NTNU
Norwegian University of
Science and Technology

Modeling Interfacial Mass Transfer during the Dynamic Evolution of the Drop Size Density Distribution of Emulsions in Turbulent Pipe Flow

Torstein Lædre

Chemical Engineering and Biotechnology
Submission date: June 2016
Supervisor: Brian Arthur Grimes, IKP

Norwegian University of Science and Technology
Department of Chemical Engineering

Preface

This thesis is submitted in partial fulfilment of the requirements for the degree of Master of Science(M.Sc.) at the Norwegian University of Science and Technology(NTNU). The work was carried out in the chemical engineering department of NTNU, supervised by associate professor Brian A. Grimes.

Acknowledgements

I would like to acknowledge the following people for their invaluable contributions, directly or indirectly, to this thesis.

First i want to express my deepest gratitude towards Brian Grimes, my thesis adviser, for his patience with my young and restless programming soul. Thank you for teaching me so much in such a short time and with such pedagogical excellence.

I am also very thankful for the support of friends and family when, on several occasions, i have been demotivated or just feeling blue.

Abstract

This thesis is focused on modeling the dynamic evolution of the drop size density distribution of emulsions in turbulent pipe flow with and without the mass balance of the surfactant. First, the interplay between coalescence and breakage was studied by changing the fitting parameters of coalescence- and breakage frequency and the fitting parameters of coalescence- and breakage efficiency. Lastly, an attempt was made to include the surfactant mass balance in the model for comparison.

The base case showed that coalescence is stronger at shorter axial positions in the pipe and grows weaker along the z-axis. Breakage showed the complete opposite trend. $L\infty$ varied greatly with changes in the fitting parameters for coalescence- and breakage frequency and it was showed that when $k_\omega = k_{g1}$, equal changes in magnitude of these fitting parameters does not change the shape of the drop size distribution. However, the shape of the drop size distribution showed great dependence on the fitting parameters for coalescence- and breakage efficiency. The volume density distribution f_v showed the expected alterations in shape when it was studied with different sets of fitting parameters. When the mass balance of the surfactant was included in the model, few or no changes were seen in the average droplet radius of the volume distribution.

Overall, the model without the mass balance worked well, showing the appropriate results when changes were made to key parameters. Larger variations in the shape of the density distribution was expected when the mass balance was included but, these changes failed to appear. It was concluded that the lumping of the retarded Hamaker constant into the fitting parameter for coalescence efficiency was insufficient to describe the dynamics of the intermolecular forces working on the surface of droplets.

Sammendrag

Denne oppgaven har fokusert på å modelllere den dynamiske utviklingen av emulsjoners dråpestørrelsesfordeling i turbulent rørstrømning med og uten massebalansen for tensider. Først, ble smaspillet mellom koalesens og brytning av dråper studert ved å endre tilpasningsparametrene for koalesens- og brytningsfrekvens og tilpasningsparametrene for koalesens- og brytningseffektivitet. Til slutt, ble det gjort ett forsøk på å inkludere massebalansen for tensider i modellen.

Den grunnleggende studien viste at koalesens er sterkere ved kortere axielle posisjoner i røret og ble svakere og svakere langs z-aksen. Brytning viste eksakt motsatt trend. L_∞ varierte betydelig med endringer i tilpasningsparametrene for koalesens- og brytningsfrekvens. Det ble vist at når $k_\omega = k_{g1}$ har endring i disse parametrene av like stor størrelse ikke noe å si for fasongen til dråpestørrelsesdistribusjonen. Formen til dråpestørrelsesdistribusjonen viste stor varians når det ble gjort endringer i tilpasningsparametrene for koalesens- og brytningseffektivitet. Volumtetthetsdistribusjonen, f_v, endret form som forventet når foskjellige sett med tilpasningsparametre ble undersøkt. Når massebalansen for tensider ble inkludert i modellen ble det observert få endringer i den gjennomsnittlige dråpestørrelsen til volumtetthetsdistribusjonen.

Alt i alt fungerte modellen bra uten massebalansen og viste passende resultater når det ble gjort endringer i nøkkelparametre. Det var forventet større endringer i formen til tetthetsdistribusjonen når massebalansen for tensider ble inkludert i modellen, men disse endringene uteble. Det ble konkludert med at inklusjonen av den forsinkede Hamakerkonstanten i tilpasningsparameteren for koalesenseffektivitet ikke var tilstrekkelig for å beskrive dynamikken i de intermolekylære kreftene som virker på overflaten til dråpene.

Contents

List of Tables

List of Figures

List of Symbols

A_d - Area of the dispersed phase droplets [m^2]

a - Weighting parameter for Jakobi orthogonal polynomial

b - Weighting parameter for Jakobi orthogonal polynomial

C - Overall surfactant concentration $\left[\frac{\mathrm{mol}}{\mathrm{m}^3}\right]$

$C_{f,0}$ - Initial concentration of surfactant in the continuous phase $\left[\frac{\mathrm{mol}}{\mathrm{m}^3}\right]$

$C_{f,\infty}$ - Concentration of surfactant in the continuous phase at infinite dilution $\left[\frac{\mathrm{mol}}{\mathrm{m}^3}\right]$

C^* - Surfactant concentration at boundary layer $\left[\frac{\mathrm{mol}}{\mathrm{m}^3}\right]$

D - Diameter of the pipe [m]

E - Gathering of constants in the mass balance of surfactant equation

F - Gathering of constants in the surfactant mass balance equation

f - Truncated orthonormal polynomial series

f_n - Dimensionless number distribution

f_v - Dimensionless volume distribution

g - Rate of breakage $\left[\frac{1}{\mathrm{s}}\right]$

$\overline{g}$ - Dimensionless rate of breakage

h - Initial condition of truncated orthonormal polynomial series

$J_p^{a,b}$ - Jakobi orthogonal polynomial of degree p

K_f - Mass transfer coefficient $\left[\frac{\mathrm{m}}{\mathrm{s}}\right]$

K - Langmuir isotherm equilibrium constant $\left[\frac{\mathrm{m}^3}{\mathrm{mol}}\right]$

k_ψ - Proportionality constant for the coalescence probability [m^2]

k_ω - Proportionality constant for the coalescence probability

k_{g1} - Proportionality constant for the breakage frequency

k_{g2} - Proportionality constant for exponential term of breakage frequency

L - Length of the pipe [m]

N_d - Number of dispersed phase particles per unit volume $\left[\frac{1}{\mathrm{m}^3}\right]$

N_s - Amount of surfactant adsorbed on the surface of the dispersed phase [mol]

n_s - Surface normal vector of dispersed phase droplets

$P_{B,+}$ - Dimensionless birth rate due to breakage

$P_{B,-}$ - Dimensionless death rate due to breakage

$P_{C,+}$ - Dimensionless birth rate due to coalescence

$P_{C,-}$ - Dimensionless death rate due to coalescence

R - Gas constant $\left[\frac{\mathrm{m}^3\mathrm{Pa}}{\mathrm{Kmol}}\right]$

$R_{B,+}$ - Birth rate of droplets due to breakage $\left[\frac{1}{\mathrm{m}^3\mathrm{s}}\right]$

$R_{B,-}$ - Death rate of droplets due to breakage $\left[\frac{1}{\mathrm{m}^3\mathrm{s}}\right]$

$R_{C,+}$ - Birth rate of droplets due to coalescence $\left[\frac{1}{\mathrm{m}^3\mathrm{s}}\right]$

$R_{C,-}$ - Death rate of droplets due to coalescence $\left[\frac{1}{\mathrm{m}^3\mathrm{s}}\right]$

Re - Residual of orthonormal polynomial series

R_{max} - The largest droplet radius of the system [m]

r_1 - Radius of one of two drops coalescing [m]

r_2 - Radius of one of two drops coalescing [m]

r - Radius of children droplets created under coalescence or breakage [m]

r' - Radius of one of two parent droplets under coalescence or breakage [m]

r'' - Radius of one of two parent droplets under coalescence or breakage [m]

r_c - Rate of coalescence $\left[\frac{1}{\mathrm{s}}\right]$

$\overline{r}_c$ - Dimensionless rate of coalescence

T - Absolute temperature [K]

t - Time [s]

U - Average velocity of dispersion in the pipe $\left[\frac{\mathrm{m}}{\mathrm{s}}\right]$

V - Total volume of dispersion [m^3]

x - Internal coordinate

z - Axial coordinate of the pipe [m]

Greek letters

α - Surface of dispersed phase ratio to the total volume of dispersion $\left[\frac{1}{\mathrm{m}}\right]$

$\overline{\alpha}$ - Dimensionless surface of dispersed phase ratio to the total volume of dispersion

β - Daughter size distribution $\left[\frac{1}{\mathrm{m}}\right]$

$\overline{\beta}$ - Dimensionless daughter size distribution

Γ - Number of moles of surfactant adsorbed on the dispersed phase interface $\left[\frac{\mathrm{mol}}{\mathrm{m}^2}\right]$

Γ_{max} - Maximum amount of moles that can be adsorbed on the dispersed phase interface $\left[\frac{\mathrm{mol}}{\mathrm{m}^2}\right]$

γ - Interfacial surface tension $\left[\frac{\mathrm{N}}{\mathrm{m}}\right]$

ϵ - Rate of energy dissipation $\left[\frac{\mathrm{m}^2}{\mathrm{s}^3}\right]$

η - Droplet number density distribution $\left[\frac{1}{\mathrm{m}^3\mathrm{m}}\right]$

θ - Dimensionless concentration of the surfactant in the continuous phase

λ - Dimensionless axial position in the pipe

μ_c - Viscosity of the continuous phase [Pa·s]

ξ - Dimensionless radius

π - The ratio between the circumference and radius of a circle

ρ_c - Density of the continuous phase $\left[\frac{\mathrm{kg}}{\mathrm{m}^3}\right]$

ρ_d - Density of the dispersed phase $\left[\frac{\mathrm{kg}}{\mathrm{m}^3}\right]$

τ - Dimensionless amount of moles adsorbed on the dispersed phase interface

ϕ - Volume fraction of droplets(dispersed phase)

ϕ_n - Function that truncated orthogonal polynomial series is expanded in terms of

χ_ψ - Dimensionless proportionality constant in the coalescence probability equation

χ_ω - Dimensionless proportionality constant in the coalescence frequency equation

χ_{g1} - Dimensionless proportionality constant in the breakage frequency equation

χ_{g2} - Dimensionless proportionality constant in the exponent of the breakage frequency equation

ψ_E - Probability of coalescence for two colliding droplets

ψ_n - Weighting function

ω_C - Frequency of droplet collisions $\left[\frac{\mathrm{m}^3}{\mathrm{s}}\right]$

Subscripts

0 - Denotes the initial state of a variable

d - Indicates that the variable is associated with the dispersed phase

f - Indicates that the variable is associated with the continuous phase

Chapter 1

Introduction

Recently, modeling of fully dispersed flow has been given an unprecedented amount of attention. Understanding of how particulate solutions act when they are moving in pipes, reactors etc. have become more important as the number of applications involving dispersions has risen in the last decades. Fully dispersed flow is present in many industrial applications such as bubble columns, oil refinery equipment and pipe flow in general.

The history of fully dispersed flow modeling can be traced back to the models of Brownian motion presented by Einstein[1] and Smoluchowski[2] in the early 20th century. These models were attempts to predict the random motion of particles in a solution based on kinetic theory, equivalent to a dispersion that contains gas bubbles or liquid drops. The equations presented by both Smoluchowski and Einstein are probability distributions of a particle with respect to time and one spatial dimension. In terms of modeling fully dispersed flow, this methodology would only work for monodisperse particles and at lower velocity non-turbulent flow regimes.

In 1938 Landau and Rumer[3] published a paper on the cascade theory of electronic showers. They presented functions that described the number of photons and the number of particles as well as their evolution in spatial coordinates. Even though electronic showers are not comparable with fully dispersed two phase flow, it was one of the earliest formulations of a population balance, which later became a very important tool for modeling fully dispersed two phase flow.

Most of the applications of fully dispersed flow mentioned above involve turbulent flow regimes. In 1955 Saffman and Turner[4] issued a paper on drop collision in turbulent clouds. This paper puts forward a model of how turbulent eddies affect

the efficiency with which drops coalesce and how drops collide. This was a huge step in the direction of a more comprehensive and accurate model of dispersed two phase flow.

In 1964 the generalized population balance equation came along as a modeling tool for fully dispersed two phase flow. Hulburt and Katz[5] published an article in which they explored the use of differential equations to show the variation in average particle size to predict the behaviour of dispersed phase systems. This is the grounds on which the current prevailing models are built. Now, there was a differential equation that could predict the distribution of particle size as a function of time and spatial variables in a more accurate manner. Going forward, the focus of model developers was directed towards finding the best way to incorporate probabilities of collision and efficiency of coalescence or other particle phenomena into the differential equations.

Introducing the population balance into the modeling of fully dispersed two phase flow has not only given more accurate predictions, it has also brought along the challenge of an immense amount of equations having to be solved. The computational efforts required for solving these differential equations increase rapidly with the amount of equations. As the last three decades have seen rapid advances in computer technology, more involved and computationally demanding models have been solved. Computational fluid dynamics(CFD) have also been combined with population balance models to predict the behaviour of two phase fully dispersed flow.

This thesis is focused around building a comprehensive model of fully dispersed flow in Matlab, by combining the population balance equation with the surfactant mass balance. Including the mass balance for surfactants is important because of the effect that surfactants have on phase separation and therefore also on the flow properties of the fluid at hand. By adsorbing on the surface of liquid droplets, surfactants change the surface properties of the droplets and therefore the extent to which they coalesce and break up. Many of the applications of fully dispersed flow involves complex mixtures of components with multiple substances and many different surfactants present. These mixtures commonly occur in the petroleum industry and predictive knowledge of how they act are therefore crucial to efficient oil recovery.

First, the population balance will be solved without adding the surfactant mass balance to the model. This is not only good modeling practice, to solve one part of the model before adding complexity, but it is also crucial to compare the model with and without the surfactant mass transfer. Next, the surfactant mass transfer equation will be incorporated into the model. A comparison will be made between

the model with and without the mass balance equations to evaluate whether a more complex approach to the surfactant mass balance has to be taken.

Chapter 2

Background and Theory

2.1 Population Balance Equation

A pipe of length L and diameter D is considered. Inside the pipe a liquid-liquid dispersion flows in a locally isotropic turbulent flow-field [6].The flow profile is assumed to be plug-flow and the fluid velocity is described by an average velocity U. An illustration of the pipe is shown in Figure 2.1. Any radial variances in droplet sizes are neglected. A density function $\eta(r, z)$ describes the distribution of droplet sizes, where r denotes the radius of droplets and z denotes the axial position in the pipe. r serves as the internal coordinate and z as the external coordinate. The difference is that the external coordinate tells the position of the drop in the pipe, while the internal coordinate relates to the properties of the droplet. Locally, the total number density $N_d(z)$ at a given axial position z is[6]

$$N_d(z) = \int_0^{R_{max}} \eta(r', z)\, \mathrm{d}r' \tag{2.1}$$

In Equation 2.1 R_{max} denotes the upper limit of the radius in the number distribution. The volume fraction, $\phi(z)$, of droplets at any radial position is then given by[6]

$$\phi(z) = \int_0^{R_{max}} \left(\frac{4\pi}{3} r'^3\right) \eta(r', z)\, \mathrm{d}r' \tag{2.2}$$

No droplets are introduced into the pipe at any radial position and therefore $\phi(z)$ should remain constant throughout the length of the pipe.

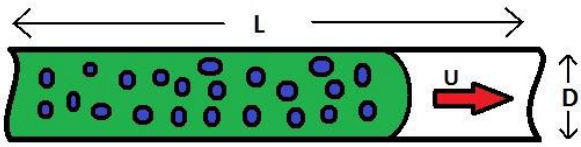

Figure 2.1: Pipe section with length L, diameter D and average fluid velocity, U.

Convection transports the droplets along the axial direction of the pipe, z, and coalescence and breakage alter the size of the droplets in the internal coordinate space, r. Based on the foregoing assumptions and definitions the governing transport equation of the droplets in the pipe is described by the population balance[6]

$$U\frac{\mathrm{d}\eta}{\mathrm{d}z} = R_{C,+} - R_{C,-} + R_{B,+} - R_{B,-}, \quad \text{for} \quad 0 \leq z \leq L, \quad 0 \leq r \leq R_{max} \tag{2.3}$$

$$\text{Initial condition: at} \quad z = 0, \quad \eta(r,0) = \eta_0(r), \quad \text{for} \quad 0 \leq r \leq R_{max} \tag{2.4}$$

where $R_{C,+}$ is the birth rate of a droplet of radius r due to coalescence, $R_{C,-}$ is the death rate of a droplet of radius r due to coalescence, $R_{B,+}$ is the birth rate of a droplet of radius r due to breakage, $R_{B,-}$ is the death rate of a droplet of radius r due to breakage and $\eta_0(r)$ is the droplet size distribution at the initial position in the pipe$(z = 0)$.

