-11,-5.5,p); % [m] different distances


        for l=1:m      % change sizes

            L1=L(l,1);   % determine L1 and L2 due to index l
            L2=L(l,2);

            for j=1:n  % change Hamaker constant

                A=AHam(j);
                alpha(l,j)=AggEfficiency(L1,L2,c1(1),c2(1)); % Agg.
efficiency
            end
        end
```

```
        % Electrostatic potential with HHF Theory
        % for model II (only aggregation)

% stability ratio / aggregation efficiency:

    otherwise
end

%% Grafic output

switch ID1

    case 0 % Aggregation efficiency as funktion of particle sizes L1, L2

        % Plot aggregation efficiency as funktion of particle sizes L1, L2

        x=x*1e9; % [m] -> [nm]

        alpha=alpha+0.001*max(max(alpha));

        colormap(jet);
        surf(x,x,alpha);
        view([0,0,1]);
        colorbar('location','EastOutside');

        load AggColorMap;
        set(gcf,'Colormap',AggColorMap)

        %colorbar('location','EastOutside');

        % Graphic format:
        set(gca,'FontSize',labelFontSize);
        set(gca,'FontSize',figFontSize);
        xlabel('Particle size L_{1} [nm]');
        ylabel('Particle size L_{2} [nm]');
        title('Aggregation efficiency \alpha');
        axis([min(x) max(x) min(x) max(x)]);

        % Save graphic:
        fileSaveName='AggregationEfficiency.eps';
        set(gcf,'PaperpositionMode','auto')
        print(gcf,'-depsc','-r250',fileSaveName)

        disp(['Saved as: ', fileSaveName])


        % Plot aggregation kernel as funktion of particle sizes L1, L2

        %betaAgg=betaAgg+0.001*max(max(alpha));

        surf(x,x,betaAgg);
        view([0,0,1]);
        colorbar('location','EastOutside');

        % Graphic format:
        set(gca,'FontSize',labelFontSize);
        set(gca,'FontSize',figFontSize);
        xlabel('Particle size L_{1} [nm]');
```

```
        ylabel('Particle size L_{2} [nm]');
        title('Aggregation kernel \beta_{agg} [m^{3}/s]');
        axis([min(x) max(x) min(x) max(x)]);

        % Save graphic:
        fileSaveName='AggregationKernel.eps';
        set(gcf,'PaperpositionMode','auto')
        print(gcf,'-depsc','-r250',fileSaveName)

        disp(['Saved as: ', fileSaveName])

    case 1  % Aggregation efficiency as function of unknown parameter
cPDIpzc

        % Plot aggregation efficiency as function of unknowns A, cPDIpzc

        semilogx(cPDIpzc,alpha(3,:),'-b','LineWidth',lineWidth)
        hold on
        semilogx(cPDIpzc,alpha(2,:),'--r','LineWidth',lineWidth)
        semilogx(cPDIpzc,alpha(1,:),'-.k','LineWidth',lineWidth)
        hold off

        % Graphic format:
        set(gca,'FontSize',labelFontSize);
        set(gca,'FontSize',figFontSize);
        xlabel('Concentration of PDI at pzc [mol/l]');
        ylabel('Aggregation efficiency');
        %axis([3e-5 3e-4 0 1.5]);

        L1=L1*1e9; % [m] -> [nm]
        L2=L2*1e9; % [m] -> [nm]

        AHam=AHam*1e20;

        legend([num2str(AHam(3)),'x10^{-20} J'], ...
            [num2str(AHam(2)),'x10^{-20} J'], ...
            [num2str(AHam(1)),'x10^{-20} J'],'Location','EastOutside');

        Pos = get(gcf,'Position');
        set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.5 Pos(4)]);

        % Save graphic:
        fileSaveName=['AggregationEffVaryUnknowns_L1-',num2str(L1), ...
            '_L2-',num2str(L2),'_X-',num2str(X(1)*100),'.eps'];
        set(gcf,'PaperpositionMode','auto')
        print(gcf,'-depsc','-r250',fileSaveName)

        disp(['Saved as: ', fileSaveName])

    case 2  % Aggregation efficiency as function of unknown parameter A

        % Plot aggregation efficiency as function of unknowns A, cPDIpzc

        AHam=AHam*1e20;

        semilogx(AHam,alpha(1,:),'-b','LineWidth',lineWidth)
        hold on
        semilogx(AHam,alpha(2,:),'--r','LineWidth',lineWidth)
        semilogx(AHam,alpha(3,:),'-.k','LineWidth',lineWidth)

        hold off
```

```
        % Graphic format:
        set(gca,'FontSize',figFontSize);
        xlabel('Hamaker constant [10^{-20} J]');
        ylabel('Aggregation efficiency');
        %axis([3e-5 3e-4 0 1.5]);

        L=L*1e9; % [m] -> [nm];


        AHam=AHam*1e20;

        legend([num2str(L(1,1)),' nm / ',num2str(L(1,2)),' nm'], ...
            [num2str(L(2,1)),' nm / ',num2str(L(2,2)),' nm'], ...
            [num2str(L(3,1)),' nm / ',num2str(L(3,2)),' nm'], ...
            'Location','NorthWest');

        % Save graphic:
        fileSaveName=['AggregationEfficiencyHam_cPDIpzc-', ...
            num2str(cPDIpzc),'_X-',num2str(X(1)*100),'.eps'];
        set(gcf,'PaperpositionMode','auto')
        print(gcf,'-depsc','-r250',fileSaveName)

        disp(['Saved as: ', fileSaveName])

    otherwise
end
```

IV. Solution for the Well-Mixed System

WellMixedCM.m

```
%% Precipitation well mixed, PBE solution with classes method
% (c) by Andreas Eitzlmayr

clear

global z1
global z2
global cequ
global D1
global D2
global rh1
global rh2
global cS1
global cS2
global S
global rhoS

global k
global NA
global K
global c10
global c20
global T
global etaW
global Vm
global sig
global cS
global nd
global Spezies
global c2pzc
global A

global ClassesNum
global L
global Lm
global GrowthConst
global MP

global e
global eps0
global epsr
global fid
global CollFactor
global c2Excess
global c1End

Spezies=0;

% 0 ... Polyacrylic acid + Protamine
% 1 ... Bariumsulfate

Calculate=1;

% 0 ... Don'T solve the sytem (only pre- and postprocessing)
```

```
% 1 ... Solve the system (prec. + solving + postprocessing)

%% Input:

% Constants:

k=1.380650424e-23;      % [J/K] Boltzmann's constant
NA=6.0221417930e23;       % [1/mol] Avogadro's constant
e=1.602e-19;    % [As] elementary charge
eps0=8.854e-12;         % [As/Vm] electric constant
epsr=80;                 % [] relative permittivity

A=1e-20;    % Hamaker constant [J]

K=0.39;   % Interfacial energy constant
           % according to Mersmann 2000 between 0.310 and 0.414

Sh=2;      % particle sherwood number for molecular growth

rhoW=997.77;  % density of water [kg/m³]

% Spezies specific parameters:
switch Spezies
    case 0 % Polyacrylic acid + Protamine

        name1='Polyacrylic acid cysteine';
        name2='Protamine';
        M1=5400;                % [g/mol] Molar masses
        M2=4300;
        z1=-56.98;                 % [] molecular charge numbers
        z2=21.27;
        rh1=1.40;           % [nm] hydrodynamic radius
        rh2=1.35;           % (corresponding to diffusion coefficient)
        rhoS=1400;              % [kg/m³] Solid density
        etaW=0.001;             % [Pa s] dynamic viscosity

        cequ=1e-10;             % [mol/l] Equilibium concentration

        % Initial concentrations:
        cm10=0.2;     % [g/l] Polyacrylic acid
        cm20=0.6;     % [g/l] Protamine
        c2pzc=7e-5;       %[mol/l] concentr. of Potential determining ions

        % Mixing volume ratio V1/V2:
        VolRatio=1;

    case 1 % Bariumsulfate

        name1='Barium';
        name2='Sulfate';
        M1=137.3;              % [g/mol] Molar masses
        M2=96.1;
        z1=2;                  % [] molecular charge numbers
        z2=-2;
        rh1=0.44482;           % [nm] hydrodynamic radius
        rh2=0.44482;           % (corresponding to diffusion coefficient)
        rhoS=4500;             % [kg/m³] Solid density
        etaW=0.001;            % [Pa s] dynamic viscosity

        cequ=sqrt(1.01e-4)*1e-3;  % [mol/l] Equilibium concentration
```

```
        B_Desired=1.18e11;     % [1/m³s] Desired nucleation rate to adjust
K

        % Initial concentrations:
        cm10=0.1373;           %[g/l] Polyacrylic acid
        cm20=0.0961;           %[g/l] Protamine
        % Mixing volume ratio V1/V2:
        VolRatio=1;

end


% Integration parameters:

tmax=0.6;
timeStep=0.001; % [s]
SampleTime=[0.001 0.01 0.03 0.1 0.2 0.3 0.4 0.6]; % [s]