2.1.1 Birth- and Death Rate Due to Coalescence

When two parent droplets of radius r' and r'' coalesce, they form a droplet of size r. Due to the conservation of volume we can write $r'' = [r^3 - r'^3]^{1/3}$. The coalescence birth rate as a function of radius r and axial position z is then given by[6]

$$R_{C,+}(r,z) = \int_0^{r/\sqrt[3]{2}} r_C(r', r'')\eta(r', z)\eta(r'', z)\,\mathrm{d}r', \quad \text{for} \quad 0 \leq z \leq L, \quad 0 \leq r \leq R_{max} \tag{2.5}$$

According to the conservation of mass the death of particles due to coalescence also have to be taken into account. The following expression gives the death rate of drops with radius r at the axial position z[6]

$$R_{C,-}(r,z) = \eta(r,z)\int_0^{R_{max}} r_C(r,r')\eta(r',z)\,\mathrm{d}r', \quad \text{for} \quad 0 \le z \le L, \quad 0 \le r \le R_{max} \tag{2.6}$$

The rate of coalescence r_C, found in Equations 2.5 and 2.6, is determined by the constitutive equations of coalescence frequency(or swept volume rate), ω_C, and probability of coalescence, ψ_E, for two colliding drops. The rate of coalescence between two drops of radius r_1 and r_2 is written as follows[6]

$$r_c(r_1,r_2) = \omega_C(r_1,r_2)\psi_E(r_1,r_2) \tag{2.7}$$

One of the most common ways of determining the constitutive equation for collision frequency is to assume that the collision of droplets in a locally isotropic flow field is analogous to the collisions between molecules in kinetic gas-theory[7, 8, 9]. In kinetic gas-theory, the collisions are determined based on the mean square fluctuation velocities of the droplets in the flow field. Neglecting the contribution of eddies in the wave number range beyond the inertial sub-range[7, 8, 9], allows for the mean square fluctuation velocities of the droplets to be expressed in terms of the local energy dissipation rate per unit mass, ϵ, and the droplet volume. The collision frequency of droplets is then expressed as[7, 8, 9]

$$\omega_C(r_1,r_2) = k_\omega \frac{\epsilon^{1/3}}{(1+\phi)}(r_1+r_2)^2[r_1^{2/3}+r_2^{2/3}]^{1/2} \tag{2.8}$$

where k_ω is a proportionality constant that is used as a fitting parameter in the model. As a general estimate, the following expression can be used to find the value of ϵ[10]

$$\epsilon = 0.01766\frac{U^3}{D}\left(\frac{\mu_c}{\rho_c U D}\right)^{3/8} \tag{2.9}$$

where D is the pipe diameter and μ_c and ρ_c are the viscosity and density of the continuous phase, respectively. An estimate of ϵ is sufficient since it is directly multiplied by the fitting parameter, k_ψ.

The probability of coalescence can be expressed as the ratio between the contact time of two droplets and the time it takes before film drainage and rupture between

two drops occur(coalescence time). The coalescence time is determined from a film drainage expression between two deformable drops[7, 8, 9, 11], where the force compressing the droplets is proportional to the mean-square velocity difference of the drops at either end of the eddy. The contact time is considered to be a random variable that is proportional to the characteristic period of velocity fluctuation of an eddy of size $2(r_1 + r_2)$[7, 8, 9, 12]. The efficiency of coalescence is thus written[7, 8, 9, 12]

$$\psi_E(r_1, r_2) = exp\left[-\frac{1}{k_\psi}\frac{\mu_c \rho_c \epsilon}{\gamma_0^2}\left(\frac{r_1 r_2}{r_1 + r_2}\right)^4\right] \tag{2.10}$$

where k_ψ is a proportionality constant that is used as a fitting parameter in the model and γ_0 is the surface tension. k_ψ is lumped in with the difference between the inverse squares of the initial film thickness and film thickness at which film rupture occurs and therefore has units of m^2.

2.1.2 Birth- and Death Rates Due to Breakage

It can be assumed that an oscillating deformed drop will break if the turbulent kinetic energy transmitted to the drop exceeds the surface energy of the droplet. The death rate of a given droplet due to breakage is therefore determined by both local flow characteristics and the breakage frequency $g(r)$. Thus, the death rate of a droplet of radius r can be determined by the product of the number distribution $\eta(r, z)$, and breakage frequency, $g(r)$[6]

$$R_{B,-} = g(r)\eta(r, z), \quad \text{for} \quad 0 \leq z \leq L, \quad 0 \leq r \leq R_{max} \tag{2.11}$$

Again, to account for the conservation of volume an expression for the birth of droplets due to breakage have to be included in the population balance equation. The birth rate due breakage can be determined by integrating $R_{B,-}$ over the interval of drop sizes, r', larger than r ($r \leq r' \leq R_{max}$). To account for the formation of at least two drops under breakage, the breakage death rate is modified by a daughter size distribution function, $\beta(r, r')$, that represents the probability that breakage of a drop of size, r', will form a drop of size, r. The birth rate of droplets due to breakage is given by the following expression[6]

$$R_{B,+}(r, z) = \int_r^{R_{max}} 2\beta(r, r')g(r')\eta(r', z)\,\mathrm{d}r', \quad \text{for} \quad 0 \leq z \leq L, \quad 0 \leq r \leq R_{max} \tag{2.12}$$

The number, two, in the integrand of Equation 2.12 is necessary because the nature of the breakage is binary. Only binary breakage has been taken into account in this model due to the fact that there is no theoretical basis to determine the number of drops formed based on the drop size, the interfacial tension or the forces applied to the drop[7, 8, 9]. In addittion, binary breakage is a good approximation for droplets of diameter less than 1 mm[9].

As mentioned above, the premise for droplet breakage is that the kinetic energy transmitted from an eddy to the droplet exceeds the the surface energy of the droplet. It can also be assumed that the distribution of total kinetic energy of drops are proportional to the distribution of kinetic energy of turbulent eddies. The following equation gives the frequency of breakage[7, 8, 9]

$$g(r) = k_{g1}\frac{\epsilon^{1/3}}{r^{2/3}(1+\phi)}exp\left[-k_{g2}\frac{\gamma_0(1+\phi)^2}{\rho_d\epsilon^{2/3}r^{5/3}}\right] \tag{2.13}$$

where ρ_d is the density of the dispersed phase, and k_{g1} and k_{g2} are proportionality constants that are used as fitting parameters in the model.

A symmetric beta distribution has been employed by Hsia and Tavlarides[9] and Azizi and Taweel[10] to represent the daughter size distribution $\beta(r, r')$. It satisfies several requirements: (1) that there is zero probability for infinitely small daughter droplets; (2) it avoids zero probability for the evolution of equal size drops; (3) the combined size of the daughter droplets always equal the size of the parent droplet. The daughter size distribution representing the probability that a droplet of size, r', will break to form a droplet of size, r, can then be written as[9, 10]

$$\beta(r, r') = \frac{45}{\sqrt[3]{2}}\frac{r^2}{r'^3}\left(\frac{r^3}{r'^3}\right)^2\left[1-\left(\frac{r^3}{r'^3}\right)^2\right] \tag{2.14}$$

A couple of other properties of the number density distribution and volume density distribution are relevant to evaluate the results of the model. The evolution of the mean droplet radii, μ_{aN} and μ_{aV}, based on the number density distribution and the volume density distribution respectively, are given by

$$\mu_{aN}(z) = \frac{1}{N_d(z)}\int_0^{R_{max}} r'\eta(r', z)\mathrm{d}r' \tag{2.15}$$

$$\mu_{aV}(z) = \frac{1}{\phi}\int_0^{R_{max}} r'\left(\frac{4\pi}{3}r'^3\right)\eta(r', z)\mathrm{d}r' \tag{2.16}$$

The standard deviations, σ_{aN} and σ_{aV}, of the droplet number density distribution and the volume density distribution respectively, are given by

$$\sigma_{aN} = \sqrt{\frac{1}{N_d(z)} \int_0^{R_{max}} (r' - \mu_{aN}(z))^2 \eta(r', z) \mathrm{d}r'} \tag{2.17}$$

$$\sigma_{aV} = \sqrt{\frac{1}{\phi} \int_0^{R_{max}} (r' - \mu_{aV}(z))^2 \left(\frac{4\pi}{3} r'^3\right) \eta(r', z) \mathrm{d}r'} \tag{2.18}$$

2.2 The Surfactant Mass Balance

Considering the pipe section shown in Figure 2.1 and general conservation of mass with no net transport through the chosen boundary, the following equation holds true

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_V C \, \mathrm{d}V = 0 \tag{2.19}$$

where $\frac{\mathrm{d}}{\mathrm{d}t}$ is the time derivative and C is the concentration of surfactant within the total volume of the dispersion, V.

If the total volume of the dispersion is divided into volume of dispersed phase, V_d, and volume of continuous phase, V_f, as shown in Figure 2.2 the conservation of mass can be written as

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{V_f} C_f \, \mathrm{d}V_f + \frac{\mathrm{d}}{\mathrm{d}t} \int_{V_d} C_d \, \mathrm{d}V_d = 0 \tag{2.20}$$

where C_f and C_d are the concentrations of the surfactant in the continuous phase and the dispersed phase respectively.

The total amount of surfactant in moles are conserved in the dispersed phase and since the surfactant adsorbs on the surface of the droplets a balance over the dispersed phase can be written

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{V_d} C_d \, \mathrm{d}V_d + \int_{A_d} N_s \cdot n_s \, \mathrm{d}A_d = 0 \tag{2.21}$$

where A_d is the surface area of the dispersed phase, N_s is the amount of surfactant adsorbed at the surface in moles and n_s is the surface normal vector. When

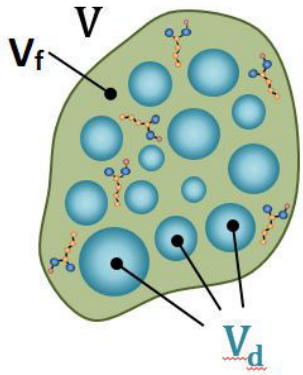

Figure 2.2: Generic colloidal system with total volume, V, dispersed phase volume, V_d and continuous phase volume, V_f[14].

combining Equation 2.20 and Equation 2.21 the following expression is found

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{V_f} C_f \,\mathrm{d}V_f - \int_{A_d} N_s \cdot n_s \,\mathrm{d}A_d = 0 \tag{2.22}$$

If all droplets are assumed to be spherical then $n_s = [1, 0, 0]$, which means that $N_s \cdot n_s = N_{sr}$. The concentration of surfactant in the continuous phase, C_f, is assumed to be constant throughout the continuous phase volume, V_f, and the amount of surfactant adsorbed at the surface, N_s, is assumed to be constant across the surface area of the dispersed phase A_d. Equation 2.22 now becomes

$$\frac{\mathrm{d}C_f}{\mathrm{d}t}V_f - N_{sr}A_d = 0 \tag{2.23}$$

The volume fraction is defined as

$$\phi = \frac{V_d}{V} \; and \; 1 - \phi = \frac{V_f}{V} \tag{2.24}$$

Combining the volume fraction relations and inserting them into Equation 2.23 yields

$$\frac{\mathrm{d}C_f}{\mathrm{d}t} = \frac{\phi}{1-\phi}\frac{A_d}{V_d}N_{sr} \tag{2.25}$$

By assuming that the solute concentration is very dilute the following relation holds true

$$N_{sr} = -K_f(C_f - C^*) \tag{2.26}$$

where K_f is the average velocity through the boundary layer formed by the surfactants adsorbed on the dispersed phase surface and C^* is the concentration at the outside of the same boundary layer. $\frac{A_d}{V_d}$ is the surface to volume ratio of the dispersed phase, when multiplied with the volume fraction, ϕ, it becomes

$$\frac{A_d}{V_d}\phi = \frac{A_d}{V_d}\frac{V_d}{V} = \frac{A_d}{V} = \alpha \tag{2.27}$$

Where α is the surface to volume ratio between the area of the dispersed phase, A_d, and the total volume of the dispersion, V. Substituting Equation 2.26 and Equation 2.27 into Equation 2.25 yields

$$\frac{\mathrm{d}C_f}{\mathrm{d}t} = -\frac{\alpha\phi}{1-\phi}K_f(C_f - C^*) \quad \text{at} \quad t = 0, \quad C_f = C_{f,0} \tag{2.28}$$

where $C_{f,0}$ is the concentration of surfactant at the initial time $t = 0$. α can be found from the volume density distribution, f_v, as follows

$$\alpha = \frac{1}{\phi}\int_0^{R_{max}} \frac{3}{r'} f_v \, \mathrm{d}r' \tag{2.29}$$

A quick look at the dimensions of the volume density distribution shows that this integral in fact becomes the area of the dispersed phase droplets over the total volume of the distribution. f_v has dimensions of $\frac{\mathrm{m^3 droplets}}{\mathrm{m^3 total}}$.

Equation 2.28 accounts for the change of surfactant mass in the continuous phase. An equation for the change of surfactant mass in the dispersed phase is still needed. The solute is assumed only to be present at the surface of the droplets and not inside them. Adsorption on the surface of the dispersed phase is approximated by the Langmuir isotherm

$$\Gamma = \Gamma_{max}\frac{KC^*}{1 + KC^*} \tag{2.30}$$

where Γ is the concentration of solute adsorbed on the dispersed phase surface, Γ_{max} is the maximum amount of solute that can be adsorbed on the dispersed

phase surface and K is an equilibrium constant. The dynamic behaviour of the adsorbed phase concentration is given by the flux to the surface and the change in surface area due to coalescence

$$\frac{\mathrm{d}\Gamma}{\mathrm{d}t} = K_f(C_f - C^*) - \frac{\Gamma}{\alpha}\frac{d\alpha}{dt} \tag{2.31}$$

By rearranging Equation 2.30 to solve for C^* and inserting it into Equation 2.31 the following expression is found

$$\frac{\mathrm{d}\Gamma}{\mathrm{d}t} = K_f\left(C_f - \frac{\Gamma}{K(\Gamma_{max} - \Gamma)}\right) - \frac{\Gamma}{\alpha}\frac{d\alpha}{dt} \quad \text{at} \quad t = 0, \quad \Gamma = \Gamma_0 \tag{2.32}$$

where Γ_0 is the amount of surfactant adsorbed on the dispersed phase at the initial time, $t = 0$.

The Langmuir isotherm can also be substituted for C^* in Equation 2.28, yielding

$$\frac{\mathrm{d}C_f}{\mathrm{d}t} = -\frac{\alpha}{1-\phi}K_f\left(C_f - \frac{\Gamma}{K(\Gamma_{max} - \Gamma)}\right) \quad \text{at} \quad t = 0, \quad C_f = C_{f,0} \tag{2.33}$$

Initial conditions for Equation 2.32 and Equation 2.33 are respectively

$$\Gamma_0 = \frac{K\Gamma_{max}}{1 + KC_{f,0}}C_{f,0} \tag{2.34}$$

$$C_{f,0} = -\frac{1}{2K}\left[1 + \frac{\alpha_0 K\Gamma_{max}}{1+\phi} - KC_{f,\infty}\right] \pm \frac{1}{2K}\sqrt{1 + 2KC_{f,\infty} + K^2C_{f,\infty}^2} + \frac{\alpha_0^2K^2\Gamma_{max}^2}{(1+\phi)^2} + \frac{2\alpha_0 K\Gamma_{max}}{1+\phi} - \frac{2\alpha K^2\Gamma_{max}C_{f,\infty}}{1+\phi} \tag{2.35}$$

where $C_{f,\infty}$ is the concentration of surfactant in the continuous phase at infinite dilution and α_0 is the surface ratio of droplet area to the total volume of the dispersion for the initial distribution, meaning f_v at $t = 0$.

The interfacial tension of the dispersed phase droplets are calculated by the following expression

$$\gamma = \gamma_0 - RT\Gamma_{max}ln(1 + KC) \tag{2.36}$$

where R is the gas constant and T is the temperature.

Considering that the interfacial tension affects the coalescence and breakage efficiency, a couple of subtle modifications has to be made to Equation 2.10 and Equation 2.13. The second order inverse ratio of surface tension to the initial surface tension, $\left(\frac{\gamma}{\gamma_0}\right)^{-2}$, is multiplied into the exponent of Equation 2.10 and the ratio of surface tension to the initial surface tension, $\frac{\gamma}{\gamma_0}$, is multiplied into the exponent of Equation 2.13 yielding

$$\psi_E(r_1, r_2) = exp\left[-\frac{1}{k_\psi}\frac{\mu_c \rho_c \epsilon}{\gamma_0^2}\left(\frac{\gamma_0}{\gamma}\right)^2\left(\frac{r_1 r_2}{r_1 + r_2}\right)^4\right] \tag{2.37}$$

$$g(r) = k_{g1}\frac{\epsilon^{1/3}}{r^{2/3}(1+\phi)} exp\left[-k_{g2}\frac{\gamma_0(1+\phi)^2}{\rho_d \epsilon^{2/3} r^{5/3}}\left(\frac{\gamma}{\gamma_0}\right)\right] \tag{2.38}$$

It is worth noting that Equations 2.37-2.38 are what couples the surfactant mass balance with the number density- and volume density distribution.

This concludes the governing equations of the model. In Appendix A the model is converted into non-dimensional variables.

Chapter 3

Programing and Software

The model equations derived in Chapter 2 have been implemented in Matlab using a combination of the built in ODE-solvers and the numerical method of collocation. The ODE solvers are based on initial value problems, which is how the model equations are formulated. On the account that the equations that constitute the population balance are integral differential equations a separate approach to solve the integral parts had to be applied. The quadrature weights generated by the collocation method have been used to solve the integral parts of the equations before implementing the initial value problem solver in the code. A brief description of the math behind the collocation method is provided in Appendix B.

3.1 Matlab Code

All the Matlab scripts and -functions used to solve the model are given in Appendix C. To start solving the model the script *main* is ran in the command window. It rescales the original distribution, sets up the grid points, calculates the initial conditions and solves the generated set of equations. The script *parameters* accounts for most of the constants that are used in the model equations. The code is generic, so any change to a parameter of the system changes the whole code to accommodate it.

Before the initial experimental distribution can be used in the model solver it has to be normalized, as it has no physical meaning in its rawest form. The script *rescale* is given the water cut and a text file that contains the raw distribution. For the experimental distribution to get the appropriate dimensions it has to be normalized so that Equation 2.2 holds true. After the distribution is normalized it is also converted to its non-dimensional form, as shown in Appendix A.

Figure 3.1 shows an example of how the collocation grid can be set up. *SetUpGrid* is the function that splits the radius domain into elements and calls the colloc routine[15] that generates the roots and weights used to calculate the integrals in the differential equation. First, strategic sub-domains are chosen so that the amount of collocation points can be varied in different regions of the distribution. This makes sure that regions of the radius domain where steep gradients in the number distribution occur can be allowed more points and regions where the number distribution is zero can be given fewer points. Domains were chosen between $r = 0$ and the first non-zero value of the distribution, between the first non-zero value of the distribution and the highest value of the distribution and between the highest value of the distribution and the last non-zero value. The next domains are just equally spaced until the end of the distribution. A minimum of four domains are required for the script to work and the amount of domains and collocation points in each domain can be changed inside *parameters*.

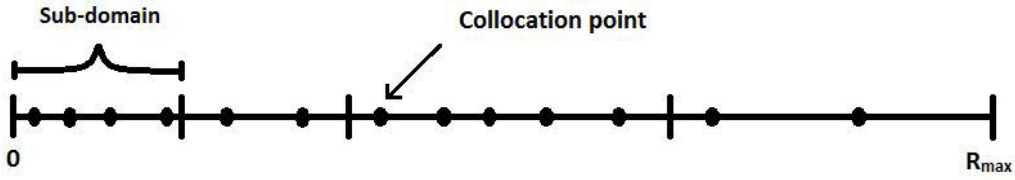

Figure 3.1: Illustration of arbitrary collocation grid for demonstration purposes.

For each domain the colloc function[15] is called and the quadrature weights and roots are stored in separate arrays. The vector of roots is then scaled to range from 0 to 1, which is the non-dimensional radius domain. Finally, the scaled number distribution is interpolated with the scaled vector of roots to yield the initial distribution and radius domain that is given to the ODE-solver.

When the initial conditions for the mass balance have been calculated through the function *alphacalc*, the ODE-solver is called. The solver calls the function *differentialeq* that creates the vector of differentials that are to be solved. Inside *differentialeq* a call is made to the function *interp*, which calculates all the birth and death rates of breakage and coalescence. This is done using the quadrature weights generated in the *SetUpGrid* function earlier. To accommodate the limits of the integrals that change depending on which point they are being solved on, interpolation with the current distribution is done. *Interp* also calculates the surface to volume ratio of the distribution and the derivative of the surface to volume ratio, which are used in the differential equations governing the surfactant mass balance.

To solve the model without the surfactant mass balance the only change that has

to be made is to replace the gamma ratios mentioned in Equations 2.37 and 2.38 with unity. This way the surfactant mass balance is no longer coupled with the number distribution and the distribution remains unaffected by the surfactant mass balance.

The functions *PlotInitialConditions*, *PlotLengthVariables*, *PlotProperties*, *PlotRadiusVariables*, *PlotTransientData*, *SurfPlots*, *PlotSurfactantVariables* and *ConPlots* provides all the plots used in the results and discussion section below.

Chapter 4

Results and Discussion

4.1 Base Case Study

A base case was studied first, to give the reader an overview of what goes into the program(see section 4.1.1) and what comes out(see section 4.1.2), in terms of plots and parameters. In the base case, one set of the fitting parameters for coalescence frequency, k_ω, coalescence efficiency, k_ψ, breakage frequency, k_{g1} and breakage efficiency, k_{g2}, generates one set of output plots. Fitting parameters, species properties[16] and other parameters used in the program are given in Table 4.1. This set of parameters has been chosen arbitrarily for demonstration purposes. In addition, the ODE-solver has to be given an experimental distribution[16] as a initial condition. The experimental distribution has arbitrary units, thus it is scaled so that Equation 2.2 holds true.

4.1.1 Input to The Program

Table 4.1 shows the parameters that are given to the program as input for the base case study. The value of R_{max} was chosen to be large enough so that the number- and volume density distribution does not exceed R_{max} as they evolve with the length of the pipe.

In Figure 4.1 the experimental number density distribution, $f_{n,exp}$, and the experimental volume density distribution, $f_{v,exp}$, is plotted as a function of the droplet radius, r. The blue and red circles show the interpolated initial number density distribution, $f_{n,0}$, and the interpolated initial volume density distribution, $f_{v,0}$, plotted as a function of the droplet radius, r. Interpolation is necessary because a set of experimental data points have to be mapped on to the set of collocation points that comprise the simulation grid mesh, and are generated in the

Matlab code in Appendix C. Piecewise Cubic Hermite Interpolating Polynomials was used in Matlab, as it is focused on curve shape preservation and therefore avoids oscillating data points. Figure 4.1 shows that the $f_{n,0}$ and $f_{v,0}$ perfectly fits the curves of $f_{n,exp}$ and $f_{v,exp}$ respectively.

Table 4.1: Parameters that are given to the program.

Parameter	Value	Explanation
ϕ	0.34	Volume fraction of water
U	$0.16\left[\frac{\mathrm{m}}{\mathrm{s}}\right]$	Average velocity of fluid
L	$3000[\mathrm{m}]$	Pipe length
R_{max}	$600[\mu\mathrm{m}]$	Upper bound of the radius domain
D	$0.0254[\mathrm{m}]$	Pipe diameter
ρ_d	$1023\left[\frac{\mathrm{kg}}{\mathrm{m}^3}\right]$	Density of the dispersed phase
μ_d	$0.001[\mathrm{Pas}]$	Viscosity of dispersed phase
ρ_c	$786\left[\frac{\mathrm{kg}}{\mathrm{m}^3}\right]$	Density of the continuous phase
μ_c	$0.0013[\mathrm{Pas}]$	Viscosity of continuous phase
γ_0	$0.05\left[\frac{\mathrm{N}}{\mathrm{m}}\right]$	Initial interfacial tension
k_ω	0.1	Fitting parameter for the coalescence frequency expression
k_{g1}	0.1	Fitting parameter for breakage frequency expression
k_ψ	$1e-19$	Fitting parameter for coalescence efficiency expression
k_{g2}	$1e-5$	Fitting parameter for breakage efficiency expression

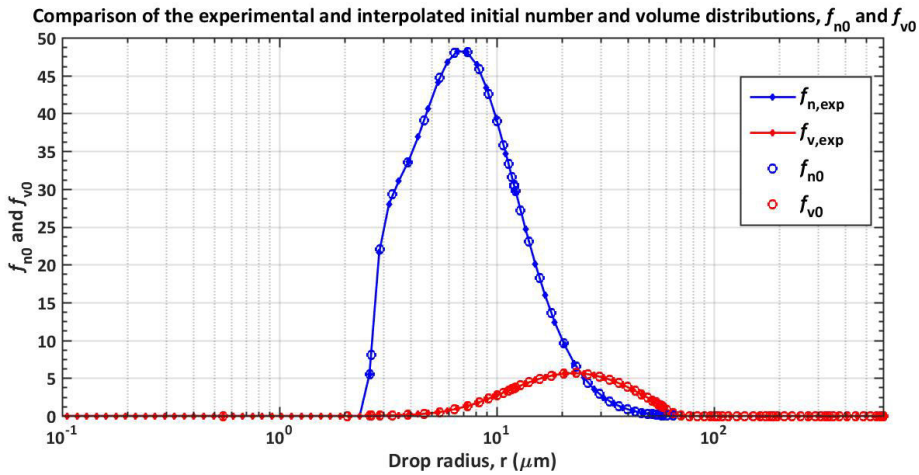

Figure 4.1: Experimental number density distribution, $f_{n,exp}$, and experimental volume density distribution, $f_{v,exp}$, plotted as a function of the droplet radius, r. Blue and red circles show the interpolated initial number density distribution, $f_{n,0}$, and the interpolated initial volume density distribution, $f_{v,0}$, respectively, plotted as a function of the droplet radius, r.

4.1.2 Output From The Program

In Figure 4.2 the number density distribution, f_n, is plotted as a function of the droplet radius, r, at nine different pipe lengths. At the initial pipe length ($z = 0$) there is a large quantity of drops of equal size and as the distribution evolves along the pipe it gets smaller in magnitude, indicating fewer drops of equal size. This is expected in the case where coalescence is dominant considering that many smaller sized drops will coalesce and form fewer larger drops, leading the curve in Figure 4.2 to flatten out as z increases.

In Figure 4.3 the volume density distribution, f_v, is plotted as a function of the droplet radius, r, at nine different pipe lengths. f_v evolves from a low and wide distribution at very low pipe lengths to a thinner and taller distribution as z increases. This confirms that overall coalescence is dominating overall breakage, considering there are more larger droplets. If breakage had dominated overall, a wider distribution than the initial distribution would be expected.

Figure 4.4 shows the average radius of the number density distribution, μ_n, and the average radius of the volume density distribution, μ_v, plotted as a function of the axial position in the pipe, z. Both μ_n and μ_v increases over the total pipe length indicating that coalescence is dominating over breakage for this set of fitting

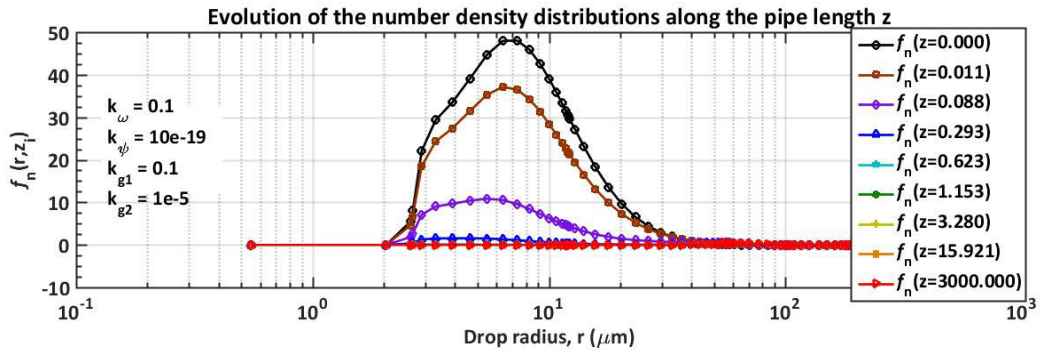

Figure 4.2: Number density distribution, f_n, plotted as a function of the droplet radius, r, at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

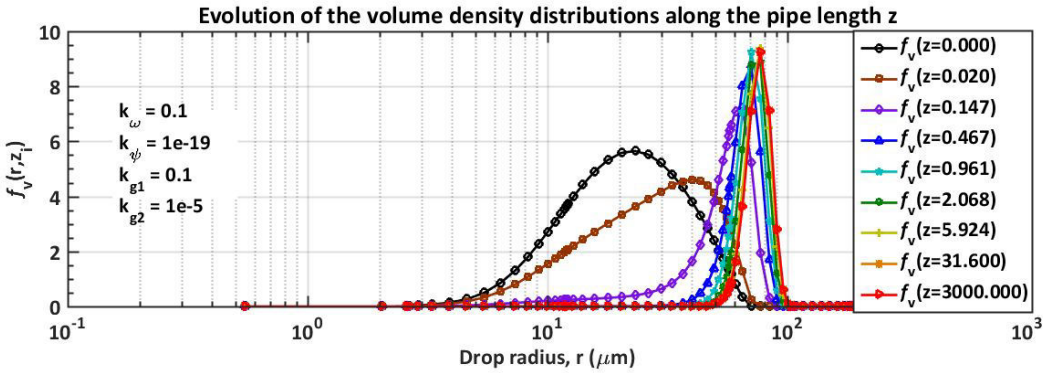

Figure 4.3: Volume density distribution, f_v, plotted as a function of the droplet radius radius, r, at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

parameters. There is a gap between μ_n and μ_v at lower z-values that becomes smaller as z increases. The reason for this becomes clear when comparing Figure 4.2 and 4.3, where it can be seen that the relative change in f_n along the pipe length is a lot larger than it is for μ_v. This means that μ_n should have a larger gradient than μ_v. The distribution also becomes narrower, which suggests that the values of μ_n and μ_v are approaching each other.

At the point where the average radius is no longer growing the distribution is said to be equilibriated, meaning that μ_n and μ_v have reached a constant value. The length at which equilibrium is achieved, L_∞, is set to the point on the z-axis where the gradient of μ_v reaches a threshold value set to be $1e-4$. μ_v is is used

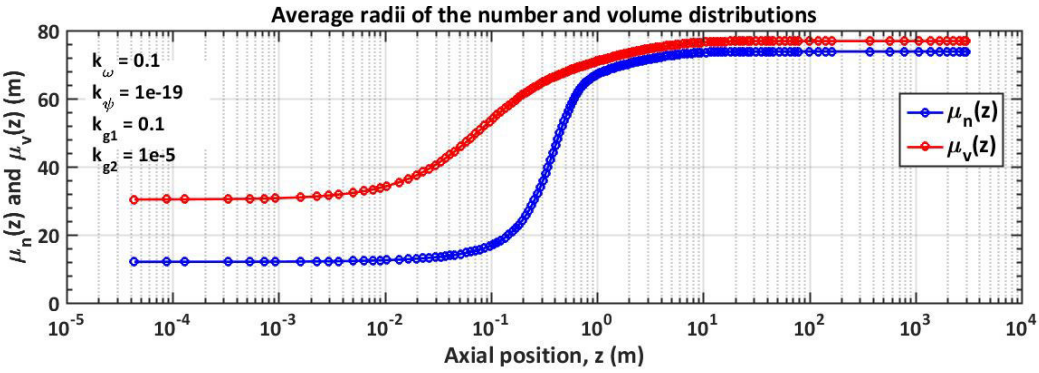

Figure 4.4: Average radii of the number distribution, μ_n, and average radii of the volume distribution, μ_v, plotted as a function of the axial position in the pipe, z. The fitting parameters that were used are shown at the left side of the plot.

to determine L_∞, because from experience with running the code it settles later than μ_n. This way, both μ_n and μ_v have reached equilibrium at the chosen value of L_∞. For this set of fitting parameters L_∞ was found to be 9.7 m. The average radii of the number density distribution at L_∞, $\mu_{n,\infty}$, and the average radii of the volume density distribution at L_∞, $\mu_{v,\infty}$, were found to be 63.3 μm and 69.3 μm respectively.

L_∞ is a very useful number to measure, because the relative effect and interplay of the coalescence and breakage mechanisms and their effects on the system dynamics can by studied by simulating different combinations of the fitting parameter given in Table 4.1. Also, L_∞ is a parameter of practical importance for pipeline flow simulations because the volume and number density distribution at L_∞ are important information to estimate the total fluid viscosity and, consequently, pumping requirements. Considering that L_∞ shows how fast the distribution reaches a constant average radius, it should vary significantly when the fitting parameters are changed. Specifically, changes in k_ω and k_{g1} should control L_∞ as they influence how fast coalescence and breakage takes place. From Equation 2.8 and 2.13 it is evident that an increase in either k_ω or k_{g1} should increase coalescence- or breakage frequency respectively. This increases $\frac{df_n}{dz}$, which is the main reason why the system equilibriates faster. Thus, k_ω and k_{g1} can be considered to be "gain" constants that control the relative magnitudes between coalescence- and breakage frequency and have a strong influence on L_∞. Variations in L_∞ due to changes in k_ω and k_{g1} are studied in Section 4.2.

The average drop size of the volume distribution, $\mu_{v,\infty}$, at L_∞ is another useful

number to measure. As argued above μ_v is used because it is equilibriated later than μ_n. $\mu_{v,\infty}$ should mainly be controlled by k_ψ and k_{g2} as these two parameters influence the efficiency with which droplets coalesce and break up respectively. An increase in k_ψ or a decrease in k_{g2} should lead to increased coalescence- and breakage efficiency respectively. If coalescence efficiency is high, larger droplets are expected to coalesce and if breakage efficiency is high, smaller drops are expected to break, leading to higher or lower average drop size respectively. k_ψ and k_{g2} are thus considered to be "gain" constants that influence the shape of the distribution. Variations in $\mu_{v,\infty}$ due to changes in k_ψ and k_{g2} are studied in Section 4.3.

The model equations are designed so that the mass balance is upheld, by having terms that account for both death and birth of droplets. The mass created and the mass that disappear therefore have to balance each other so that the following equations hold true

$$\int_0^{R_{max}} vR_{C,+}\,\mathrm{d}r - \int_0^{R_{max}} vR_{C,-}\,\mathrm{d}r = 0 \tag{4.1}$$

$$\int_0^{R_{max}} vR_{B,+}\,\mathrm{d}r - \int_0^{R_{max}} vR_{B,-}\,\mathrm{d}r = 0 \tag{4.2}$$

Since the use of interpolation introduces errors that may accumulate over time, measures have to be taken to make sure that the mass balance remains intact. The ratio of the mass balance of coalescence, M_C, and the ratio of the mass balance of breakage, M_B, are given by the following expressions

$$M_C = \frac{\int_0^{R_{max}} vR_{C,-}\,\mathrm{d}r}{\int_0^{R_{max}} vR_{C,+}\,\mathrm{d}r} \tag{4.3}$$

$$M_B = \frac{\int_0^{R_{max}} vR_{B,-}\,\mathrm{d}r}{\int_0^{R_{max}} vR_{B,+}\,\mathrm{d}r} \tag{4.4}$$

To make sure that ϕ stays constant, meaning that no drops are introduced or taken away along the pipe length, M_C and M_B are multiplied with the birth rate of coalescence and the birth rate of breakage respectively at every iteration of the ODE-solver.

Figure 4.5 shows the volume fraction of water, ϕ, the coalescence mass balance ratio, M_C, and the breakage mass balance ratio, M_B, plotted as a function of the axial position in the pipe, z. In an ideal situation these should all remain constant throughout the length of the pipe, which ϕ and M_B does. M_C on the other hand

is never equal to one and makes a leap up to around 1.3 at a low pipe length and settles there. From Equation 4.1 is is clear that $R_{C,+}$ is reduced relative to $R_{C,-}$ since M_C is larger than 1. As long as M_C does not escalate further, it is reasonable to assume that this does not affect the models accuracy to a large extent. The behavior of M_C most likely indicates that the coalescence birth rate is missing a term that arises from the assumption of representing the droplet pair correlation function as simply the product of the number density distribution of droplets with size r' and r''[17].