% Conditions:
T=295.15;                      % [K] Temperature

% Graphic format:
FontSize=18;
lineWidth=2;

%% Initialize Logfile

LogFile='Results/LogWellMixed.txt';

fid=fopen(LogFile,'w');
fclose(fid);
fid=fopen(LogFile,'a');


cequPot=fix(log10(cequ));
c2Pot=fix(log10(c2pzc))-1;
APot=fix(log10(A));

DataName=['Results/K0_',num2str(round(10000*K)), ...
    'cEqu',num2str(round(cequ/10^cequPot)),'e',num2str(cequPot), ...
    'c2pzc',num2str(round(c2pzc/10^c2Pot)),'e',num2str(c2Pot), ...
    'A',num2str(round(A/10^APot)),'e',num2str(APot), ...
    'cm10_0_', num2str(round(100*cm10)), ...
    'cm20_0_', num2str(round(100*cm20)), ...
    'OnlyAgg', num2str(OnlyAgg),'.mat'];

disp(DataName);
%% Calculation of parameters:

%Diffusion coefficients [m²/s]:
D1=k*T/(6*pi*etaW*rh1*1e-9);
D2=k*T/(6*pi*etaW*rh2*1e-9);

% Initial molar concentrations [mol/l]:
c10ini=cm10/M1;
c20ini=cm20/M2;

% Molar concentrations in the mixture:
c10=c10ini*VolRatio/(1+VolRatio);
c20=c20ini*1/(1+VolRatio);
```

```
%Excess concentration of protamine [mol/l]:
c2Excess=c20+c10*z1/z2;

%Remaining concentration of PAC [mol/l]:
c1End=cequ^2/c2Excess;

% Solid concentrations [mol/l]:
% (due to electrical neutrality)
cS1=z2*rhoS/(z2*M1-z1*M2); % Polyacrylic acid (1)
cS2=-cS1*z1/z2;            % Protamine (2)
nd=2;%1-z1/z2;                % Dissociation number
cS=cS1+cS2;           % Total

% Mean solid molecular volume:
Vm=1/(1000*NA*cS);    %[m³]

% Interfacial energy:
sig=K*k*T*(1000*cS*NA)^(2/3)*log(cS/cequ); %[J/m²]

% Growth constant:
GrowthConst=2*Sh*cequ/cS; % [-]

% Mean molecular weight of particles:
MP = (M1*cS1+M2*cS2)/(cS1+cS2); % [g/mol]

% Dimensioned part of collision kernel:
CollFactor=2*k*T/(3*etaW); % [m³/s]


%% Calculate Timescales:

EpsTurb=10;            % dissipation rate [W/kg]
CharLength=0.001;      % characteristic length of the reactor [m]
Dmean=(D1+D2)/2;       % mean diffusion coefficient [m²/s]
Sc=etaW/(rhoW*Dmean);  % Schmidt-Number
Nexpected=3.3e17;      % Expected final particle number [1/m³]
Lgrowth=50e-9;         % mean particle size for growth timemscale [m]
SizeRatio=0.1;         % Size ratio for collision timescale

Tmacro=5*(CharLength^2/EpsTurb)^(1/3);
Tmicro=5*log(Sc)*sqrt(etaW/(rhoW*EpsTurb));
Tnuc=Nexpected/NucleationRate(Dmean,sqrt(c10*c20),cequ,sig,T,Vm,nd);
Tgrowth=1/(2*Sh*Dmean*Lgrowth*pi*Nexpected);
Tcoll=1/(Collisionkernel(SizeRatio)*Nexpected);

% Print timescales
disp(['Timescale for Macromixing:        ', num2str(Tmacro), ' s'])
disp(['Timescale for Micromixing:        ', num2str(Tmicro), ' s'])
disp(['Timescale for Nucleation:         ', num2str(Tnuc), ' s'])
disp(['Timescale for Growth:             ', num2str(Tgrowth), ' s'])
disp(['Timescale for Collisions:         ', num2str(Tcoll), ' s'])

%% Solve System

% Discretisation of internal coordinate (size classes):


ClassesNum=40;  % Number of classes
Lmin=2;    %[nm]
Lmax=300;  %[nm]
L=ones(1,ClassesNum+1)*Lmin*1e-9 + linspace(0,1,ClassesNum+1).^2*...
```

```
    (Lmax-Lmin)*1e-9;  % upper and lower sizes of classes

Lm=zeros(1,ClassesNum);   % mean sizes of classes

Lm(1)=(L(2)+L(1))/2; % [m]

for i=3:ClassesNum+1
    Lm(i-1)=(L(i)+L(i-1))/2;
end

if Calculate==1
    % Precalculate Aggregation Efficiency (for Interpolation):
    CalcAggEfficiency;
end

disp('Solving ODE-System ...')

% Initial conditions:

y0=zeros(1,ClassesNum+2);
y0(ClassesNum+1)=c10;
y0(ClassesNum+2)=c20;


if Calculate==1
    % Solve ODE System:
    options = odeset('RelTol',1e-3);
    %,'AbsTol',ones(1,ClassesNum+2)*1e-1
    [t,y]=ode45('OdeWellmixedCM',0:timeStep:tmax,y0,options);
    data=[t,y];
    save(DataName,'data','-mat')
else
    load(DataName)
    t=data(:,1);
    y=data(:,2:size(data,2));
end

% Calculate Supersaturation, critical Radius:
 [m,n]=size(y);

S=zeros(m,1);
rC=zeros(m,1);

for i=1:m
    S(i)=sqrt(y(i,ClassesNum+1)*y(i,ClassesNum+2))/cequ;
    S(i)=real(S(i)); % complex numbers occur due to inaccuracy
    if S(i)<1   % Solution not exact, make correction:
        S(i)=1;
        y(i,ClassesNum+1)=cequ^2/y(i,ClassesNum+2);
    end
    rC(i)=2*sig*Vm/(nd*k*T*log(S(i)))*1e9; % [nm]

    for j=1:ClassesNum
        if y(i,j)<0
                    y(i,j)=0;
        end
    end

    % Convert PSD from q0 to q3:
    y3Sum=0;
```

```
    for j=1:ClassesNum
        y3(i,j)=y(i,j)*Lm(j)^3;
        y3Sum=y3Sum+y3(i,j)*(L(j+1)-L(j));
    end

    for j=1:ClassesNum
        y3(i,j)=y3(i,j)/y3Sum;
    end

end



% Calculate PSD Mean and Variance:

ns=size(SampleTime,2);
Lmean_q3=zeros(ns,1);
LSigma_q3=zeros(ns,1);
Lmean_n=zeros(ns,1);
LSigma_n=zeros(ns,1);
nSum=zeros(ns,1);

n=zeros(ns,ClassesNum); % number density distribution
q3=zeros(ns,ClassesNum); % Volume distribution

for j=1:ns
    Idx=1+int16(SampleTime(j)/timeStep);

    % Separate Samples
    for i=1:ClassesNum
        n(j,i)=y(Idx,i);
        q3(j,i)=y3(Idx,i);
    end

    % Calculate PSD Mean Values:
    for i=1:ClassesNum
        Lmean_q3(j)=Lmean_q3(j)+q3(j,i)*Lm(i)*(L(i+1)-L(i));
        Lmean_n(j)=Lmean_n(j)+n(j,i)*Lm(i)*(L(i+1)-L(i));
        nSum(j)=nSum(j)+n(j,i)*(L(i+1)-L(i));
    end
    Lmean_n(j)=Lmean_n(j)/nSum(j);

    % Calculate PSD Variances:
    for i=1:ClassesNum
        LSigma_q3(j)=LSigma_q3(j)+(Lm(i)-Lmean_q3(j))^2*q3(j,i)* ...
            (L(i+1)-L(i));
        LSigma_n(j)=LSigma_n(j)+(Lm(i)-Lmean_n(j))^2*n(j,i)* ...
            (L(i+1)-L(i));
    end
    LSigma_q3(j)=sqrt(LSigma_q3(j));
    LSigma_n(j)=sqrt(LSigma_n(j)/nSum(j));

    % Convert to nm and round:
    Lmean_n(j)=0.1*round(Lmean_n(j)*1e10);
    Lmean_q3(j)=0.1*round(Lmean_q3(j)*1e10);
    LSigma_n(j)=0.1*round(LSigma_n(j)*1e10);
    LSigma_q3(j)=0.1*round(LSigma_q3(j)*1e10);
end