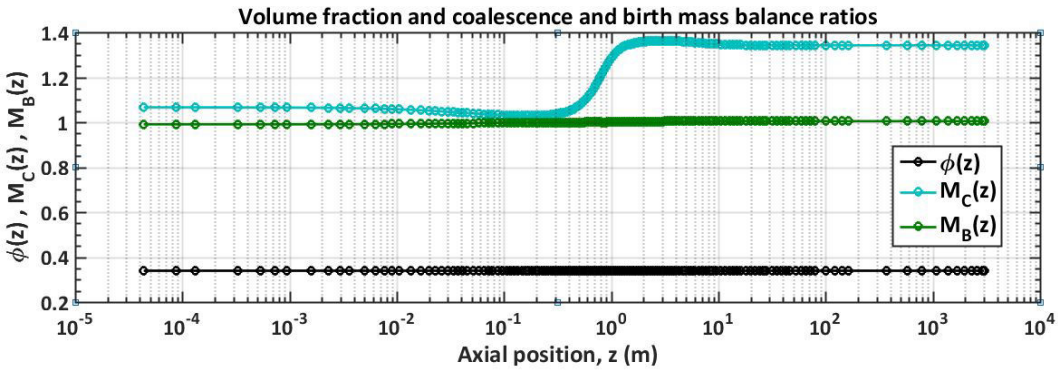

Figure 4.5: Volume fraction of water, ϕ, coalescence mass balance ratio, M_C, and breakage mass balance ratio, M_B, plotted as a function of axial position in the pipe, z.

Figure 4.6 shows the total rate of coaelscence, R_{Ct}, plotted as a function of the droplet radius, r, for nine different pipe lengths. Figure 4.7 shows the total rate of breakage, R_{Bt}, plotted as a function of droplet radius, r, for nine different pipe lengths, z. In Figure 4.6 and 4.7 the negative sections of the curves represent the death of droplets due to coalescence and breakage respectively and the positive parts represent the birth of droplets due to coalescence and breakage respectively. It is logical that R_{Ct} is negative at low r-values, since smaller drops are the most likliest to coalesce and vice versa for R_{Bt}.

In Figure 4.6 R_{Ct} is large at the outset and becomes smaller and smaller as z increases, indicating that coalescence is strong at the beginning of the pipe. The rate of coalescence is determined by multiplying the frequency of collisions with the efficiency of coalescence, as can be seen in Equation 2.7. At lower z-values there are a lot of small droplets that will coalesce and turn into fewer larger droplets. As droplets grow larger, fewer and fewer are able to coalesce and the rate of coalescence declines.

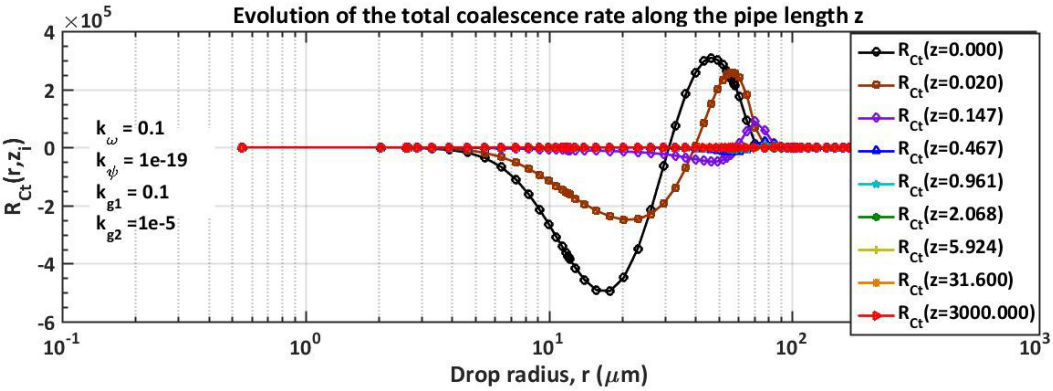

Figure 4.6: Total coalescence rate, R_{Ct}, plotted as a function of the droplet radius radius, r, at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

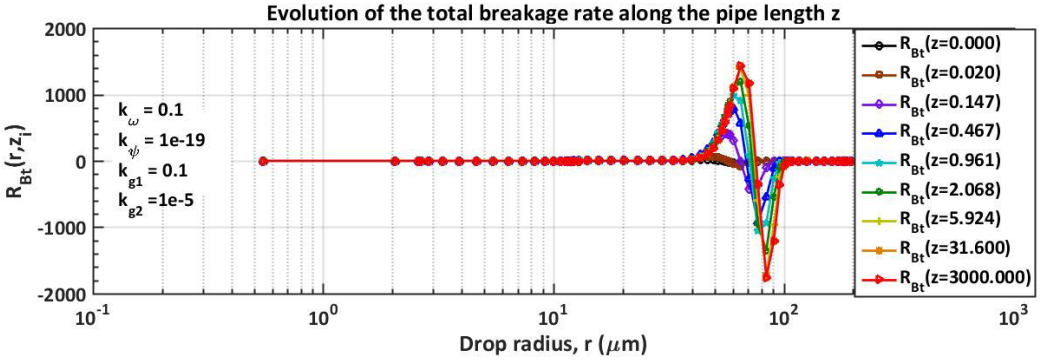

Figure 4.7: Total breakage rate, R_{Bt}, plotted as a function of the droplet radius radius, r, at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

From Figure 4.7 it is evident that R_{Bt} has the exact opposite trend that R_{Ct} showed. It becomes larger as z increases indicating that breakage becomes stronger towards the end of the pipe. The rate of breakage, in opposition to the rate of coalescence, does not depend on the collision of droplets and therefore not on the amount of droplets that are present in the system. Breakage becomes stronger when the droplets in the system become larger, because when droplets become larger breakage frequency increases as can be seen from Equation 2.13.

With these two opposing trends in mind it is clear why, in Figure 4.4, μ_n and μ_v increases at lower pipe lengths as coalescence is strong here. At the inflection

point of μ_n and μ_v drop sizes have become sufficiently large that breakage is now overcoming coalescence. Then, breakage becomes larger and larger with increasing z until an equilibrium is reached and μ_n and μ_v reach constant values.

4.2 Analysis of System Behaviour Based on Variations in Coalescence- and Breakage Frequencies

As discussed in section 4.1.2 changes made to k_ω and k_{g1} should control L_∞ as these two parameters govern how fast coalescence and breakage occur in the system. It is therefore interesting to look at how L_∞ changes when k_ω and k_{g1} are varied and see if this analysis is correct.

Figure 4.8 shows the length at which the volume density distribution is equilibriated, L_∞, plotted as a function of the fitting parameter for breakage frequency, k_{g1}, and the fitting parameter for coalescence frequency, k_ω. The fitting parameters of coalescence efficiency, k_ψ, and breakage efficiency, k_{g2}, were kept constant at $1e-19$ and $1e-4$ respectively.

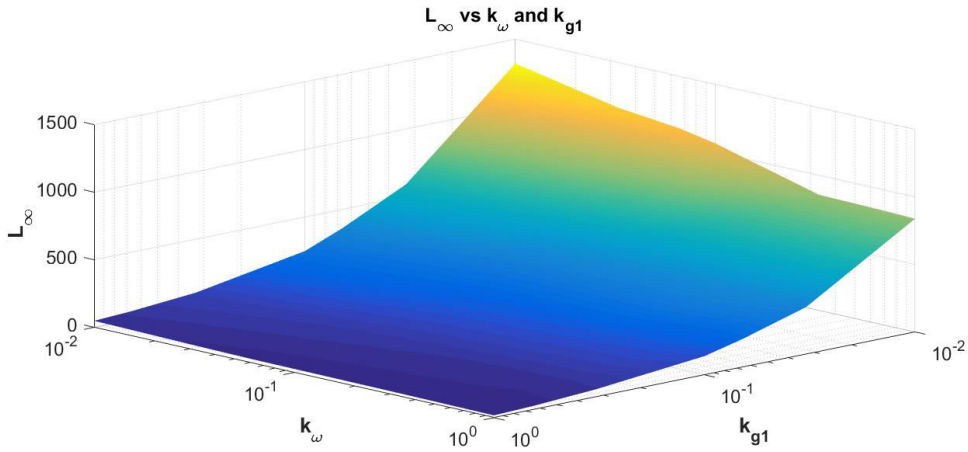

Figure 4.8: Length at which μ_v is equilibriated, L_∞, plotted as a function of the fitting parameter for breakage frequency, k_{g1}, and the fitting parameter for coalescence frequency, k_ω. The fitting parameters of coalescence efficiency and breakage efficiency were kept constant at $1e-19$ and $1e-4$ respectively.

An increase in the magnitude of k_ω or k_{g1} should result in increased coalescence frequency or breakage frequency respectively. As discussed in section 4.1.2 this should in term shorten L_∞. This is exactly what can be seen in Figure

4.8, where L_∞ reaches its lowest magnitudes when both k_ω and k_{g1} have low magnitudes. L_∞ also seems to vary more along the the k_ω-axis when k_{g1} is at lower magnitudes. This makes sense, because at lower magnitudes of k_{g1} the frequency of breakage is low allowing the variation in magnitude of k_ω to have a larger impact on L_∞.

Figure 4.9 shows the average radius of the volume density distribution, $\mu_{v,\infty}$, at L_∞, plotted as a function of the fitting parameter for breakage frequency, k_{g1}, and the fitting parameter for coalescence frequency, k_ω. The fitting parameters of coalescence efficiency, k_ψ, and breakage efficiency, k_{g2}, were kept constant at $1e-19$ and $1e-4$ respectively.

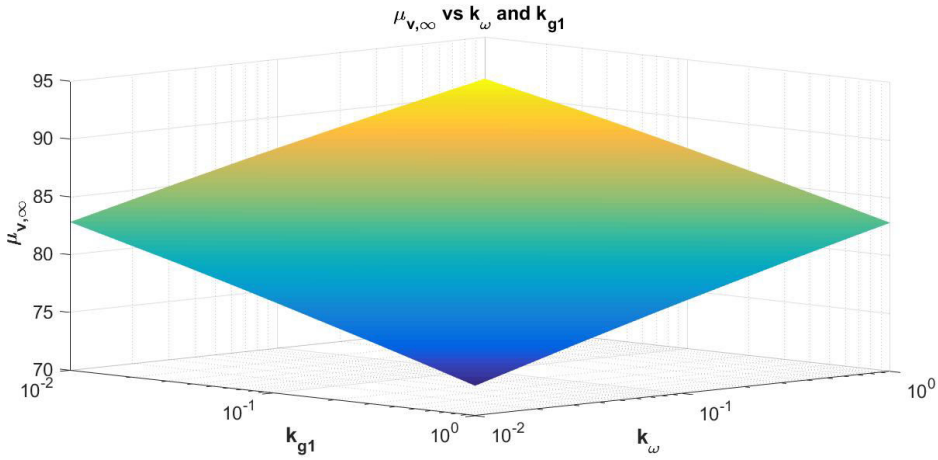

Figure 4.9: Average radius of the volume density distribution, $\mu_{v,\infty}$, at L_∞, plotted as a function of the fitting parameter for breakage frequency, k_{g1}, and the fitting parameter for coalescence frequency, k_ω. The fitting parameters of coalescence efficiency and breakage efficiency were kept constant at $1e-19$ and $1e-4$ respectively.

Moving along any horizontal line in Figure 4.9 corresponds to changing the values of k_ω and k_{g1} by equal magnitudes either up or down. $\mu_{v,\infty}$ is constant along any horizontal line in Figure 4.9 as well, indicating that the average radius of the volume density distribution does not change as long as changes of equal magnitude are made in the values of k_ω and k_{g1}. In other words, it demonstrates that as long as $k_\omega = k_{g1}$ $\mu_{v,\infty}$ does not change. This observation suggests that as long as changes of equal magnitude are made to the value of k_ω and k_{g1}, only changes made in the magnitude of k_ψ and k_{g2} can change the shape of the final distribution. This is an important result to consider when attempting to use this

model to fit experimental data sets. Specifically, if k_ω and k_{g1} are treated as the same parameter, k_{frq}, then the statistical reliability of the model fit should improve since k_{frq} would mainly control the axial translation of the distribution while k_ψ and k_{g2} would mainly determine the width and shape.

Figure 4.10 shows the length at which the volume density distribution is equilibriated, L_∞, plotted as a function of the fitting parameter for breakage frequency, k_{g1}, and the fitting parameter for coalescence frequency, k_ω. The fitting parameters of coalescence efficiency, k_ψ, and breakage efficiency, k_{g2}, were kept constant at $1e-20$ and $1e-5$ respectively. This means that coalescence will not create as large drops and breakage will occur on smaller drops than the previous case.

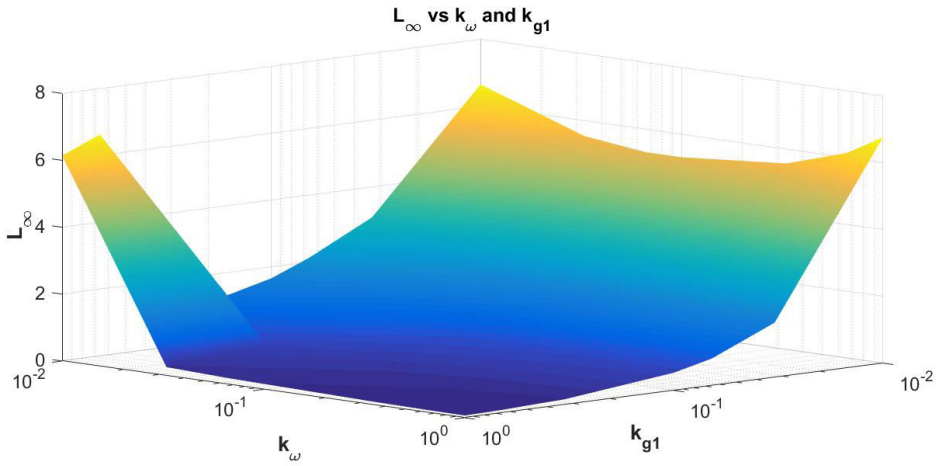

Figure 4.10: Length to equilibriated number distribution, L_∞, plotted as a function of the fitting parameter for breakage frequency, k_{g1}, and the fitting parameter for coalescence frequency, k_ω. The fitting parameters of coalescence efficiency and breakage efficiency were kept constant at $1e-20$ and $1e-5$ respectively.

Figure 4.10 looks similar to Figure 4.8 except for a sharp elevation in L_∞ when k_{g1} is large and k_ω is small. In Figure 4.10 both k_ψ and k_{g2} have been decreased in magnitude, which should lead to lower coalescence efficiency and higher breakage efficiency as discussed above. Since the peak in Figure 4.10 appears at low k_ω indicating low coalescence frequency, it might be an indication that total rate of coalescence is so low that the distribution reaches equilibrium slower. However, since overall breakage usually increases with z as discussed above, it is unusual that L_∞ increases when k_{g1} is large. It might indicate that the parameter combination of k_ω and k_{g1} is in an unlikely state.

The variation in L_∞ along the k_ω-axis everywhere else in Figure 4.10 is small, which corresponds to the trend seen in Figure 4.8. This could indicate that L_∞ is more dependent on the magnitude of k_{g1}, which makes sense considering that breakage is known to grow larger with increasing z, as discussed above. L_∞ is a lot lower in general in Figure 4.10 than it is in Figure 4.8. This is an indication that changes in k_ψ and k_{g2} also affect L_∞ to a large extent.

Figure 4.11 shows the average radius of the volume distribution $\mu_{v,\infty}$, at L_∞, plotted as a function of the fitting parameter for breakage frequency, k_{g1}, and the fitting parameter for coalescence frequency, k_ω. The fitting parameters of coalescence efficiency, k_ψ, and breakage efficiency, k_{g2}, were kept constant at $1e-20$ and $1e-5$ respectively.

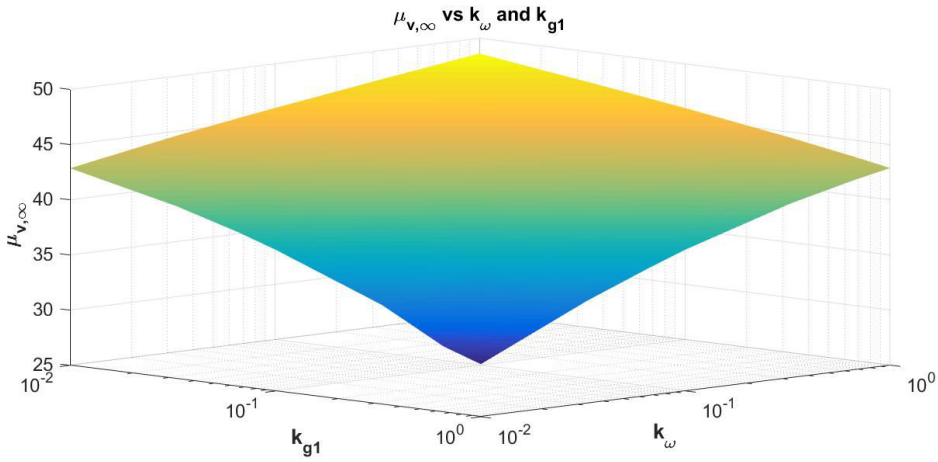

Figure 4.11: Average radius of the volume density distribution, $\mu_{v,\infty}$, at L_∞, plotted as a function of the fitting parameter for breakage frequency, k_{g1}, and the fitting parameter for coalescence frequency, k_ω. The fitting parameters of coalescence efficiency and breakage efficiency were kept constant at $1e-20$ and $1e-5$ respectively.

Figure 4.11 shows the exact same trend as Figure 4.9. When changes of equal magnitude are made to the values of k_ω and k_{g1}, $\mu_{v,\infty}$ is constant. This shows that the trend is consistent for a different set of k_ψ and k_{g2} values and proves that it is not an anomaly. In Figure 4.11 average drop size is smaller in general. Since k_ψ and k_{g2} are lower, the coalescence efficiency and the breakage efficiency have been decreased and increased respectively. When coalescence efficiency decreases larger drops are not able to coalesce and when breakage efficiency

increases smaller drops are able to break. This should amount to a significant lowering of the average drop size.

4.3 Analysis of System Behaviour Based on Variations in Coalescence- and Breakage Efficiencies

As discussed in Section 4.1.2 changes in the magnitudes of k_ψ and k_{g2} should have an effect on the equilibriated average drop size. To better understand the dynamics of coalescence- and breakage efficiency it is interesting to look at how $\mu_{v,\infty}$ changes in magnitude when k_ψ and k_{g2} are varied.

Figure 4.12 shows the average radius of the volume density distribution, $\mu_{v,\infty}$, at L_∞ plotted against the fitting parameter for coalescence efficiency, k_ψ, and the fitting parameter for breakage efficiency, k_{g2}. The fitting parameter of coalescence frequency, k_ω, and the fitting parameter for breakage frequency, k_{g1}, were kept constant at 0.1.

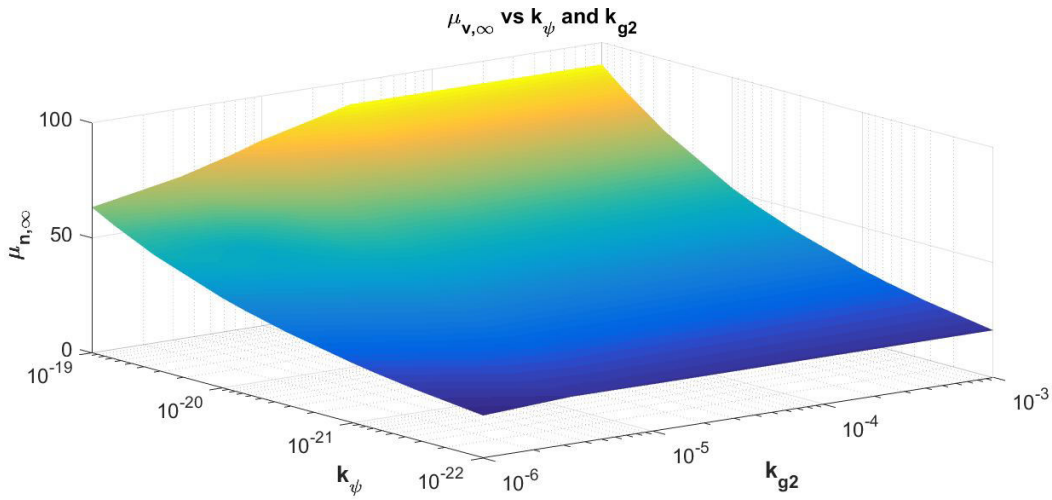

Figure 4.12: Average radius of the volume density distribution, $\mu_{v,\infty}$, at L_∞ plotted as a function of the fitting parameter for coalescence efficiency, k_ψ, and the fitting parameter for breakage efficiency, k_{g2}. The fitting parameter for coalescence frequency and the fitting parameter for breakage frequency were kept constant at 0.1.

From Figure 4.12 it is clear that the efficiency of breakage only has an effect on the average drop size at lower values of k_{g2}. As discussed in section 4.1.2 decreasing the magnitude of k_{g2} increases the efficiency of breakage and therefore

lowers $\mu_{v,\infty}$. This can be seen in Figure 4.12 for lower magnitudes of k_{g2}. For higher magnitudes of k_{g2} there seem to be no effect on $\mu_{v,\infty}$, which is most likely caused by breakage efficiency becoming so small that coalescence is completely dominant and changes in k_ψ completely determines $\mu_{v,\infty}$. Increasing the magnitude of k_{g2}, exponentially decreases the magnitude of breakage efficiency, as can be seen in Equation 2.13. This explains why at higher magnitudes of k_{g2} the effect of breakage is lost on $\mu_{v,\infty}$, because at such low breakage efficiency not even the larger drops are able to break.

As discussed in section 4.1.2 the efficiency of coalescence should increase with increasing magnitude of k_ψ and with higher coalescence efficiency more drops will coalesce and $\mu_{v,\infty}$ should increase. This is exactly the case in Figure 4.12 where the trend clearly shows that the magnitude of $\mu_{v,\infty}$ increases when the magnitude of k_ψ is increased. When the magnitude of k_{g2} becomes smaller, hence increasing breakage efficiency, the effect of increasing k_ψ is counteracted and becomes less clear than for higher magnitudes of k_{g2}. This shows a very clear and dynamic balance between coalescence efficiency and breakage efficiency.

Figure 4.13 shows the average of the volume density distribution, $\mu_{v,\infty}$, at L_∞ plotted against the fitting parameter for coalescence efficiency, k_ψ, and the fitting parameter for breakage efficiency, k_{g2}. The fitting parameter for coalescence frequency, k_ω, and the fitting parameter for breakage frequency, k_{g1}, were kept constant at 1 and 0.01 respectively.

Figure 4.13 indicates the same trend that Figure 4.12 does. Also here $\mu_{v,\infty}$ varies greatly with the magnitude of k_ψ, but to a lesser extent on the magnitude of k_{g2}. The fact that the these trends occur in both Figure 4.12 and 4.13 proves that the trend is not an anomaly. However, the extent to which $\mu_{v,\infty}$ varies with k_{g2} is even lesser in Figure 4.13 than it is 4.12. The reason for this is most likely that k_ω is two orders of magnitude larger than k_{g1} causing coalescence to occur much more often than breakage. More coalescence than breakage increase $\mu_{v,\infty}$. The difference in the magnitudes of $\mu_{v,\infty}$ between these two figures is however modest compared to the magnitude of change induced by variations in k_ψ in each figure individually.This shows that $\mu_{v,\infty}$ is much more dependent on the coalescence efficiency than it is on the coalescence frequency.

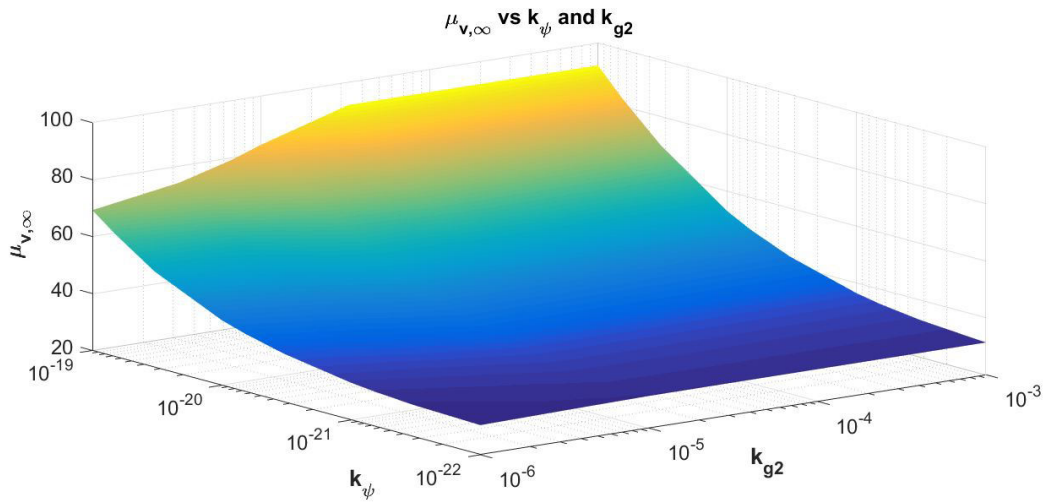

Figure 4.13: Average radius of the volume density distribution, $\mu_{v,\infty}$, at L_∞ plotted as a function of the fitting parameter for coalescence efficiency, k_ψ, and the fitting parameter for breakage efficiency, k_{g2}. The fitting parameter for coalescence frequency and the fitting parameter for breakage frequency were kept constant at 0.1 and 0.01 respectively.

4.4 Analysis of System Behaviour Based on Total Coalescence Rate, Total Breakage Rate and Evolution of The Volume Density Distribution

To further expand the analysis of the system four different sets of fitting parameters have been studied in the context of total coalescence rate, R_{Ct}, total breakage rate, R_{Bt}, and the volume density distribution, f_v. This is done with the intention of comparing these variables and their evolution as coalescence and breakage are varied in their influence.

Figures 4.14-4.16 show the total coalescence rate, R_{ct}, the total breakage rate, R_{bt}, and the volume density distribution, f_v, respectively as a function of droplet radius at nine different pipe lengths. k_ω is set to $0.33e-2$, k_{g1} is set to $0.33e-2$, k_ψ is set to $1e-20$ and k_{g2} is set to $1e-5$ in all three plots.

Figures 4.17-4.19 show the total coalescence rate, R_{ct}, the total breakage rate, R_{bt} and the volume density distribution, f_v, respectively as a function of droplet radius at nine different pipe lengths. k_ω is set to 0.66, k_{g1} is set to 0.66, k_ψ is set to $1e-20$ and k_{g2} is set to $1e-5$ in all three plots.

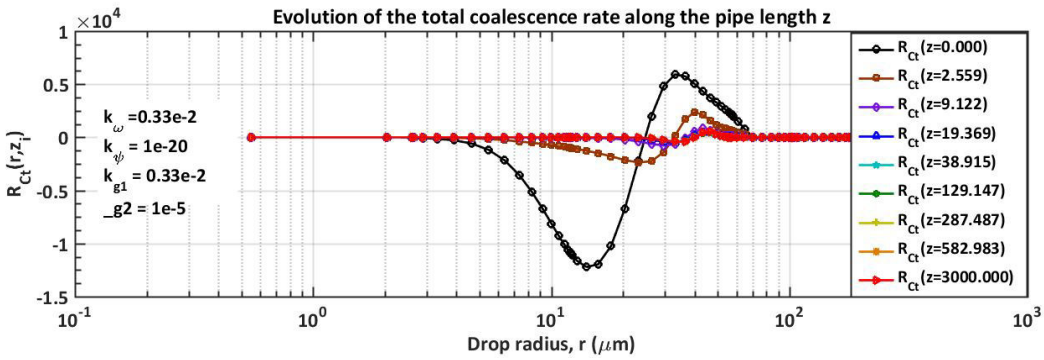

Figure 4.14: Total rate of coalescence, R_{ct}, plotted against droplet radius, r, at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

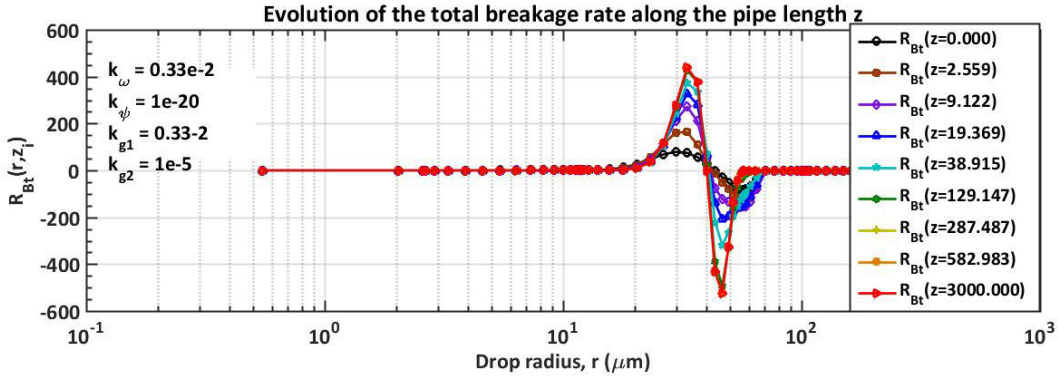

Figure 4.15: Total rate of breakage, R_{bt}, plotted against droplet radius, r, at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

Comparing the system in Figure 4.14-4.16 and the system in Figure 4.17-4.19 several distinct differences become apparent. The magnitude of R_{Ct} is larger for all z-values in Figure 4.17 than it is in Figure 4.14. This makes sense because k_ω is larger in Figure 4.17 and as discussed in section 4.1.2 an increase in k_ω should increase the rate of coalescence. R_{Bt} is larger for all z-values in Figure 4.18 than it is in Figure 4.15. k_{g1} is also larger in Figure 4.18 than it is in Figure 4.15 and as discussed in Section 4.1.2 this should increase the rate of breakage. The shape of the volume density distributions in Figure 4.16 and Figure 4.19 appear

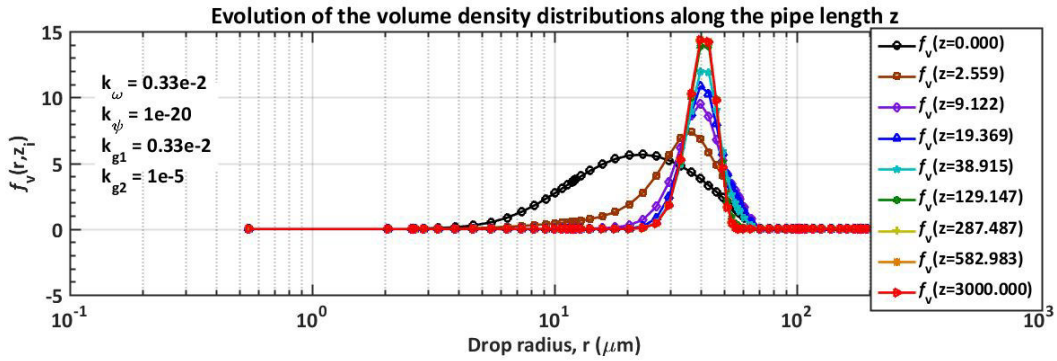

Figure 4.16: Volume density distribution, f_v, plotted at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

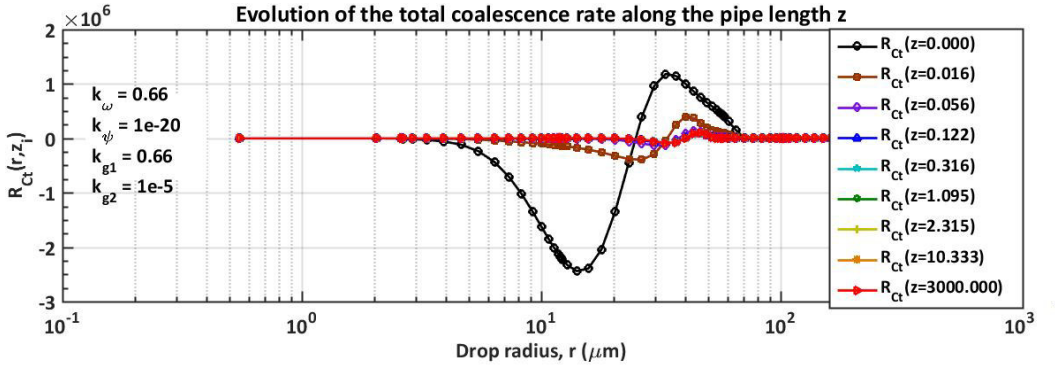

Figure 4.17: Total rate of coalescence, R_{ct}, plotted against droplet radius, r, at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

to be identical which supports the previously made argument that increasing k_ω and k_{g1} by equal magnitudes yields no change in average drop size. This is logical because an increase of k_ω and k_{g1} of equal magnitude should induce the same increase in the magnitude of coalescence frequency as in the magnitude of breakage frequency. k_ω and k_{g1} are direct multiplying factors for coalescence frequency and breakage frequency as seen in Equation 2.8 and 2.13, respectively.

In Figure 4.20 the volume density distribution, f_v, is plotted as a function of the droplet radius, r, at nine different pipe lengths. k_ω is set to 0.33, k_{g1} is set to $0.33e-2$, k_ψ is set to $1e-20$ and k_{g2} is set to $1e-5$.

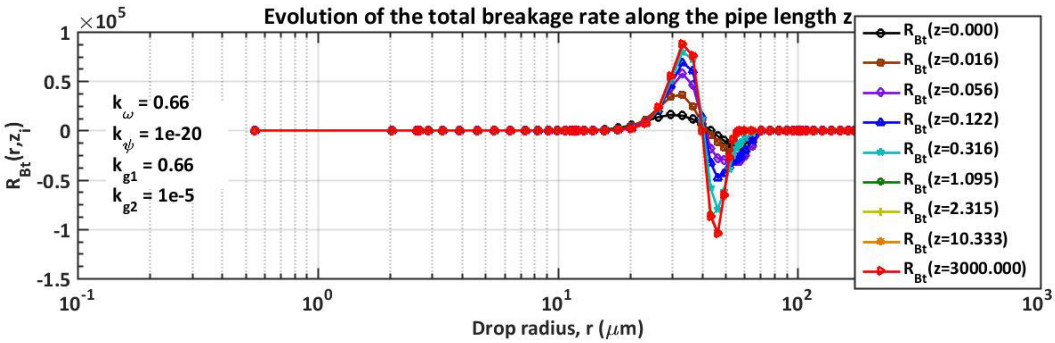

Figure 4.18: Total rate of breakage, R_{bt}, plotted against droplet radius, r, at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

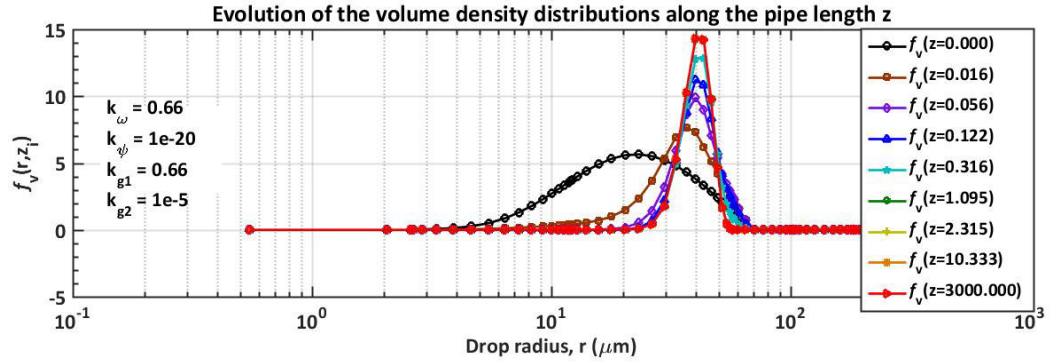

Figure 4.19: Volume density distribution, f_v, plotted at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

In Figure 4.21 the volume density distribution, f_v, is plotted as a function of the droplet radius, r, at nine different pipe lengths. k_ω is set to $0.33e-2$, k_{g1} is set to 0.66, k_ψ is set to $1e-20$ and k_{g2} is set to $1e-5$.

Note that in Figure 4.16, 4.19, 4.20 and 4.21 k_ψ and k_{g2} have the same magnitude. This means that the relationship between coalescence efficiency and breakage efficiency remains unchanged and any alterations to the shape of the volume density distribution is caused by changes in the frequency of coalescence and the frequency of breakage. As discussed above the shape of the volume density distribution does not change as long as changes of equal magnitude are made to k_ω and k_{g1}.