% Select data for 3D Plot:
num=101;
```

```
n3D=zeros(num,ClassesNum);
q33D=zeros(num,ClassesNum);
t3D=zeros(num,1);

for i=1:num
    Idx=fix((m-1)/num*i)+1;
    n3D(i,:)=y(Idx,1:ClassesNum);
    q33D(i,:)=y3(Idx,1:ClassesNum);
    t3D(i)=t(Idx)*1000;
end

fclose(fid);

%% Graphical Output:

% PSD:

for j=1:ns

    % Plot single PSD n (one point of time):
    plot(Lm*1e9,n(j,:),'LineWidth',lineWidth);

    fileSaveName=['Results/FigWellmixedPSDn',num2str(j),'.eps'];
    set(gca,'FontSize',FontSize);
    xlabel('Particle size L [nm]');
    ylabel('Number density distribution n [1/m^{4}]');
    title(['t=',num2str(SampleTime(j)*1000),'ms, L_{mean}=', ...
        num2str(Lmean_n(j)),'nm, s=',num2str(LSigma_n(j)),'nm']);

    Pos = get(gcf,'Position');
    set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.2 Pos(4)]);

    set(gcf,'PaperpositionMode','auto');
    print(gcf,'-depsc','-r250',fileSaveName)
    disp(['Saved as: ', fileSaveName]);
    close

    % Plot single PSD q3 (one point of time):
    plot(Lm*1e9,q3(j,:),'LineWidth',lineWidth);

    fileSaveName=['Results/FigWellmixedPSDq3',num2str(j),'.eps'];
    set(gca,'FontSize',FontSize);
    xlabel('Particle size L [nm]');
    ylabel('Volume distribution q_{3} [1/m]');
    title(['t=',num2str(SampleTime(j)*1000),'ms, L_{mean}=', ...
        num2str(Lmean_q3(j)),'nm, s=',num2str(LSigma_q3(j)),'nm']);

    Pos = get(gcf,'Position');
    set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.2 Pos(4)]);

    set(gcf,'PaperpositionMode','auto');
    print(gcf,'-depsc','-r250',fileSaveName)
    disp(['Saved as: ', fileSaveName]);
    close
end

% Plot final q3 PSD logarithmical:
semilogx(Lm*1e9,q3(ns,:),'LineWidth',lineWidth);

fileSaveName=['Results/FigWellmixedPSDfinal_log.eps'];
set(gca,'FontSize',FontSize);
```

```
xlabel('Particle size L [nm]');
ylabel('Volume distribution q_{3} [1/m]');
axis([10 1000 0 2e7]);
title(['t=',num2str(SampleTime(ns)*1000),'ms, L_{mean}=', ...
    num2str(Lmean_q3(ns)),'nm, s=',num2str(LSigma_q3(ns)),'nm']);

Pos = get(gcf,'Position');
set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.2 Pos(4)]);

set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);
close

% Plot single PSD q3 (one point of time):
LineFormat=char('-ro','-b^','-k*','-mx','-rd','-bs','-k+','-m');
lineWidth=[0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6];

% LineFormat=char('-k','-b',':m',':r','-.g','-.c','--y','--k');
% lineWidth=[1.5 0.5 1.5 0.5 1.5 0.5 1.5 0.5];

% Plot all n PSDs in 1 plot:

plot(Lm*1e9,n(1,:),LineFormat(1,:),'LineWidth',lineWidth(1));
hold on

for i=2:ns
    plot(Lm*1e9,n(i,:),LineFormat(i,:),'LineWidth',lineWidth(i));
end

hold off

fileSaveName=['Results/FigWellmixedPSDn_all.eps'];
set(gca,'FontSize',FontSize);
xlabel('Particle size L [nm]');
ylabel('Number density distribution n [1/m^{4}]');
legend(['t = ',num2str(SampleTime(1)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(2)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(3)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(4)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(5)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(6)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(7)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(8)*1000),' ms'], ...
    'location','EastOutside')

Pos = get(gcf,'Position');
set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.6 Pos(4)]);

set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);
close


% All q3 PSDs in one plot:
plot(Lm*1e9,q3(1,:),LineFormat(1,:),'LineWidth',lineWidth(1));
hold on

for i=2:ns
    plot(Lm*1e9,q3(i,:),LineFormat(i,:),'LineWidth',lineWidth(i));
```

```
end

hold off

lineWidth=2;

fileSaveName=['Results/FigWellmixedPSDq3_all.eps'];
set(gca,'FontSize',FontSize);
xlabel('Particle size L [nm]');
ylabel('Volume distribution q_{3} [1/m]');
legend(['t = ',num2str(SampleTime(1)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(2)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(3)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(4)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(5)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(6)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(7)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(8)*1000),' ms'], ...
    'location','EastOutside')

Pos = get(gcf,'Position');
set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.6 Pos(4)]);

set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);
close


% 3D Plot of n PSDs:
surf(Lm*1e9,t3D,n3D(:,1:ClassesNum));
colormap('Jet');
view([1,-5,10]);

fileSaveName=['Results/FigWellmixed_3D_PSDn.eps'];
set(gca,'FontSize',FontSize);
xlabel('Particle size L [nm]');
ylabel('Time [ms]');
title('Number density distribution n [1/m^{4}]');
set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);


% 3D Plot of p6 PSDs:
surf(Lm*1e9,t3D,q33D(:,1:ClassesNum));
view([1,-5,10]);

load MyColormap;
set(gcf,'Colormap',MyColorMap)

fileSaveName=['Results/FigWellmixed_3D_PSDq3.eps'];
set(gca,'FontSize',FontSize);
xlabel('Particle size L [nm]');
ylabel('Time [ms]');
title('Volume distribution q_{3} [1/m]');
set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);
```

```
% Concentrations:
t=t*1000;

fileSaveName='Results/FigWellmixed_Concentrations.eps';
plot(t,y(:,ClassesNum+1),t,y(:,ClassesNum+2),'LineWidth',lineWidth);
legend('c_{1}','c_{2}');
set(gca,'FontSize',FontSize*1.33);
xlabel('Time [ms]');
ylabel('Concentrations [mol/l]');
%axis([min(t), max(t),0, max(y(:,ClassesNum+2))]);
set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);

ConcWellMixed=[t,y(:,ClassesNum+1),y(:,ClassesNum+2)];
save('ConcWellMixed.mat','ConcWellMixed','-mat')

% Supersaturation:
fileSaveName='Results/FigWellmixed_Supersat.eps';
semilogy(t,S,'LineWidth',lineWidth);
set(gca,'FontSize',FontSize*1.33);
xlabel('Time [ms]');
ylabel('Supersaturation');
axis([min(t) max(t) 1 10^(fix(log10(max(S))/2)*2+2)]);
set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);

SupWellMixed=[t,S];
save('SupWellMixed.mat','SupWellMixed','-mat')

Matrix1=[Lm,Lmean_q3(ns)];
Matrix2=[q3(ns,:),LSigma_q3(ns)];
FinalQ3WellM=[Matrix1;Matrix2];
save('FinalQ3WellM.mat','FinalQ3WellM','-mat')

% Critical Radius:
fileSaveName='Results/FigWellmixed_CriticalRadius.eps';
plot(t,rC,'LineWidth',lineWidth);
axis([min(t), max(t), 0, max(rC)]);
set(gca,'FontSize',FontSize*1.33);
xlabel('Time [ms]');
ylabel('Critical Radius [nm]');
set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);


%% Output:

% Text:

disp(' ')
disp('************************************************************')
disp(' ')

disp('INPUT:')

disp([name1,' (1):'])
disp(['Molar mass:                        ', num2str(M1), ' g/mol'])
disp(['Electrical charge number:          ', num2str(z1)])
```

```
disp(['hydrodynamic radius:              ', num2str(rh1), ' nm'])
disp(['Initial mass concentration:       ', num2str(cm10), ' g/l'])
disp(' ')

disp([name2,' (2):'])
disp(['Molar mass:                       ', num2str(M2), ' g/mol'])
disp(['Electrical charge number:         ', num2str(z2)])
disp(['hydrodynamic radius:              ', num2str(rh2), ' nm'])
disp(['Initial mass concentration:       ', num2str(cm20), ' g/l'])
disp(' ')

disp(['Interfacial energy constant:      ', num2str(K)])
disp(['Dissociation number:              ', num2str(nd)])
disp(['Solid density:                    ', num2str(rhoS), ' kg/m³'])
disp(['Dynamic fluid viscosity:          ', num2str(etaW), ' Pa s'])
disp(['Solubility product:               ', num2str(cequ^2), ...
    ' (mol/l)²'])
disp(['Temperature:                      ', num2str(T), ' K'])
disp(' ')
disp(' ')

disp('OUTPUT:')

disp(['Diffusion coefficient 1:          ', num2str(D1), ' m²/s'])
disp(['Diffusion coefficient 2:          ', num2str(D2), ' m²/s'])
disp(['Initial concentration 1:          ', num2str(c10), ' mol/l'])
disp(['Initial concentration 2:          ', num2str(c20), ' mol/l'])
disp(['Solid concentration 1:            ', num2str(cS1), ' mol/l'])
disp(['Solid concentration 2:            ', num2str(cS2), ' mol/l'])
disp(['Total solid concentration:        ', num2str(cS), ' mol/l'])
disp(['Mean solid molecular volume:      ', num2str(Vm), ' m³'])
disp(['Equilibrium concentration:        ', num2str(cequ), ' mol/l'])
disp(['Interfacial energy:               ', num2str(sig), ' J/m²'])
```

CalcAggEfficiency.m