When changes were made only to either k_ω or k_{g1}, as in Figure 4.20 and 4.21

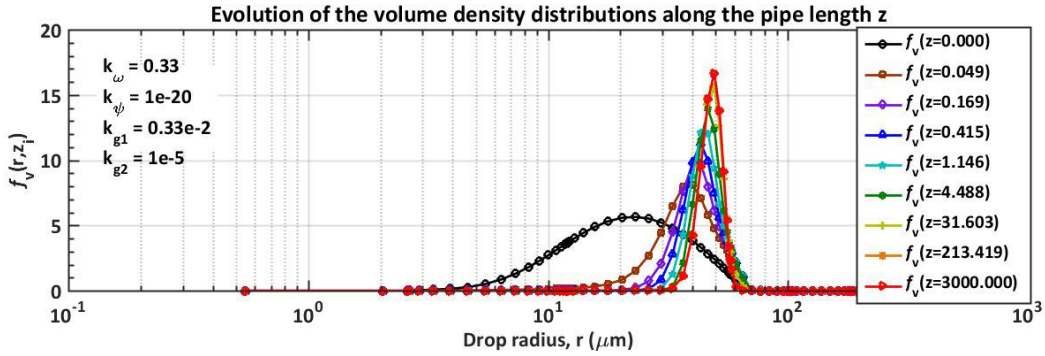

Figure 4.20: Volume density distribution, f_v, plotted at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

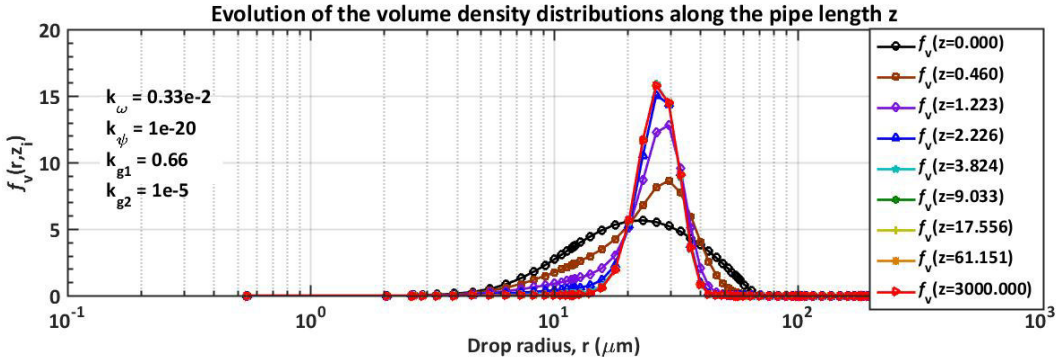

Figure 4.21: Volume density distribution, f_v, plotted at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

respectively, the shape of the volume density distribution was altered. When k_ω is increased it is expected that the volume density distribution should become taller and thinner and skewed towards larger drop sizes as coalescence frequency is then increased in magnitude while breakage frequency remains unchanged. This is exactly what happens, as seen by comparing Figure 4.16 and Figure 4.20, where the final shape of f_v is both taller, narrower and skewed more towards larger drop sizes in Figure 4.20. In Figure 4.21 k_{g1} has been increased compared to Figure 4.16. With an increase in the k_{g1} an increase in breakage frequency would be expected as discussed in section 4.1.2. This should lead to a wider and shorter volume density distribution skewed towards smaller drop sizes, as breakage frequency is now increased in magnitude compared to coalescence frequency. Comparing Figure 4.16 and 4.21 this is indeed the case and f_v at larger z-values appear wider and skewed more towards lower drop sizes in Figure 4.21.

In Figures 4.14-4.21 changes were made in the magnitudes of the coalescence frequency and in the magnitude of breakage frequency. A couple of cases where the magnitude of k_ψ and k_{g2} are changed while k_ω and k_{g1} are kept constant will now be studied.

In Figure 4.22 the volume density distribution, f_v, is plotted as a function of droplet radius, r, at nine different pipe lengths. k_ω is set to 1, k_{g1} is set to 0.01, k_ψ is set to $1e-22$ and k_{g2} is set to $1e-5$.

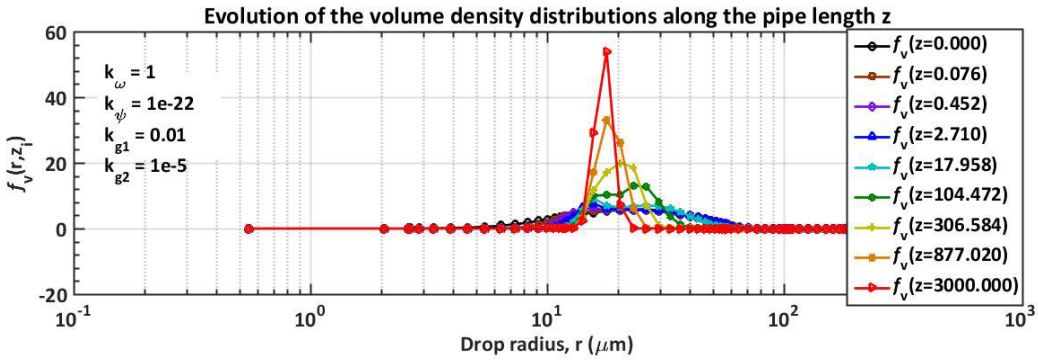

Figure 4.22: Volume density distribution, f_v, plotted at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

In Figure 4.22 breakage appear to be dominant as the distribution is skewed far to the left. k_ψ and k_{g2} are both low which as discussed in Section 4.1.2 should mean high breakage efficiency and low coalescence efficiency. Since k_ω is two orders of magnitude larger than k_{g1} this shows how dominant the influence of coalescence- and breakage efficiency is on f_v compared to the coalescence- and breakage frequency. The sensitivity of change in the magnitude of k_ψ and k_{g2} is also a lot larger than the sensitivity in the change in the magnitude of k_ω and k_{g1}, because k_ψ and k_{g2} contribute to the exponential part of Equation 2.8 and 2.13 respectively.

Figure 4.22 f_v has a concerning spike that keeps growing as z becomes larger. The pointy appearance of the f_v-curve could be due to the distribution of grid points and more grid points in the domain could smooth the curve. f_v is becoming larger and larger because the coalescence efficiency is so low that only very small drops are able to coalesce, while smaller and smaller drops are able to break because breakage efficiency is so high. This is a good indication that the chosen set of parameters are outside the models range of realistic results, which is only

natural when k_ψ are k_{g2} are set to opposite extremes.

In Figure 4.23 the volume density distribution, f_v, is plotted as a function of droplet radius, r, at nine different pipe lengths. k_ω is set to 1, k_{g1} is set to 0.01, k_ψ is set to $1e-19$ and k_{g2} is set to $1e-5$.

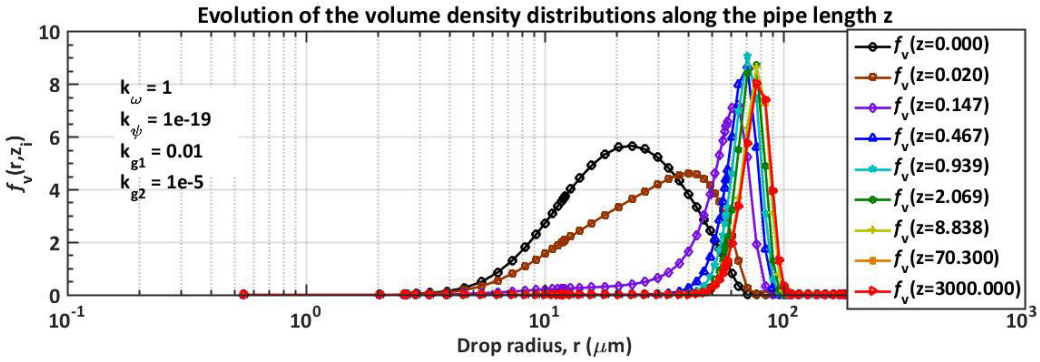

Figure 4.23: Volume density distribution, f_v, plotted at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

In contrast to Figure 4.22 the efficiency of coalescence is larger in Figure 4.23 considering that k_ψ is increased three orders of magnitude. It is evident that coalescence has reached a level that is able to cope with the breakage efficiency creating a shape of f_v more resembling of the case shown in Figure 4.20. Unlike in Figure 4.22 f_v does not have spikes in Figure 4.23, which indicates that it is a more reasonable set of parameters. Coalescence efficiency in Figure 4.23 is large enough that smaller drops are able to coalesce in spite of the high rate of breakage efficiency, creating a more physically realistic result.

In Figure 4.24 the volume density distribution, f_v, is plotted as a function of droplet radius, r, at nine different pipe lengths. k_ω is set to 1, k_{g1} is set to 0.01, k_ψ is set to $1e-22$ and k_{g2} is set to $1e-2$.

In Figure 4.24 k_ψ is small and k_{g2} is large, which should lead to a case of lower coalescence efficiency and lower breakage efficiency than the case in Figure 4.23. In the case shwon in Figure 4.24 both coalescence efficiency and breakage efficiency are so low that larger drops do not break or coalesce at all. This is indicated by the fact that above a certain radius the distribution remains unchanged along the entire pipe length.

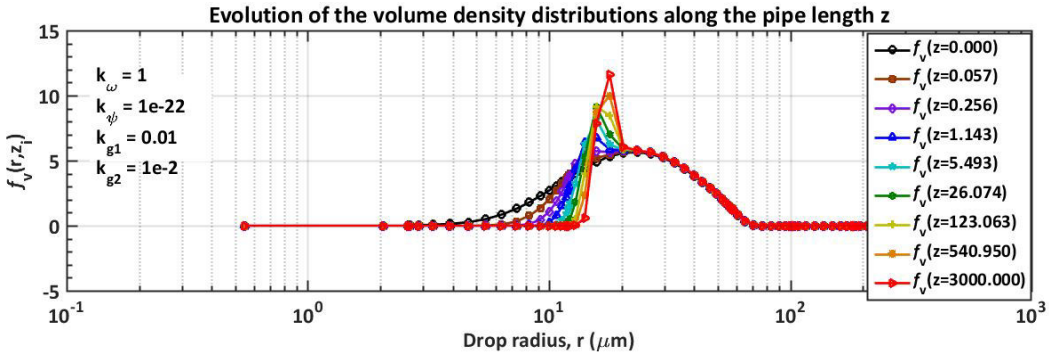

Figure 4.24: Volume density distribution, f_v, plotted at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

In Figure 4.25 the volume density distribution, f_v, is plotted as a function of droplet radius, r, at nine different pipe lengths. k_ω is set to 1, k_{g1} is set to 0.01, k_ψ is set to $1e-19$ and k_{g2} is set to $1e-2$.

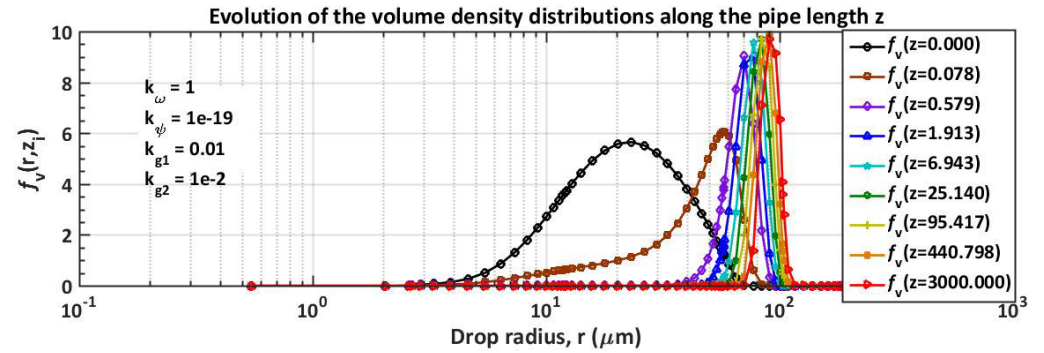

Figure 4.25: Volume density distribution, f_v, plotted at nine different pipe lengths. The fitting parameters that were used are shown at the left side of the plot.

In Figure 4.25 coalescence efficiency is dominant as a lot of large drops are able to coalesce and very few are able to break, on the account that the distribution is tall, narrow and skewed far towards larger radii. This is in accordance with the magnitude of k_ψ and k_{g2} that are both high and should therefore give high efficiency of coalescence and low efficiency of breakage.

4.5 Dynamic Drop Growth Model With The Surfactant Mass Balance

To investigate the dynamics of the system when the mass balance of the surfactant is included, a study has been done to see how the concentration of surfactant in the bulk phase, C, the amount of surfactant adsorbed on the dispersed phase, Γ, and the interfacial tension, γ, changes when the concentration of surfactant at infinite dilution is varied. A couple of new parameters are added to the model in addition to the ones in Table 4.1. These are used in the equations for the mass balance of the surfactant and can be found in Table 4.2. In addition the fitting parameters of coalescence frequency, breakage frequency, coalescence efficiency and breakage efficiency that were used to generate all the results in this section are given in Table 4.2.

Table 4.2: Additional parameters needed for the surfactant mass balance equations and fitting parameter used in the model.

Parameter	Value	Explanation
K	100	Langmuir isotherm equilibrium constant
K_f	$5e-5\left[\frac{\mathrm{m}}{\mathrm{s}}\right]$	Mass transfer coefficient
Γ_{max}	$1e-6\left[\frac{\mathrm{mol}}{\mathrm{m}^2}\right]$	Maximum amount of surfactant that can be adsorbed on the surface of the dispersed phase droplets
T	293.15[K]	Ambient temperature
k_ω	0.66	Fitting parameter for the coalescence frequency expression
k_{g1}	0.66	Fitting parameter for breakage frequency expression
k_ψ	$1e-20$	Fitting parameter for coalescence efficiency expression
k_{g2}	$1e-5$	Fitting parameter for breakage efficiency expression

In Figure 4.26 the surface to volume ratio, α, is plotted as a function of axial coordinate, z, for three different concentrations at infinite dilution, $C_\infty(0.05, 0.10, 0.15)$.

Variations in the magnitude of α are miniscule between the three cases dispelayed in Figure 4.26. This suggest that there is little change in the amount of coalescence and breakage between the cases, since surface area is not conserved under coalescence and breakage. It would therefore be expected to see more of difference

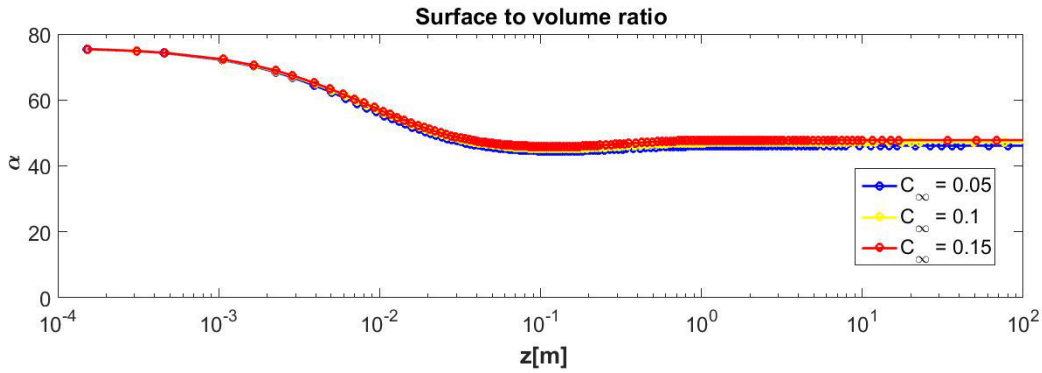

Figure 4.26: Surface to volume ratio, α, plotted as a function of axial position, z, for three different concentrations at infinite dilution, C_∞.

between the three cases if changes in the magnitude of C_∞ had an effect on the dynamics of coalescence and breakage.

Figure 4.27 shows the concentration of surfactant in the bulk phase normalized by the concentration of surfactant at infinite dilution, $\frac{C}{C_\infty}$, plotted as a function of axial position, z, for three different concentrations at infinite dilution, $C_\infty(0.05, 0.1, 0.15)$.

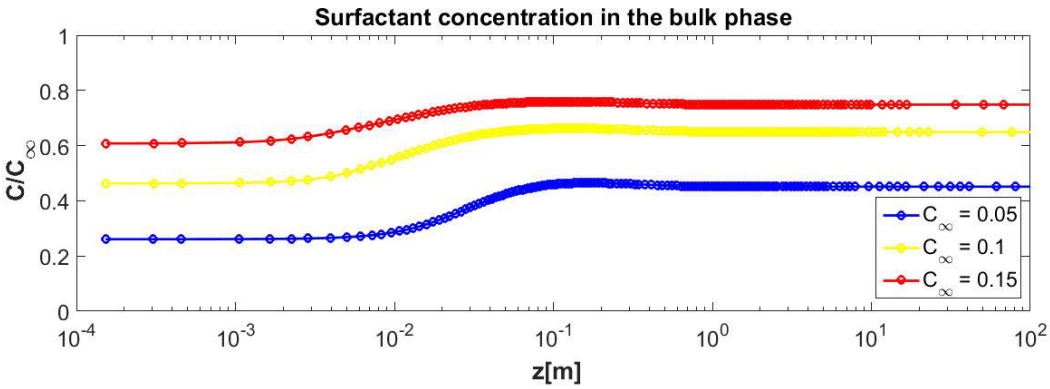

Figure 4.27: Bulk concentration of surfactant normalized by the concentration of surfactant at infinite dilution, $\frac{C}{C_\infty}$, plotted as a function of axial position, z, at three different concentrations of infinite dilution, C_∞.

As seen in Figure 4.27 C increases moderately along the axial direction before settling well below C_∞. This indicates that surfactants are migrating from the dispersed phase and over to the bulk phase. The decline in α seen for all three

cases in Figure 4.26 supports this, because the amount of available surface area relative to the volume is decreasing. This means less surface area of dispersed phase for the surfactant to adsorb on and hence more surfactant ends up in the continuous phase(bulk phase).