```
%% Calculate Aggregation efficiency
% (c) by Andreas Eitzlmayr

global z1
global z2
global c10
global c20

global ClassesNum
global Lm

global CollFactor
global AggregationTable
global AggT
global Conversion

%% Calculation

disp(' ')
disp('Precalculating Aggregation Efficiency ...')

sx=11;
f=1;    % natural number
% (Factor to reduce Classes in look-up table for Aggregation efficiency)
CalculatedClasses=(ClassesNum-1)/f+1;
% Classes wherefore Aggregation kernel will be calculated (others
interpolated)

Conversion=linspace(0,1,sx);
AggregationTable=ones(ClassesNum, ClassesNum, sx);
AggT=ones(sx,CalculatedClasses,CalculatedClasses);


f=(ClassesNum-1)/(CalculatedClasses-1);
if int8(f)==f
    % OK
else
    disp('ClassesNum+1 is not equal CalculatedClasses*f+1 !')
end

% Calculate Aggregation Efficiency:

for iC=1:CalculatedClasses
    for jC=1:iC
        for l=1:sx
            c1=c10*(1-Conversion(l));
            c2=c20-c10*Conversion(l)*(-z1/z2);
            i=(iC-1)*f+1;
            j=(jC-1)*f+1;
            AggregationTable(i,j,l)= ...
                CollFactor*(2+Lm(i)/Lm(j)+Lm(j)/Lm(i)) ...
                *AggEfficiency(Lm(i),Lm(j),c1,c2);
            AggregationTable(j,i,l)=AggregationTable(i,j,l);
            AggT(l,i,j)=AggregationTable(i,j,l);
            AggT(l,j,i)=AggregationTable(i,j,l);
        end
    end
    disp(['Class ', num2str(i), ' finished ...'])
```

```
end

% Interpolate missing values:

for l=1:sx
    for iC=1:CalculatedClasses
        i=(iC-1)*f+1;
        for j=1:ClassesNum
            y=(j-1)/f+1;

            if y==fix(y)
                % no interpolation
            else
                % interpolate in j direction:
                j1=(fix(y)-1)*f+1;
                j2=j1+f;
                AggregationTable(i,j,l)=(AggregationTable(i,j1,l)* ...
                    (j2-j)+ AggregationTable(i,j2,l)*(j-j1))/f;
                AggregationTable(j,i,l)=AggregationTable(i,j,l);
            end
        end
    end

    for i=1:ClassesNum
        for j=1:i
            x=(i-1)/f+1;
            y=(j-1)/f+1;

            if fix(x)==x || fix(y)==y % || = logical or
                % no interpolation
            else
                % interpolate in i direction:
                i1=(fix(x)-1)*f+1;
                i2=i1+f;
                AggregationTable(i,j,l)=(AggregationTable(i1,j,l)* ...
                    (i2-i)+ AggregationTable(i2,j,l)*(i-i1))/f;
                AggregationTable(j,i,l)=AggregationTable(i,j,l);
            end
        end
    end
end


%surf(Lm,Lm,AggregationTable(:,:,11));
surf(Lm,Conversion,AggT(:,:,1));
disp('Precalculation of Aggregation Efficiency finished')
```

OdeWellMixedCM.m

```
function dy=OdeWellmixedCM(t,y)
% ODE-System for Nucleation + Growth + Aggregation

global cequ
global D1
global D2
global cS1
global c10
global k
global T
global Vm
global sig
global nd
global z1
global z2

global ClassesNum
global L
global Lm
global GrowthConst
global fid
global Conversion
global AggregationTable

dy=zeros(ClassesNum+2,1);

% y(1 - ClassesNum) ... particle number density of every class
% y(ClassesNum+1) ... Concentration of Polyacrylic acid [mol/l]
% y(ClassesNum+2) ... Concentration of Protamine [mol/l]

for i=1:ClassesNum+2
    %y(i)=real(y(i));
    if y(i)<0
        y(i)=0;
    end
end

% Mean concentration:
c=sqrt(y(ClassesNum+1)*y(ClassesNum+2)); %[mol/l]

% Supersaturation:
S=c/cequ; %[-]

if S<=1
    S=1;
    y(ClassesNum+1)=cequ^2/y(ClassesNum+2);
    c=cequ;
end

% Critical radius:
rc=2*sig*Vm/(nd*k*T*log(S+1e-10));

% Mean diffusion coefficient:
D = (D1*y(ClassesNum+1) + D2*y(ClassesNum+2)) / ....
    (y(ClassesNum+1) + y(ClassesNum+2));

% Nucleation rate:
```

```
Bhom = NucleationRate(D,c,cequ,sig,T,Vm,nd);

% Write data to Log File:
fprintf(fid,'t = %2.8f    ',t);


% Evaluate Aggregation Kernel:
valueAggKernel=zeros(ClassesNum,ClassesNum);

X=1-y(ClassesNum+1)/c10; % Actual conversion


% determine Conversion Index:
if X==1
    CIdx=size(Conversion,2);
    IntFac=0;
else
    i=1;
    while  X>=Conversion(i)
         if X<Conversion(i+1)
             % Conversion Index:
             CIdx=i;
             % Interpolation factor:
             IntFac=(X-Conversion(i))/(Conversion(i+1)-Conversion(i));
         end
         i=i+1;
    end
end

for i=1:ClassesNum
    for j=1:i
        if X==1
            valueAggKernel(i,j)=AggregationTable(i,j,CIdx);
        else
            valueAggKernel(i,j)=AggregationTable(i,j,CIdx)*(1-IntFac)+...
                AggregationTable(i,j,CIdx+1)*IntFac;
        end
        valueAggKernel(j,i)=valueAggKernel(i,j);
    end
end

TotalBirth=0;
TotalDeath=0;
ySum=0;

% Aggregation:

AggBirth=zeros(ClassesNum,1);
AggDeath=zeros(ClassesNum,1);

for i=1:ClassesNum
    ySum = ySum + y(i) * (L(i+1)-L(i));

    for p=1:ClassesNum
        CollisionRate = valueAggKernel(i,p) * y(p) * y(i) * ...
            (L(p+1)-L(p)) * (L(i+1)-L(i)); % [1/m³s]

        AggDeath(i) = AggDeath(i) + CollisionRate / (L(i+1)-L(i));
        TotalDeath = TotalDeath + CollisionRate;

        Lresult=(Lm(i)^3+Lm(p)^3)^(1/3);  % Size of the resulting particle
```

```
        if Lresult<=Lm(1)
            l=1;
            AggBirth(l) = AggBirth(l) +  0.5 * CollisionRate / ...
                (L(l+1)-L(l));
            TotalBirth = TotalBirth + CollisionRate/2;

        elseif Lresult>Lm(ClassesNum)
            l=ClassesNum;
            AggBirth(l) = AggBirth(l) +  0.5 * CollisionRate / ...
                (L(l+1)-L(l));
            TotalBirth = TotalBirth + CollisionRate/2;
        else
            j=1;
            while Lresult>Lm(j)   % Is Lresult in this class j ?
                if Lresult<=Lm(j+1)
                    l=j; % Resulting particle is between Lm(l) and Lm(l+1)

     % Resulting particle is partitioned between 2 classes Lm(l) and
Lm(l+1):
     % (due to particle number conservation and mass conservation)

                    w1 = (1 - (Lresult/Lm(l+1))^3) / ...
                        (1 - (Lm(l)/Lm(l+1))^3);
                    w2 = 1 - w1;

                    AggBirth(l) = AggBirth(l) +  w1/2 * ...
                        CollisionRate / (L(l+1)-L(l));
                    AggBirth(l+1) = AggBirth(l+1) + w2/2 * ...
                        CollisionRate / (L(l+2)-L(l+1));
                    TotalBirth = TotalBirth + CollisionRate/2;
                end
                j=j+1;
            end
        end

    end

end

if TotalDeath==TotalBirth*2
    % OK
else
    disp('Collision Number conservation Error')
    disp('TotalDeath = ',num2str(TotalDeath))
    disp('TotalBirth x 2 = ',num2str(TotalBirth*2))
end

% ODE's:
% Particle classes
for i=1:ClassesNum

    %Nucleation:

    Nuc=0;  % Set Nucleation to zero (default)
    if 2*rc>=L(i)  % Is critical nuclei size within this class?
        if 2*rc<L(i+1)
            Nuc = Bhom/(L(i+1)-L(i)); % Set Nucelation term
        end
    end
```

```
    % Growth (Upwind Discretisation)

    if i==1 % lower boundary of internal coordinate L
        Growth=0;
        %Growth=GrowthConst*D*(S-1)*(y(i+1)/Lm(i+1) - ...
        % y(i)/Lm(i))/(Lm(i+1)-Lm(i));
    elseif i==ClassesNum % upper boundary of internal coordinate L
        Growth=GrowthConst*D*(S-1)*(y(i)/Lm(i) - ....
            y(i-1)/Lm(i-1))/(Lm(i)-Lm(i-1));
    else
        Growth=GrowthConst*D*(S-1)*(y(i)/Lm(i) - ...
            y(i-1)/Lm(i-1))/(Lm(i)-Lm(i-1));
    end

    % Balance:

    dy(i) = Nuc - Growth + AggBirth(i) - AggDeath(i);

end


GrowthSum=0;

for i=1:ClassesNum
    GrowthSum = GrowthSum + Lm(i)*y(i)*(L(i+1)-L(i)); % [1/m²]
end

GrowthSum = GrowthSum*GrowthConst*D*(S-1)*pi*cS1/2; % [kmol/m³s]

% Polyacrylic acid concentration
dy(ClassesNum+1) = -Bhom * 4*pi/3*rc^3*cS1 - GrowthSum;

% Protamine concentration
dy(ClassesNum+2) = dy(ClassesNum+1)*(-z1/z2);  % (cS2/cS1=-z1/z2)

% Write data to Log File:
fprintf(fid,'  Sup %6.3f  ',S);
% dc1/dt due to nucleation
fprintf(fid,'  Nuc %1.10f  ',-Bhom * 4*pi/3*rc^3*cS1);
% dc1/dt due to growth
fprintf(fid,'  Growth %1.10f  ',-GrowthSum/(1-z1/z2));
% disappered Particles per second and sum of particles
fprintf(fid,'  Agg.TotalBirth %3.10f  ',TotalDeath/ySum);

% Write data to Log File:
fprintf(fid,'%1.0f **\n',0);
```

V. Engulfment Model

XMeso.m

```
%% Mesomixed volume fraction

function X=XMeso(t)

global tmeso
global X0

% Calculate:
X = exp(t/tmeso)/(exp(t/tmeso)-1+1/X0);
```

Xmicro.m

```
%% Micromixed volume fraction

function X=XMicro(t)

global tmeso
global E
global X0

% Calculate:
X = 1/(E*(1-X0)/((E-1/tmeso)*X0)*(exp(-t/tmeso)-exp(-E*t))+ ...
    (1/X0-1)*exp(-E*t)+1);
```

EngulfmentModel.m

```
%% Engulfment Model
% (c) by Andreas Eitzlmayr

clear

global tmeso
global E
global X0

FontSize=16;

%% Input

n=50;
t=linspace(0,0.005,n); % Calculation time [s]

ny=1e-6;  % viscosity [m²/s]
eps=1000; % power input [W/kg=m²/s³]
d=0.0005; % [m] inlet diameter

%% Calculation

% Model parameters:
E=1/(12.7*(ny/eps)^0.5); % [1/s]
tmeso=2*(d^2/eps)^(1/3); % [s]

% Initial condition:
X0=0.5;

% Calculate mesomixed and micromixed volume fraction:

XMe=zeros(n,1);
XMi=XMe;

for i=1:n
   XMe(i)=XMeso(t(i));
   XMi(i)=XMicro(t(i));
end

%% Graphical output

t=t*1000;
fileSaveName='ResultFig_EngulfmentModel.eps'
plot(t,XMe,'-b',t,XMi,'--k','lineWidth',2)
hold on
plot([tmeso,tmeso]*1000,[0 1],'-.k',[1000/E, 1000/E],[0 1], ...
    ':k','lineWidth',1)
legend('X_{Meso}','X_{Micro}','t_{meso}','t_{micro}=1/E', ...
    'Location','SouthEast')
set(gca,'FontSize',FontSize*1.33);
xlabel('Time [ms]');
ylabel('Volume fraction');
axis([min(t), max(t),0, 1]);
set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);
```

VI. Solution for the Coupled Model

EngulfMixedCM.m

```
%% Precipitation + Engulfment mixing Model, PBE solution with classes
method
% (c) by Andreas Eitzlmayr

clear

global z1
global z2
global cequ
global D1
global D2
global rh1
global rh2
global cS1
global cS2
global S
global rhoS

global k
global NA
global K
global c10
global c20
global T
global etaW
global Vm
global sig
global cS
global nd
global Spezies
global c2pzc
global A

global ClassesNum
global L
global Lm
global GrowthConst
global MP

global e
global eps0
global epsr
global fid
global CollFactor
global c2Excess
global c1End

global tmeso
global E
global X0
global Conversion

Spezies=0;
```

```
% 0 ... Polyacrylic acid + Protamine
% 1 ... Bariumsulfate

Calculate=1;
% 0 ... Don'T solve the sytem (only pre- and postprocessing)
% 1 ... Solve the system (prec. + solving + postprocessing)

%% Input:

% Constants:

k=1.380650424e-23;      % [J/K] Boltzmann's constant
NA=6.0221417930e23;       % [1/mol] Avogadro's constant
e=1.602e-19;    % [As] elementary charge
eps0=8.854e-12;         % [As/Vm] electric constant
epsr=80;                 % [] relative permittivity

A=1e-20;    % Hamaker constant [J]

K=0.39;   % Interfacial energy constant
           % according to Mersmann 2000 between 0.310 and 0.414

Sh=2;      % particle sherwood number for molecular growth

rhoW=997.77;  % density of water [kg/m³]

% Spezies specific parameters:
switch Spezies
    case 0 % Polyacrylic acid + Protamine

        name1='Polyacrylic acid cysteine';
        name2='Protamine';
        M1=5400;                % [g/mol] Molar masses
        M2=4300;
        z1=-56.99;                 % [] molecular charge numbers
        z2=20.72;
        rh1=1.40;           % [nm] hydrodynamic radius
        rh2=1.35;           % (corresponding to diffusion coefficient)
        rhoS=1400;              % [kg/m³] Solid density
        etaW=0.001;             % [Pa s] dynamic viscosity

        cequ=1e-10;             % [mol/l] Equilibium concentration

        B_Desired=1e17;     % [1/m³s] Desired nucleation rate to adjust K

        % Initial concentrations:
        cm10=0.2;     % [g/l] Polyacrylic acid
        cm20=0.6;     % [g/l] Protamine
        c2pzc=7e-5;       %[mol/l] concentr. of Potential determining ions

        % Mixing volume ratio V1/V2:
        VolRatio=1;
    case 1 % Bariumsulfate

        name1='Barium';
        name2='Sulfate';
        M1=137.3;             % [g/mol] Molar masses
        M2=96.1;
        z1=2;                 % [] molecular charge numbers
        z2=-2;
        rh1=0.44482;          % [nm] hydrodynamic radius
```

```
        rh2=0.44482;            % (corresponding to diffusion coefficient)
        rhoS=4500;              % [kg/m³] Solid density
        etaW=0.001;             % [Pa s] dynamic viscosity

        cequ=sqrt(1.01e-4)*1e-3;  % [mol/l] Equilibium concentration

        B_Desired=1.18e11;      % [1/m³s] Desired nucleation rate to adjust
K

        % Initial concentrations:
        cm10=0.1373;            %[g/l] Polyacrylic acid
        cm20=0.0961;            %[g/l] Protamine

        % Mixing volume ratio V1/V2:
        VolRatio=1;
end

% Mixing parameters:
InletDia=0.005;  % [m] inlet diameter
EpsTurb=1;  % [W/kg] Mean power input

% Integration parameters:
LongTime=1;
if LongTime==0
    tmax=0.6;
    timeStep=0.001; % [s]
    SampleTime=[0.001 0.01 0.03 0.1 0.2 0.3 0.4 0.6]; % [s]
else
    tmax=2;
    timeStep=0.001; % [s]
    SampleTime=[0.1 0.2 0.4 0.6 1 1.4 1.8 2]; % [s]
end

% Conditions:
T=295.15;                       % [K] Temperature

% Graphic format:
FontSize=18;
lineWidth=2;

%% Initialize Logfile

LogFile='Results/LogEngulfMixed.txt';

fid=fopen(LogFile,'w');
fclose(fid);
fid=fopen(LogFile,'a');


cequPot=fix(log10(cequ));
c2Pot=fix(log10(c2pzc))-1;
APot=fix(log10(A));