In Figure 4.28 the amount of surfactant adsorbed on the dispersed phase interface normalized by the maximum amount of surfactant that can be adsorbed on the dispersed phase interface, $\frac{\Gamma}{\Gamma_{max}}$, is plotted as a function of axial position, z, for three different magnitudes of concentration at infinite dilution, $C_\infty(0.05, 0.10, 0.15)$.

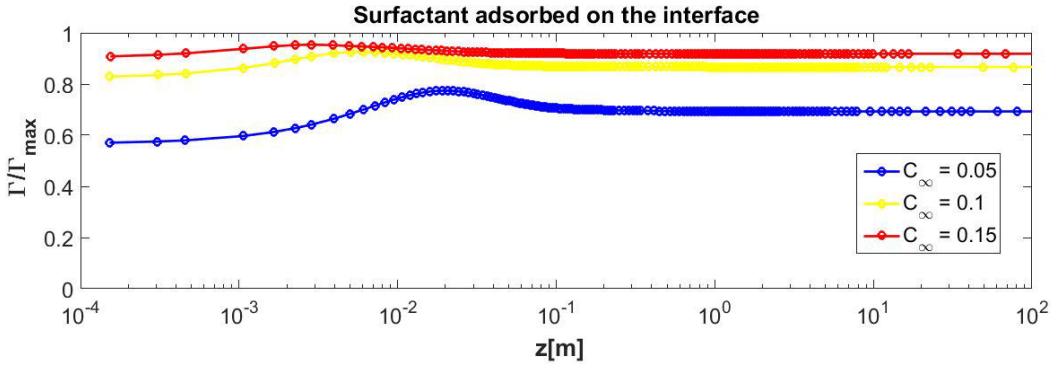

Figure 4.28: Amount of surfactant adsorbed on the surface of the dispersed phase normalized by the maximum amount of surfactant that can adsorb on the dispersed phase, $\frac{\Gamma}{\Gamma_{max}}$, plotted as a function of axial position, z, for three different concentrations at infinite dilution, C_∞.

In neither of the three different cases shown in Figure 4.28 there seem to be large changes in Γ along the axial direction. A slight increase in Γ is present with each different magnitude of C_∞, but the change is very small. This indicates that the surface is almost saturated with surfactant even at lower axial positions, which is also supported by the increase in bulk phase concentration seen in Figure 4.27. An increase in C when α is decreasing, as seen in Figure 4.26, indicates that the surface is saturated. This is because as less surface area is available for adsorption, which is the case when α decreases, more surfactant migrates to the bulk phase.

In Figure 4.29 the interfacial tension of the dispersed phase droplets normalized by the initial interfacial tension of dispersed phase doplets, $\frac{\gamma}{\gamma_0}$, is plotted as a function of axial position, z, for three different magnitudes of concentration at infinite dilution, $C_\infty(0.05, 0.10, 0.15)$.

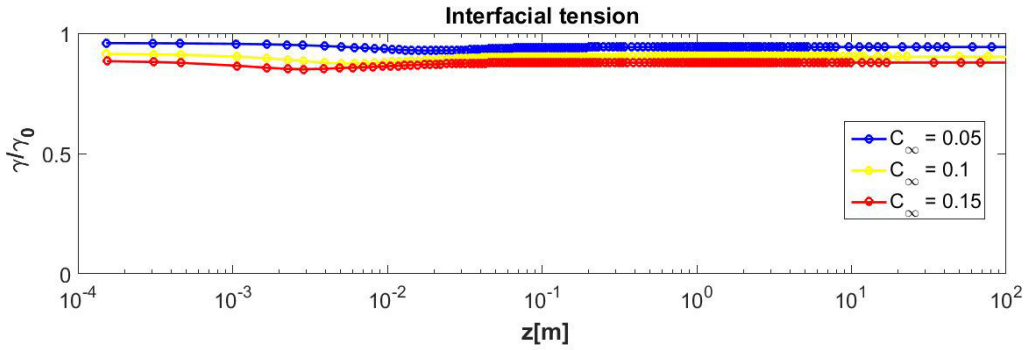

Figure 4.29: Interfacial tension normalized by the initial interfacial tension, $\frac{\gamma}{\gamma_0}$, plotted as a function of axial position, z, for three different concentrations at infinite dilution, C_∞.

The changes in γ are minuscule along the axial direction for all the three magnitudes of C_∞ shown in Figure 4.29. As argued above the dispersed phase droplets are saturated with surfactant, which means that the surfactant is unable to change the surface tension of the dispersed phase droplets to a significant extent. This is an important discovery as the only thing that couples the surfactant mass balance to the number density distribution and volume density distribution is the ratio $\frac{\gamma}{\gamma_0}$, as seen in Equation 2.37 and 2.38. In turn, this means that the ability of the surfactant mass balance to affect the number- and volume density distribution is rather limited with this model formulation.

In Figure 4.30 the average drop size of the volume density distribution, μ_v, is plotted as a function of axial position, r, for three different concentrations at infinate dilution, $C_\infty(0.05, 0.10, 0.15)$, and the case where the mass balance of the surfactant is not included in the model.

As seen by comparing the lines in Figure 4.30 it is evident that including the surfactant mass balance does not have a significant effect on the average drop size evolution. Very small differences in μ_v can be seen as C_∞ is changed. However, Grimes et al.[18] have demonstrated that the presence of surfactant in the solution is expected to have a much larger effect on coalescence and breakage than Figure 4.30 indicates. Because surfactants have a large impact on the surface tension of droplets, one would expect that taking the surfactant mass balance into account in the model would yield quite different results than without it.

One reason that the model is unable to predict the expected dynamics of coalescence and breakage when the mass balance is included, is the way that k_ψ is defined. The retarded Hamaker constant has been lumped into the fitting

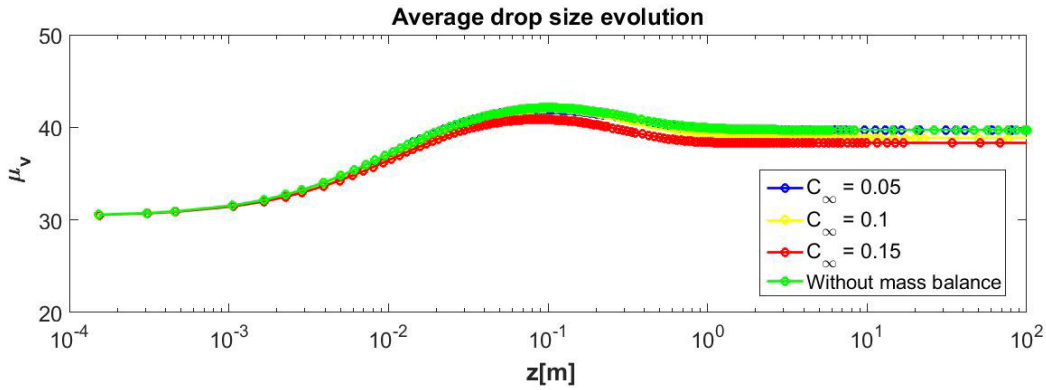

Figure 4.30: Average drop size of the volume density distribution, μ_v, plotted as a function of axial position, z, for three different concentrations at infinite dilution, C_∞, and for the case where the surfactant mass balance is not included in the model.

parameter of coalescence efficiency, k_ψ, thus it has units of m^2. The retarded Hamaker constant describes the interfacial forces between surfactants adsorbed on two opposing interfaces that gives rise to the disjoining pressure and is a strong function of the interfacial surfactant concentration, Γ. Thus, the model as formulated is insufficient to describe the intermolecular forces at the surface of droplets as a function of the surfactant concentration. A more comprehensive model that replaces k_ψ with a Γ dependant functionality of the retarded Hamaker constant should be formulated.

Chapter 5

Conclusion

In this thesis the dynamic evolution of the drop size distribution of emulsions in turbulent pipe flow was studied, with and without the surfactant mass balance. The study was performed for a pipe length of $3000\ \mathrm{m}$, pipe diameter of $0.0254\ \mathrm{m}$, a maximum droplet radius of $600\ \mu\mathrm{m}$ and an average fluid velocity of $0.160\ \frac{\mathrm{m}}{\mathrm{s}}$. First, a base case scenario of the model without the surfactant mass balance was presented to give the reader an overview of how the data from the model is interpreted. A study was then performed to see how the system dynamics changed when k_ω and k_{g1} was varied in between $1e-2$ and 1. This was followed by a similar investigation of how the system dynamics changed when k_ψ was varied between $1e-22$ and $1e-19$ and k_{g2} was varied between $1e-5$ and $1e-2$. The total rate of coalescence, the total rate of breakage and the average volume density distribution was then studied for several different sets of fitting parameters. Lastly, the surfactant mass balance was included in the model to see how it affected the dynamics of the system.

The base case showed that coalescence is strong at lower values of z and gets weaker and weaker along the z-axis. On the other hand, breakage is weaker at lower values of z and grows stronger along the z-axis. It was also determined that the coalescence mass balance ratio M_C was larger than one, but that this should not compromise the integrity of the model notably.

In section 2.2 the system behaviour when changes in magnitude were made to the coalescence and breakage frequency parameters k_ω and k_{g1} were studied and proved that both parameters have a significant effect on the length at which equilibrium is reached, L_∞. However, k_{g1} was found to have a slightly larger effect on L_∞ than k_ω , when k_ψ and k_{g2} where kept constant. Another remarkable

observation was made when k_ω and k_{g1} was varied, namely that $\mu_{v,\infty}$ remained unchanged when changes of equal magnitudes were made to k_ω and k_{g1}. This also indicated that as long as k_ω and k_{g1} were changed with equal magnitudes the only thing able to affect the shape of the distribution were changes in k_ψ and k_{g2}. This is an important discovery to consider when the model is to be fitted with experimental data, since now k_ω and k_{g1} can be merged to one single parameter that controls the axial translation of the distribution.

In Section 2.3 the magnitude of k_ψ and k_{g2} were varied to study the effect that coalescence- and breakage efficiency has on the shape of f_v. It was found that k_{g2} only had an effect on $\mu_{v,\infty}$ at lower magnitudes, which means only at higher efficiencies of breakage. The conclusion is that for the tested range of k_ψ and k_{g2}, the magnitude of efficiency of coalescence had a far larger impact on $\mu_{v,\infty}$ than the magnitude of breakage efficiency had. A change made in the magnitude of k_{g1} showed small differences in the variance of $\mu_{v,\infty}$, leading to the conclusion that k_ψ and k_{g2} are more influential on $\mu_{v,\infty}$ than k_ω and k_{g1} are.

In Section 2.4 the total rate of breakage, the total rate of coalescence and the volume density distribution was studied for different sets of parameters. The total rate of coalescence and the total rate of breakage was found to vary greatly with the magnitude of k_ω and k_{g1}. The shape of f_v remained unchanged when changes of equal magnitude where made to k_ω and k_{g1}, as predicted earlier. When only k_ω was increased the distribution moved towards larger drop sizes and became thinner and taller, as would be expected when the frequency of coalescence increases. Increasing only k_{g1} had the opposite effect and the distribution was skewed towards the left and became wider.

When k_ψ and k_{g2} where changed in magnitude, larger changes occurred in the shape of f_v than it did when k_ω and k_{g1} were changed in magnitude. This is due to the fact that the coalescence- and breakage efficiency parameters are directly affecting an exponential equation, while the coalescence- and breakage frequency parameters only are multiplying factors in the model equations. When both k_ψ and k_{g2} were set to low values, which indicates high breakage efficiency and low coalescence efficiency, f_v was skewed far to the left and got very large in magnitude. Smaller and smaller drops are able to break when breakage efficiency is high and since coalescence efficiency is low, not even the smallest drops will coalesce. At this extreme point where k_{g2} was at its lowest and k_ω was at its lowest, f_v had some spikes that lead to the conclusion that the parameters were moving outside the range where the model gives reasonable results.

All the other changes that where made to k_ψ and k_{g2} gave the expected results in terms of the shape of f_v. High coalescence frequency and high breakage frequency lead to a distribution that narrowed and became slightly taller. With high coalescence efficiency and low breakage efficiency the distribution grew even higher, because larger drops were unable to break. The case where both coalescence frequency and breakage frequency was low gave a case were a lot of droplets were unable to break or coalesce.

Overall, the changes that were made in k_ω, k_{g1}, k_ψ and k_{g2} gave the changes that were expected in f_v. This is a clear sign that the model is responding like it should according to the equations in the model formulation. It also became clear as was discovered earlier that k_{g2} and k_ψ have a far larger effect on the shape of the volume density distribution than k_ω and k_{g1} have.

The effect of the mass balance on the volume density distribution and the number density distribution were found to be minuscule at best. From simulations made with different magnitude of C_∞ it was clear that the interfacial surface tension did not change notably. Considering that the interfacial surface tension is the only thing coupling the surfactant mass balance to the evolution of the number- and volume density distribution, no major changes are expected with the mass balance incorporated. The addition of the mass balance of the surfactant is therefore ineffective as the model is formulated.

One more issue with the model was confirmed in regards to the fitting parameter of coalescence efficiency. The retarded Hamaker constant has been lumped into k_ψ, which rendered the model unfitted to describe the intermolecular forces at the surface of droplets as a function of the surfactant concentration. This lead to the conclusion that a more comprehensive model were k_ψ is replaced by a Γ dependent functionality of the retarded Hamaker constant should be formulated. This warrants further study and should be the focus of future research on this topic.

Bibliography

[1] Einstein, A. *On the theory of the Brownian movement.* Annalen der Physik. (1906)

[2] Smoluchowski, M.V. *Versuch einer mathmatischen Theorie der Koagulationskinetik kolloider Lösnugen.* Phys. Chem. 92(1917) 129.

[3] Landau, L. and Rumer, G. *The Cascade Theory of Electronic Showers.* Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences(1938), 166, 213-228.

[4] Saffman, P.G. and Turner, J.S. *On the collision of drops in turbulent clouds* Fluid Mechanics 1(1956) 16.

[5] Hulburt, H.M. and Katz, S.(1964) *Some problems in particle technology. A statistical mehcanical formulation.* Chemical Engineering Science, 19, 555-574.

[6] Grimes, B.A.(2014) *Drop Growth Model III* Internal Report to Statoil.

[7] Grimes, B.A.(2013) *Drop Growth Model II* Internal Report to Statoil.

[8] Coulaloglou, C.A. and Tavlarides, L.L.(1977) *Description of Interaction Processes in Agitated Liquid-Liquid Dispersions.* Chemical Engineering Science, 32, 1289.

[9] Hsia, M.A. and Tavlarides, L.L.(1980) *A Simulation Model for Homogeneous Dispersions in Stirred Tanks.* Chemical Engineering Journal, 20, 225.

[10] Azizi, F. and Al Taweel, A.M.(2011) *Turbulently Flowing Liquid Liquid Dispersions. Part I: Drop Breakage and Coalescence.* Chemical Engineering Journal, 166, 715.

[11] Khatibi, M.(2013) *Experimental Study on Droplet Size of Dispersed Oil-Water Flow.* Masters thesis, Norwegian University of Science and Technology, NTNU, Trondheim.

[12] Pope, S.B.(2003) *Turbulent Flows.* Cambridge University Press, Cambridge.

[13] Grimes, B.A.(2016) *TKP4115 - SimLecture 01* Slides used in the course TKP4115, P. 10.

[14] Grimes, B.A.(2016) *TKP4115 - SimLecture 01* Slides used in the course TKP4115, P. 10.

[15] Eaton, J.W.(1996) Copyright (C), Matlab code distributed during the course TKP6, Chemical Engineering Department, NTNU, Autumn 2015.

[16] Schömann, H. (2016), Personal correspondence with Heiner.

[17] Dorao, C.A.(2005) *Ph.D. Dissertation*, Dept. Chemical Engineering, NTNU, Trondheim, 2005

[18] Grimes, B.A. Dorao, C.A. Simon, S. Nordgård, E.L. Sjøblom, J.(2010) *Analysis of dynamic surfactant mass transfer and its relationship to the transient stabilization of coalescing liquid liquid dispersions*, Journal of Colloid and Dispersion Chemistry, 348, 479.

[19] Ramkrishna D. *Population Balances : Theory and Applications to Particulate Systems in Engineering* Academic Press: San Diego, 2000

[20] Solsvik, J. *Introduction to the orthogonal collocation method - Basic theory and Matlab examples*, Chemical Engineering Department, TKP6, autumn 2015.