DataName=['Results/Engulf_K0_',num2str(round(10000*K)), ...
    'cEqu',num2str(round(cequ/10^cequPot)),'e',num2str(cequPot), ...
    'c2pzc',num2str(round(c2pzc/10^c2Pot)),'e',num2str(c2Pot), ...
    'A',num2str(round(A/10^APot)),'e',num2str(APot), ...
    'cm10_0_', num2str(round(100*cm10)), ...
    'cm20_0_', num2str(round(100*cm20)), ...
    'd_',num2str(1e4*InletDia),'e-4EPS_',num2str(EpsTurb*100), ...
    'e-2.mat'];
```

```
disp(DataName);
%% Calculation of parameters:

%Diffusion coefficients [m²/s]:
D1=k*T/(6*pi*etaW*rh1*1e-9);
D2=k*T/(6*pi*etaW*rh2*1e-9);

% Initial molar concentrations [mol/l]:
c10=cm10/M1;
c20=cm20/M2;

%Excess concentration of protamine [mol/l]:
c2Excess=c20+c10*VolRatio*z1/z2;
c2End=c2Excess/(VolRatio+1);

%Remaining concentration of PAC [mol/l]:
c1End=cequ^2/c2End;

% Solid concentrations [mol/l]:
% (due to electrical neutrality)
cS1=z2*rhoS/(z2*M1-z1*M2); % Polyacrylic acid (1)
cS2=-cS1*z1/z2;            % Protamine (2)
nd=2;%1-z1/z2;               % Dissociation number
cS=cS1+cS2;           % Total

% Mean solid molecular volume:
Vm=1/(1000*NA*cS);    %[m³]

% Interfacial energy:
sig=K*k*T*(1000*cS*NA)^(2/3)*log(cS/cequ); %[J/m²]

% Growth constant:
GrowthConst=2*Sh*cequ/cS; % [-]

% Mean molecular weight of particles:
MP = (M1*cS1+M2*cS2)/(cS1+cS2); % [g/mol]

% Dimensioned part of collision kernel:
CollFactor=2*k*T/(3*etaW); % [m³/s]

% Mixing parameters:
tmicro=12.7*sqrt(etaW/(rhoW*EpsTurb));  % [s] Micromixing time for E-model
E=1/tmicro;                             % [1/s] Engulfment constant
tmeso=2*(InletDia^2/EpsTurb)^(1/3);     % [s] Mesomixing time for E-model
X0=1/(VolRatio+1);                      % Initial value for Xmicro and
Xmeso

disp(['Mesomixing time for engulfment model:       ', ...
    num2str(tmeso), ' s'])
disp(['Micromixing time for engulfment model:      ', ...
    num2str(tmicro), ' s'])

%% Calculate Timescales:

CharLength=InletDia;   % characteristic length of the reactor [m]
Dmean=(D1+D2)/2;       % mean diffusion coefficient [m²/s]
Sc=etaW/(rhoW*Dmean);  % Schmidt-Number
Nexpected=3.3e17;      % Expected final particle number [1/m³]
Lgrowth=50e-9;         % mean particle size for growth timemscale [m]
SizeRatio=0.1;         % Size ratio for collision timescale
Tmacro=5*(CharLength^2/EpsTurb)^(1/3);
```

```
Tmicro=5*log(Sc)*sqrt(etaW/(rhoW*EpsTurb));
Tnuc=Nexpected/NucleationRate(Dmean,sqrt(c10*c20),cequ,sig,T,Vm,nd);
Tgrowth=1/(2*Sh*Dmean*Lgrowth*pi*Nexpected);
Tcoll=1/(Collisionkernel(SizeRatio)*Nexpected);

% Print timescales
disp(['Timescale for Macromixing:        ', num2str(Tmacro), ' s'])
disp(['Timescale for Micromixing:        ', num2str(Tmicro), ' s'])
disp(['Timescale for Nucleation:         ', num2str(Tnuc), ' s'])
disp(['Timescale for Growth:             ', num2str(Tgrowth), ' s'])
disp(['Timescale for Collisions:         ', num2str(Tcoll), ' s'])

%% Solve System

% Discretisation of internal coordinate (size classes):
ClassesNum=40;  % Number of classes
Lmin=2;    %[nm]
Lmax=300;  %[nm]
  % upper and lower sizes of classes
L=ones(1,ClassesNum+1)*Lmin*1e-9 + ...
    linspace(0,1,ClassesNum+1).^2*(Lmax-Lmin)*1e-9;

Lm=zeros(1,ClassesNum);   % mean sizes of classes

Lm(1)=(L(2)+L(1))/2; % [m]

for i=3:ClassesNum+1
    Lm(i-1)=(L(i)+L(i-1))/2;
end

if Calculate==1
    % Precalculate Aggregation Efficiency (for Interpolation):
    CalcAggEfficiency;
    %load('AggregationTable.mat')
    %Conversion=[0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1];
end

disp('Solving ODE-System ...')

% Initial conditions:

y0=zeros(1,ClassesNum+2);
y0(ClassesNum+1)=0;
y0(ClassesNum+2)=c20;


if Calculate==1
    % Solve ODE System:
    options = odeset('RelTol',1e-2);
    %,'AbsTol',ones(1,ClassesNum+2)*1e-1
    [t,y]=ode45('OdeEngulfMixedCM',0:timeStep:tmax,y0,options);
    data=[t,y];
    save(DataName,'data','-mat')
else
    load(DataName)
    t=data(:,1);
    y=data(:,2:size(data,2));
end

% Calculate Supersaturation, critical Radius:
[m,n]=size(y);
```

```
S=zeros(m,1);
rC=zeros(m,1);

% factor to prevent numerical inaccuracies in Supersaturation:
SupFac=1;

for i=1:m
    for j=1:ClassesNum
        if y(i,j)<0
            y(i,j)=0;
        end
    end
    S(i)=sqrt(y(i,ClassesNum+1)*y(i,ClassesNum+2))/cequ;
    S(i)=real(S(i)); % complex numbers occur due to inaccuracy
    if S(i)<1   % Solution not exact, make correction:
        S(i)=1;
        y(i,ClassesNum+1)=cequ^2/y(i,ClassesNum+2);
    end

    if i>1 && S(i-1)-S(i)>0 && S(i)==1
        SupFac=0; % For all later timesteps set S=1
    end

    S(i)=(S(i)-1)*SupFac+1;

    rC(i)=2*sig*Vm/(nd*k*T*log(S(i)))*1e9; % [nm]


    % Convert PSD from q0 to q3:
    y3Sum=0;

    for j=1:ClassesNum
        y3(i,j)=y(i,j)*Lm(j)^3;
        y3Sum=y3Sum+y3(i,j)*(L(j+1)-L(j));
    end

    for j=1:ClassesNum
        y3(i,j)=y3(i,j)/y3Sum;
    end

end

% Calculate PSD Mean and Variance:

ns=size(SampleTime,2);
Lmean_q3=zeros(ns,1);
LSigma_q3=zeros(ns,1);
Lmean_n=zeros(ns,1);
LSigma_n=zeros(ns,1);
nSum=zeros(ns,1);

n=zeros(ns,ClassesNum); % number density distribution
q3=zeros(ns,ClassesNum); % Volume distribution

for j=1:ns
    Idx=1+int16(SampleTime(j)/timeStep);

    % Separate Samples
    for i=1:ClassesNum
        n(j,i)=y(Idx,i);
        q3(j,i)=y3(Idx,i);
```

```
    end

    % Calculate PSD Mean Values:
    for i=1:ClassesNum
        Lmean_q3(j)=Lmean_q3(j)+q3(j,i)*Lm(i)*(L(i+1)-L(i));
        Lmean_n(j)=Lmean_n(j)+n(j,i)*Lm(i)*(L(i+1)-L(i));
        nSum(j)=nSum(j)+n(j,i)*(L(i+1)-L(i));
    end
    Lmean_n(j)=Lmean_n(j)/nSum(j);

    % Calculate PSD Variances:
    for i=1:ClassesNum
        LSigma_q3(j)=LSigma_q3(j)+(Lm(i)-Lmean_q3(j))^2*q3(j,i)* ...
            (L(i+1)-L(i));
        LSigma_n(j)=LSigma_n(j)+(Lm(i)-Lmean_n(j))^2*n(j,i)* ...
            (L(i+1)-L(i));
    end
    LSigma_q3(j)=sqrt(LSigma_q3(j));
    LSigma_n(j)=sqrt(LSigma_n(j)/nSum(j));

    % Convert to nm and round:
    Lmean_n(j)=0.1*round(Lmean_n(j)*1e10);
    Lmean_q3(j)=0.1*round(Lmean_q3(j)*1e10);
    LSigma_n(j)=0.1*round(LSigma_n(j)*1e10);
    LSigma_q3(j)=0.1*round(LSigma_q3(j)*1e10);
end

% Select data for 3D Plot:
num=101;
n3D=zeros(num,ClassesNum);
q33D=zeros(num,ClassesNum);
t3D=zeros(num,1);

for i=1:num
    Idx=fix((m-1)/num*i)+1;
    n3D(i,:)=y(Idx,1:ClassesNum);
    q33D(i,:)=y3(Idx,1:ClassesNum);
    t3D(i)=t(Idx)*1000;
end

fclose(fid);

%% Graphical Output:

% PSD:

for j=1:ns

    % Plot single PSD n (one point of time):
    plot(Lm*1e9,n(j,:),'LineWidth',lineWidth);

    fileSaveName=['Results/FigEngmixedPSDn',num2str(j),'.eps'];
    set(gca,'FontSize',FontSize);
    xlabel('Particle size L [nm]');
    ylabel('Number density distribution n [1/m^{4}]');
    title(['t=',num2str(SampleTime(j)*1000),'ms, L_{mean}=', ...
        num2str(Lmean_n(j)),'nm, s=',num2str(LSigma_n(j)),'nm']);

    Pos = get(gcf,'Position');
    set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.2 Pos(4)]);
    set(gcf,'PaperpositionMode','auto');
```

```
    print(gcf,'-depsc','-r250',fileSaveName)
    disp(['Saved as: ', fileSaveName]);
    close

    % Plot single PSD q3 (one point of time):
    plot(Lm*1e9,q3(j,:),'LineWidth',lineWidth);

    fileSaveName=['Results/FigEngmixedPSDq3',num2str(j),'.eps'];
    set(gca,'FontSize',FontSize);
    xlabel('Particle size L [nm]');
    ylabel('Volume distribution q_{3} [1/m]');
    title(['t=',num2str(SampleTime(j)*1000),'ms, L_{mean}=', ...
        num2str(Lmean_q3(j)),'nm, s=',num2str(LSigma_q3(j)),'nm']);

    Pos = get(gcf,'Position');
    set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.2 Pos(4)]);

    set(gcf,'PaperpositionMode','auto');
    print(gcf,'-depsc','-r250',fileSaveName)
    disp(['Saved as: ', fileSaveName]);
    close
end

% Plot final q3 PSD logarithmical:
semilogx(Lm*1e9,q3(ns,:),'LineWidth',lineWidth);

fileSaveName=['Results/FigEngmixedPSDfinal_log.eps'];
set(gca,'FontSize',FontSize);
xlabel('Particle size L [nm]');
ylabel('Volume distribution q_{3} [1/m]');
axis([10 1000 0 2e7]);
title(['t=',num2str(SampleTime(ns)*1000),'ms, L_{mean}=', ...
    num2str(Lmean_q3(ns)),'nm, s=',num2str(LSigma_q3(ns)),'nm']);

Pos = get(gcf,'Position');
set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.2 Pos(4)]);

set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);
close

% Plot single PSD q3 (one point of time):
LineFormat=char('-ro','-b^','-k*','-mx','-rd','-bs','-k+','-m');
lineWidth=[0.6 0.6 0.6 0.6 0.6 0.6 0.6 0.6];

% Plot all n PSDs in 1 plot:

plot(Lm*1e9,n(1,:),LineFormat(1,:),'LineWidth',lineWidth(1));
hold on

for i=2:ns
    plot(Lm*1e9,n(i,:),LineFormat(i,:),'LineWidth',lineWidth(i));
end

hold off

fileSaveName=['Results/FigEngmixedPSDn_all.eps'];
set(gca,'FontSize',FontSize);
xlabel('Particle size L [nm]');
ylabel('Number density distribution n [1/m^{4}]');
```

```
legend(['t = ',num2str(SampleTime(1)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(2)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(3)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(4)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(5)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(6)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(7)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(8)*1000),' ms'], ...
    'location','EastOutside')

Pos = get(gcf,'Position');
set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.6 Pos(4)]);

set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);
close


% All q3 PSDs in one plot:
plot(Lm*1e9,q3(1,:),LineFormat(1,:),'LineWidth',lineWidth(1));
hold on

for i=2:ns
    plot(Lm*1e9,q3(i,:),LineFormat(i,:),'LineWidth',lineWidth(i));
end

hold off

lineWidth=2;

fileSaveName=['Results/FigEngmixedPSDq3_all.eps'];
set(gca,'FontSize',FontSize);
xlabel('Particle size L [nm]');
ylabel('Volume distribution q_{3} [1/m]');
legend(['t = ',num2str(SampleTime(1)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(2)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(3)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(4)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(5)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(6)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(7)*1000),' ms'], ...
    ['t = ',num2str(SampleTime(8)*1000),' ms'], ...
    'location','EastOutside')

Pos = get(gcf,'Position');
set(gcf,'Position',[Pos(1) Pos(2) Pos(3)*1.6 Pos(4)]);

set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);
close


% 3D Plot of n PSDs:
surf(Lm*1e9,t3D,n3D(:,1:ClassesNum));
colormap('Jet');
view([1,-5,10]);

fileSaveName=['Results/FigEngmixed_3D_PSDn.eps'];
set(gca,'FontSize',FontSize);
```

```
xlabel('Particle size L [nm]');
ylabel('Time [ms]');
title('Number density distribution n [1/m^{4}]');
set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);


% 3D Plot of p6 PSDs:
surf(Lm*1e9,t3D,q33D(:,1:ClassesNum));
view([1,-5,10]);

load MyColormap;
set(gcf,'Colormap',MyColorMap)

fileSaveName=['Results/FigEngmixed_3D_PSDq3.eps'];
set(gca,'FontSize',FontSize);
xlabel('Particle size L [nm]');
ylabel('Time [ms]');
title('Volume distribution q_{3} [1/m]');
set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);


% Concentrations:
t=t*1000;

fileSaveName='Results/FigEngmixed_Concentrations.eps';
plot(t,y(:,ClassesNum+1),t,y(:,ClassesNum+2),'LineWidth',lineWidth);
legend('c_{1}','c_{2}');
set(gca,'FontSize',FontSize*1.33);
xlabel('Time [ms]');
ylabel('Concentrations [mol/l]');
%axis([min(t), max(t),0, max(y(:,ClassesNum+2))]);
set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);

ConcEngCM=[t,y(:,ClassesNum+1),y(:,ClassesNum+2)];
save('ConcEngCM.mat','ConcEngCM','-mat')


% Supersaturation:
fileSaveName='Results/FigEngmixed_Supersat.eps';
semilogy(t,S,'LineWidth',lineWidth);
set(gca,'FontSize',FontSize*1.33);
xlabel('Time [ms]');
ylabel('Supersaturation');
axis([min(t), max(t), 1, 10^(fix(log10(max(S))/2)*2+2)]);
set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);

SupEngCM=[t,S];
save('SupEngCM.mat','SupEngCM','-mat')

Matrix1=[Lm,Lmean_q3(ns)];
Matrix2=[q3(ns,:),LSigma_q3(ns)];
FinalQ3EngM=[Matrix1;Matrix2];
save('FinalQ3EngM.mat','FinalQ3EngM','-mat')
```

```
% Critical Radius:
fileSaveName='Results/FigEngmixed_CriticalRadius.eps';
plot(t,rC,'LineWidth',lineWidth);
axis([min(t), max(t), 0,max(rC)]);
set(gca,'FontSize',FontSize*1.33);
xlabel('Time [ms]');
ylabel('Critical Radius [nm]');
set(gcf,'PaperpositionMode','auto');
print(gcf,'-depsc','-r250',fileSaveName)
disp(['Saved as: ', fileSaveName]);


%% Output:
% Text:

disp(' ')
disp('************************************************************')
disp(' ')

disp('INPUT:')

disp([name1,' (1):'])
disp(['Molar mass:                      ', num2str(M1), ' g/mol'])
disp(['Electrical charge number:        ', num2str(z1)])
disp(['hydrodynamic radius:             ', num2str(rh1), ' nm'])
disp(['Initial mass concentration:      ', num2str(cm10), ' g/l'])
disp(' ')

disp([name2,' (2):'])
disp(['Molar mass:                      ', num2str(M2), ' g/mol'])
disp(['Electrical charge number:        ', num2str(z2)])
disp(['hydrodynamic radius:             ', num2str(rh2), ' nm'])
disp(['Initial mass concentration:      ', num2str(cm20), ' g/l'])
disp(' ')

disp(['Interfacial energy constant:     ', num2str(K)])
disp(['Dissociation number:             ', num2str(nd)])
disp(['Solid density:                   ', num2str(rhoS), ' kg/m³'])
disp(['Dynamic fluid viscosity:         ', num2str(etaW), ' Pa s'])
disp(['Solubility product:              ', num2str(cequ^2), ...
    ' (mol/l)²'])
disp(['Temperature:                     ', num2str(T), ' K'])
disp(' ')
disp(' ')

disp('OUTPUT:')

disp(['Diffusion coefficient 1:         ', num2str(D1), ' m²/s'])
disp(['Diffusion coefficient 2:         ', num2str(D2), ' m²/s'])
disp(['Initial concentration 1:         ', num2str(c10), ' mol/l'])
disp(['Initial concentration 2:         ', num2str(c20), ' mol/l'])
disp(['Solid concentration 1:           ', num2str(cS1), ' mol/l'])
disp(['Solid concentration 2:           ', num2str(cS2), ' mol/l'])
disp(['Total solid concentration:       ', num2str(cS), ' mol/l'])
disp(['Mean solid molecular volume:     ', num2str(Vm), ' m³'])
disp(['Equilibrium concentration:       ', num2str(cequ), ' mol/l'])
disp(['Interfacial energy:              ', num2str(sig), ' J/m²'])
```

OdeEngulfMixedCM.m

```
function dy=OdeEngulfMixedCM(t,y)
% ODE-System for Nucleation + Growth + Aggregation with Engulfment mixing

global cequ
global D1
global D2
global cS1
global c10
global c20
global k
global T
global Vm
global sig
global nd
global z1
global z2

global ClassesNum
global L
global Lm
global GrowthConst
global fid
global Conversion
global AggregationTable
global E
global c2Excess
global X0

dy=zeros(ClassesNum+2,1);

% y(1 - ClassesNum) ... particle number density of every class
% y(ClassesNum+1) ... Concentration of Polyacrylic acid [mol/l]
% y(ClassesNum+2) ... Concentration of Protamine [mol/l]

for i=1:ClassesNum+2
    %y(i)=real(y(i));
    if y(i)<0
        y(i)=0;
    end
end

% Mean concentration:
c=sqrt(y(ClassesNum+1)*y(ClassesNum+2)); %[mol/l]

% Supersaturation:
S=c/cequ; %[-]

if S<=1
    S=1;
    y(ClassesNum+1)=cequ^2/y(ClassesNum+2);
    c=cequ;
end

% Critical radius:
rc=2*sig*Vm/(nd*k*T*log(S+1e-10));

% Mean diffusion coefficient:
D = (D1*y(ClassesNum+1) + D2*y(ClassesNum+2)) / ...
```

```
    (y(ClassesNum+1) + y(ClassesNum+2));

% Nucleation rate:

Bhom = NucleationRate(D,c,cequ,sig,T,Vm,nd);

% Write data to Log File:
fprintf(fid,'t = %2.8f    ',t);


% Evaluate Aggregation Kernel:
valueAggKernel=zeros(ClassesNum,ClassesNum);

%X=1-y(ClassesNum+1)/c10;
X=1-(y(ClassesNum+2)*XMicro(t)/X0-c2Excess)/(c20-c2Excess); % Actual
conversion

if X<0 % repair inaccuracies;
    X=0;
elseif X>1
    X=1;
end

% determine Conversion Index:
if X==1
    CIdx=size(Conversion,2);
    IntFac=0;
else
    i=1;
    while  X>=Conversion(i)
         if X<Conversion(i+1)
             % Conversion Index:
             CIdx=i;
             % Interpolation factor:
             IntFac=(X-Conversion(i))/(Conversion(i+1)-Conversion(i));
         end
         i=i+1;
    end
end

for i=1:ClassesNum
    for j=1:i
        if X==1
            valueAggKernel(i,j)=AggregationTable(i,j,CIdx);
        else
            valueAggKernel(i,j)=AggregationTable(i,j,CIdx)*(1-IntFac)+...
                AggregationTable(i,j,CIdx+1)*IntFac;
        end
        valueAggKernel(j,i)=valueAggKernel(i,j);
    end
end

TotalBirth=0;
TotalDeath=0;
ySum=0;

% Aggregation:

AggBirth=zeros(ClassesNum,1);
AggDeath=zeros(ClassesNum,1);
```

```
for i=1:ClassesNum
    ySum = ySum + y(i) * (L(i+1)-L(i));

    for p=1:ClassesNum
        CollisionRate = valueAggKernel(i,p) * y(p) * y(i) * ...
            (L(p+1)-L(p)) * (L(i+1)-L(i)); % [1/m³s]

        AggDeath(i) = AggDeath(i) + CollisionRate / (L(i+1)-L(i));
        TotalDeath = TotalDeath + CollisionRate;

        Lresult=(Lm(i)^3+Lm(p)^3)^(1/3);  % Size of the resulting particle

        if Lresult<=Lm(1)
            l=1;
            AggBirth(l) = AggBirth(l) +  0.5 * CollisionRate / ...
                (L(l+1)-L(l));
            TotalBirth = TotalBirth + CollisionRate/2;

        elseif Lresult>Lm(ClassesNum)
            l=ClassesNum;
            AggBirth(l) = AggBirth(l) +  0.5 * CollisionRate / ...
                (L(l+1)-L(l));
            TotalBirth = TotalBirth + CollisionRate/2;
        else
            j=1;
            while Lresult>Lm(j)   % Is Lresult in this class j ?
                if Lresult<=Lm(j+1)
                    l=j; % Resulting particle is between Lm(l) and Lm(l+1)

   % Resulting particle is partitioned between 2 classes Lm(l) and Lm(l+1):
   % (due to particle number conservation and mass conservation)

                    w1 = (1 - (Lresult/Lm(l+1))^3) / ...
                        (1 - (Lm(l)/Lm(l+1))^3);
                    w2 = 1 - w1;

                    AggBirth(l) = AggBirth(l) +  w1/2 * ...
                        CollisionRate / (L(l+1)-L(l));
                    AggBirth(l+1) = AggBirth(l+1) + w2/2 * ...
                        CollisionRate / (L(l+2)-L(l+1));
                    TotalBirth = TotalBirth + CollisionRate/2;
                end
                j=j+1;
            end
        end

    end

end

if TotalDeath==TotalBirth*2
    % OK
else
    disp('Collision Number conservation Error')
    disp('TotalDeath = ',num2str(TotalDeath))
    disp('TotalBirth x 2 = ',num2str(TotalBirth*2))
end

% Mixing-term:
MixVolIncrease = E*(1-XMicro(t)/XMeso(t));
```

```
% ODE's:
% Particle classes
for i=1:ClassesNum

    %Nucleation:

    Nuc=0;  % Set Nucleation to zero (default)
    if 2*rc>=L(i)  % Is critical nuclei size within this class?
        if 2*rc<L(i+1)
            Nuc = Bhom/(L(i+1)-L(i)); % Set Nucelation term
        end
    end

    % Growth (Upwind Discretisation)

    if i==1 % lower boundary of internal coordinate L
        Growth=0;
    elseif i==ClassesNum % upper boundary of internal coordinate L
        Growth=GrowthConst*D*(S-1)*(y(i)/Lm(i) - y(i-1)/Lm(i-1))/ ...
            (Lm(i)-Lm(i-1));
    else
        Growth=GrowthConst*D*(S-1)*(y(i)/Lm(i) - y(i-1)/Lm(i-1))/ ...
            (Lm(i)-Lm(i-1));
    end

    % Balance:

    dy(i) = Nuc - Growth + AggBirth(i) - AggDeath(i) - ...
        MixVolIncrease*y(i);

end


GrowthSum=0;

for i=1:ClassesNum
    GrowthSum = GrowthSum + Lm(i)*y(i)*(L(i+1)-L(i)); % [1/m²]
end

GrowthSum = GrowthSum*GrowthConst*D*(S-1)*pi*cS1/2; % [kmol/m³s]

% Polyacrylic acid concentration
dc1WellMixed = -Bhom * 4*pi/3*rc^3*cS1 - GrowthSum;
dy(ClassesNum+1) = dc1WellMixed + MixVolIncrease*(c10-y(ClassesNum+1));

% Protamine concentration
dy(ClassesNum+2) = dc1WellMixed*(-z1/z2) + MixVolIncrease* ...
    (0-y(ClassesNum+2));

% Write data to Log File:
fprintf(fid,'  Sup %6.3f  ',S);
% dc1/dt due to nucleation
fprintf(fid,'  Nuc %1.10f  ',-Bhom * 4*pi/3*rc^3*cS1);
 % dc1/dt due to growth
fprintf(fid,'  Growth %1.10f  ',-GrowthSum/(1-z1/z2));
% born Particles per second and sum of particles
fprintf(fid,'  Agg.TotalBirth %3.10f  ',TotalBirth/ySum);

% Write data to Log File:
fprintf(fid,'%1.0f **\n',0);
```