Appendix A - Non Dimenzionalizing the Model

To generate a stable numerical solution the model equations have to be non-dimenzionalized before they are solved in Matlab[6]. Five dimensionless variables will be introduced into the equations to achieve this, namely dimensionless axial position in the pipe, dimensionless radius of droplets, dimensionless number density distribution, dimensionless concentration of surfactant in the continuous phase and dimensionless amount of surfactant adsorbed on the dispersed phase interface [6]

$$\lambda = \frac{z}{L} \equiv \text{dimensionless axial position in the pipe} \tag{A-1}$$

$$\xi = \frac{r}{R_{max}} \equiv \text{dimensionless radius of the number density distribution} \tag{A-2}$$

$$f_n = \frac{R_{max}}{N_{d0}}\eta \equiv \text{dimensionless number density distribution} \tag{A-3}$$

$$\theta = \frac{C_f}{C_{f,\infty}} \equiv$$
$$\text{dimensionless concentration of surfactant in the continuous phase} \tag{A-4}$$

$$\tau = \frac{\Gamma}{\Gamma_{max}} \equiv$$
$$\text{dimensionless amount of surfactant adsorbed on the dispersed phase} \tag{A-5}$$

where N_{d0} is the number density at the initial position in the pipe(z=0). It can be found by integrating the initial number density, η_0, over the entire radius domain[6]

$$N_{d0} = \int_0^{R_{max}} \eta_0(r')\mathrm{d}r' \tag{A-6}$$

The overall population balance equation in dimensionless form is[6]

$$\frac{df_n}{d\lambda} = P_{C,+} - P_{C,-} + P_{B,+} - P_{B,-} \quad \text{for} \quad 0 \leq \gamma \leq 1, \quad 0 \leq \xi \leq 1 \tag{A-7}$$

Initial condition: at $\lambda = 0, \quad f_n(\xi, 0) = f_{n0}(\xi), \quad \text{for} \quad 0 \leq \xi \leq 1$ (A-7.1)

where $P_{C,+}$ is the dimensionless birth rate due to coalescence, $P_{C,-}$ is the dimensionless death rate due to coalescence, $P_{B,+}$ is the dimensionless birth rate due to breakage and $P_{C,-}$ is the dimensionless death rate due to breakage. The birth and death rate due to coalescence, on dimensionless form are, respectively[6]

$$P_{C,+}(\xi,\lambda) = \int_0^{\xi/\sqrt[3]{2}} \overline{r}_C(\xi', \left[\xi^3 - \xi'^3\right]^{1/3}) f_n(\xi',\lambda) \times f_n(\left[\xi^3 - \xi'^3\right]^{1/3}, \lambda)\, \mathrm{d}\xi' \tag{A-8}$$

$$P_{C,-} = f_n(\xi,\lambda) \int_0^1 \overline{r}_C(\xi,\xi') f_n(\xi',\lambda) \mathrm{d}\xi' \tag{A-9}$$

where $\overline{r}_C(\xi_1, \xi_2)$ is the dimensionless rate of coalescence of two drops with radius ξ_1 and ξ_2 given by the following expression[6]

$$\overline{r}_C(\xi_1,\xi_2) = \chi_\omega(\xi_1,\xi_2)^2 \left[\xi_1^{2/3} + \xi_2^{2/3}\right]^{1/2} exp\left[-\chi_\psi \left(\frac{\xi_1\xi_2}{\xi_1+\xi_2}\right)^4\right] \tag{A-10}$$

where χ_ω and χ_ψ are respectively[6]

$$\chi_\omega = k_\omega \frac{4\sqrt[3]{2}\epsilon^{1/3} R_{max}^{7/3} N_{d0} L}{U(1+\phi)} \tag{A-11}$$

$$\chi_\psi = \frac{1}{k_\psi} \frac{16\mu_c\rho_c\epsilon R_{max}^4}{\lambda^2(1+\phi)^3} \tag{A-12}$$

The value of χ_ω represents the ratio between residense time for a drop in the pipe to the average time between droplet collisions and the value of χ_ψ is the ratio between the film drainage time constant and the droplet collision time constant.
The birth rate of breakage, $P_{C,+}$, and death rate of breakage, $P_{C,-}$, on dimensionless form are respectively[6]

$$P_{B,+}(\xi,\lambda) = \int_\xi^1 2\overline{\beta}(\xi,\xi')\overline{g}(\xi') f_n(\xi',\lambda)\mathrm{d}\xi' \tag{A-13}$$

$$P_{B,-}(\xi,\lambda) = \overline{g}(\xi) f_n(\xi,\lambda) \tag{A-14}$$

where $\overline{\beta}(\xi, \xi')$ is the dimensionless daughter size distribution and $\overline{g}(\xi)$ is the dimensionless rate of breakage, and they are respectively[6]

$$\overline{\beta}(\xi, \xi') = \frac{45}{\sqrt[3]{2}} \frac{\xi^2}{\xi'^3} \left(\frac{\xi^3}{\xi'^3}\right)^2 \left[1 - \left(\frac{\xi^3}{\xi'^3}\right)^2\right] \tag{A-15}$$

$$\overline{g}(\xi) = \frac{\chi_{g1}}{\xi^{2/3}} exp\left[-\frac{\chi_{g2}}{\xi^{5/3}}\right] \tag{A-16}$$

where χ_{g1} and χ_{g2} are respectively[6]

$$\chi_{g1} = k_{g1} \frac{\epsilon^{1/3} L}{\sqrt[3]{4}(1+\phi) R_{max}^{2/3} U} \tag{A-17}$$

$$\chi_{g2} = k_{g2} \frac{\gamma(1+\phi)^2}{\sqrt[3]{32}\rho_d \epsilon^{2/3} R_{max}^{5/3}} \tag{A-18}$$

The dimensionless number described by χ_{g1} represents a combination of the droplet residence time in the pipe to the breakage frequency of the drop in the turbulent flow field. The dimensionless number described by χ_{g2} represents the ratio of the surface energy of the drop to the mean turbulent kinetic energy of an eddy.

The dimensionless form of the bulk surfactant mass balance is given by the following expression

$$\frac{d\theta}{d\lambda} = E\overline{\alpha}\left[\theta - \frac{\tau}{C_\infty K(1-\tau)}\right] \quad \text{at} \quad \lambda = 0, \quad \theta = \frac{C_0}{C_\infty} \tag{A-19}$$

where E is given by

$$E = \frac{K_f L}{(1-\phi)U} \tag{A-20}$$

and the dimensionless form of the ratio between the dispersed phase area and the total dispersion volume is

$$\overline{\alpha} = \frac{1}{\phi} \int_0^1 \frac{3}{\xi'} f_v(\xi')\, \mathrm{d}\xi' \tag{A-21}$$

The non-dimensional form of the change in amount of surfactant adsorbed on the surface of the dispersed phase is given by

$$\frac{d\tau}{d\lambda} = F\left[\theta - \frac{\tau}{C_\infty K(1-\tau)}\right] - \frac{\tau}{\overline{\alpha}} \frac{d\overline{\alpha}}{d\lambda} \quad \text{at} \quad \lambda = 0, \quad \theta = \frac{\Gamma_0}{\Gamma_{max}} \tag{A-22}$$

where F is given by

$$F = \frac{K_f C_\infty L}{\Gamma_{max} U} \tag{A-23}$$

and the dimensionless form of the derivative of the ratio between the dispersed phase area and the total dispersion volume is

$$\frac{d\overline{\alpha}}{d\lambda} = \frac{1}{\phi} \int_0^1 \frac{3}{\xi'} \frac{df_v(\xi')}{d\lambda} \, \mathrm{d}\xi' \tag{A-24}$$

Appendix B - The Collocation Method

Collocation is a method under the category of weighted residual methods. Weighted residual methods are based on expanding the solution of a differential equation, or a set of differential equations, into a set of finite functions[19]. These equations are normally obtained by a truncated orthonormal polynomial series. For the case of a two-dimensional equation, or set of equations varying with time t and internal coordinate x (could be radius, mass etc.) the expansion looks like[19]

$$f(x,t) \simeq h(x) + \sum_{n=0}^{M} C_n(t)\phi_n(x) \tag{B-1}$$

where h is the initial condition of f, C_n are the unknown coefficients and ϕ_n are the functions that the wanted solution is expanded in terms of. ϕ_n can be chosen to be a number of functions including Lagrangian polynomials, Laguerre functions and Legendre polynomials. It is the residual of these expansions that are minimized to find the solution of the differential equation. This is done by taking the integral of the residual function multiplied by an appropriate weighting function and equate it to 0[19]

$$\int_0^\infty Re(f)\psi_j(x)\,\mathrm{d}x = 0 \tag{B-2}$$

where Re is the residual of the expansion function in Equation B-1 and ψ_j is the weighting function. Weighting functions may be chosen differently depending on what specific weighted residual method is being used.

In the orthogonal collocation method the solution domain of the differential equations is discretized into nodes, where the accuracy of the solution depends on how many nodes there are in the domain. If steeper gradients are encountered in certain parts of the domain, more collocation points can be generated here. Usually, this is done by splitting the domain into several separate regions and performing collocation on each region. The family of Jakobi othogonal polynomials can be used to decide the distribution of collocation points. The domain of these polynomials are given by[20]

$$\mathbf{J}_P^{a,b} = (J_P^{\alpha,\beta}(x), p = 0, 1, 2, ...) \tag{B-3}$$

where $(J_P^{a,b})_{p=0,1,2,..}$ is a system of algebraic polynomials with degree p. a and b are weighting parameters that decide the relative displacement of the roots of the

polynomial. The degree of the polynomial decides the amount of roots and therefore the amount of nodes. The basis functions that the wanted solution is expanded in terms of, shown as $\phi_n(x)$ in Equation B-1 can be chosen to be Lagrangian polynomials. Lagrangian polynomials are given by

$$L_{P,j}(x) = \prod_{i=1}^{P} \frac{(x - x_i)}{x_j - x_i} \tag{B-4}$$

where $L_{P,j}(x)$ is 1 if $x = x_i = x_j$ and $L_{P,j}(x)$ is 0 if $x = x_i \neq x_j$. In the case of orthogonal collocation the weighting function in Equation B-2 is chosen to be the Dirac-Delta function. This is the reason for the simplicity of the orthogonal collocation method compared to other weighted residual methods. The Dirac-Delta function is a function that is zero everywhere on the real domain except for at the origin.

Appendix C - Matlab Code

main

```
clear all
clc
close all
tic;
% Calling the script containing all the constants
parameters
global Ne Ni Nd0
[r_scale,mew_scale,Nd0] = rescale('InitialDistribution.txt',0,0.34);

 [end_vec, ele_vec, rts,wts,P,N_inter] = SetUpGrid(Ne,Ni,mew_scale...
     ,r_scale);

[C0,G0,alfa0] = alphacalc(N_inter,P,wts,ele_vec);

% Setting options for ODE-solver
options = odeset('RelTol',10^-3,'AbsTol',10^-6);
% Runing ODE-solver
[z,fn] = ode15s(@(z,fn)differentialeq(z,fn,P,wts,ele_vec),...
          [0 1],[N_inter,C0,G0],options)

toc;
```

parameters

```
% Definng global variables used in scripts and functions
global Ne Ni phi U L Di rho_d mu_d rho_c mu_c...
    gamma k_o k_p k_g1 k_g2 Rmax Nibc Vmax K Gmax Cinf Kf G F R T
Ne = 6; % Number of elements
Ni = [5,15,15,15,5,5]; % Number of internal colloc points for each element
Nibc = [7,16,16,16,6,6]; % Number of colloc points with boundary
% points added
phi = 0.34; % Water cut
U = 0.16; % Velocity of fluid [m/s]
L = 300; % Length of pipe [m]
Di = 0.0254; % Pipe diameter [m]
rho_d = 1023; % Density of dispersed phase [kg/m3]
mu_d = 1E-3; % Viscosity of dispersed phase [Pa*s]
```

```
rho_c = 786; % Density of the continous phase [kg/m3]
mu_c = 1.3E-3; % Viscosity of the continous phase [Pa*s]
gamma = 50E-3; % Interfacial tension [N/m]
k_o = 1E-1; % Fitting parameter for coalescence frequency expression
k_p = 1E-19; % Fitting paramter for coalescence efficiency expression
k_g1 = 1E-1; % Fitting parameter for breakage frequency expression
k_g2 = 1E-4; % Fitting parameter for breakage efficiency expression
Rmax = 300E-6; % Highest radius of measure [m]
Vmax = Rmax^3*(4*pi()/3); % Maximum drop volume [m^3]

K = 100; % Konstant from langmuir equation [m^3/mol]
Gmax = 1E-6; % Maximum mols of surfactant adorbed on surface [mol/m^2]
Cinf = 0.1; % Concentration at infinate dilution [mol/m^3]
Kf = 5E-5; % Constant for surfactant in contninous phase [m/s]
R = 8.314; % Universal gas constant [m^3*Pa/mol*K]
T = 293.15; % Temperature in Kelvin
% Gathering of constants
G = (L*Kf)/((1-phi)*Rmax*U);
F = (Kf*L*Cinf)/(Gmax*U);
```

rescale

```
% Function that takes in a textfile with radius and number distribution
% , either n=0 if volume distribution og n=1 if number distribution and
% water cut phi. Distribution is then scaled appropriately yielding a
% vector of r values and values of distribution.
function [r_scale,mew_scale,Nd0] = rescale(textfile,n,phi)
global Rmax
% Open file for reading
fileID = fopen(textfile,'r');
% Chosing reading format
formatspec = '%f %f';
sizeA = [2 Inf];
data = fscanf(fileID,formatspec,sizeA);
data = data';
% Rescaling r into meters
for i = 1:length(data)
    data(i,1) = data(i,1)*1.0e-6;
end
 int1 = 0;
 nData = length(data);
```

```
if n == 0;
      % Scale raw vol dist data to make the integral equal phi
      integral = 0;
      for i = 2:nData
          integral = integral + (data(i,1)-data(i-1,1))*((data(i-1,2)...
              +data(i,2))/2);
      end
      data(:,2) = data(:,2)*phi/integral;
      % Scaling distribution with volume
      for i = 1:nData
          data(i,2) = data(i,2)*(3/(4*pi()*data(i,1)^3));
      end
      % Calculate Nd0
      Nd0 = 0;
      for i = 2:nData
          Nd0 = Nd0 + (data(i,1)-data(i-1,1))*((data(i-1,2)...
              +data(i,2))/2);
      end
      % Expanding largest radius
      maxvec = max(data);
      if Rmax >= maxvec(1);
         app = linspace(maxvec(1),Rmax,10)';
         r_scale = [data(:,1);app(2:10)];
         mew_scale = [data(:,2); zeros(9,1)];
         nData = nData+9;
      else
      r_scale = [data(:,1)];
      mew_scale = [data(:,2)];
      end
      % Non-dimentionalization
      for i = 1:nData
          r_scale(i) = r_scale(i)/Rmax;
          mew_scale(i) = mew_scale(i)*(Rmax/Nd0);
      end

elseif n == 1;
     % Multiplying distribution with volume before integrating
     data2 = data;
     for i = 1:nData
         data2(i,2) = data(i,2)*(((4*3.14)/3.0)*data(i,1)^3);
     end
```

```
61
62      for i = 2:nData
63           int1 = int1 + (data2(i,1)-data2(i-1,1))*((data2(i-1,2)...
64               +data2(i,2))/2.0);
65      end
66      % Scalling distribution with phi/(1st moment with respect to v)
67      for i = 1:nData
68           data(i,2) = data(i,2)*(phi/int1);
69      end
70
71      % Calculate Nd0
72       Nd0 = 0;
73       for i = 2:nData
74           Nd0 = Nd0 + (data(i,1)-data(i-1,1))*((data(i-1,2)...
75               +data(i,2))/2);
76       end
77      % Non dimensionalization
78      maxvec = max(data);
79      % r
80      for i = 1:nData
81          data(i,1) = data(i,1)/maxvec(1);
82      end
83      % mew
84      for i = 1:nData
85          data(i,2) = data(i,2)*maxvec(1)/Nd0;
86      end
87   r_scale = data(:,1);
88   mew_scale = data(:,2);
89  else
90      'Plug in either 0 for volume distribution or 1 for number distribution'
91
92  end
93
94  end
```

SetUpGrid

```
1  % Function that takes in properties of distribution, number of wanted
2  % colloc elements and number of elements in each colloc domain. The
3  % function also takes in key values of the distribution and out comes
4  % a scaled vector of roots, an unscaled vector of roots and a vector with
```

```
% corresponding quadrature weights.

% Ne = number of elements for collocation
% Ni = array of number of collocation points for each element
% Rf = radius at first non-zero value of distribution
% Rp = radius at highest value of distribution
% Rc = radius at last non-zero value of distribution
% Rmax = max radius of scaled variable (=1)
function [end_vec, ele_vec, rts,wts,P,N_inter]=...
    SetUpGrid(Ne,Ni,mew_scale,r_scale)
% Values of r at reference points for elements
idx = find(mew_scale); % Finding idecies of non-zero values of mew_scale
Rf = r_scale(idx(1));
Rc = r_scale(idx(length(idx)));
idx2 = find(mew_scale==max(mew_scale)); % Finding indecies of max value
% of mew_scale
Rp = r_scale(idx2);
Rm = 1;
end_vec = []; % Creating empty vector to put domain end points in.

if Ne == 1
    end_vec = [0,1];
elseif Ne == 2
    end_vec = [0,Rc,Rf];
elseif Ne == 3
    end_vec = [0,Rf,Rc,Rm];
elseif Ne == 4
    end_vec = [0,Rf,Rp,Rc,Rm];
elseif Ne == 5
    end_vec = [0,Rf,Rp,Rc,Rc*2,Rm]
elseif Ne > 5
    end_vec = zeros(Ne+1,1);
    end_vec(1) = 0;
    end_vec(2) = Rf;
    end_vec(3) = Rp;
    end_vec(4) = Rc;
    end_vec(5) = Rc*2;
    for i = 6:Ne+1
        end_vec(i)= end_vec(i-1)+((Rm-Rc*2)/(Ne-4));
    end

```

```
end

ele_vec = zeros(Ne,1); % Creating empty vector for lengths of elements
rts = zeros(sum(Ni)+Ne+1,1); %Creating empty vector to store roots
% from collocation call.
wts = zeros(sum(Ni)+Ne+1,1); % Creating empty vector to store roots from
% collocation.
% Creating index maping functions.
    function  [x] = maplower(n)
        if n == 1
        x = 1;
        else
        x = sum(Ni(1:(n-1)))+(n+1);
        end
    end
    function [y] = mapupper(n)
        y = sum(Ni(1:(n-1)))+(n-1) + Ni(n)+2;
    end

for n=1:Ne
    nx1 = maplower(n);
    nx2 = mapupper(n);
    if n == 1
        ele_vec(n) = end_vec(n+1) - end_vec(n);
        [r,A,B,q] = colloc(Ni(n),1,1);
        rts(1:Ni(n)+2) = r;
        wts(1:Ni(n)+2) = q;
    else
        ele_vec(n) = end_vec(n+1) - end_vec(n);
        [r,A,B,q] = colloc(Ni(n),1,1);
        rts(nx1:nx2)= r(2:length(r));
        wts(nx1:nx2)= q(2:length(q));
    end
end

%Making vector where the roots are scaled from 0 to 1 over the whole
%domain.

P = zeros(1,length(rts));
for n=1:Ne
    nx1 = maplower(n);
```

```
87      nx2 = mapupper(n);
88      if n == 1
89          P(1:Ni(n)+2) = rts(1:Ni(n)+2)*ele_vec(n);
90      else
91          P(nx1:nx2) = end_vec(n) + rts(nx1:nx2)*ele_vec(n);
92      end
93  end
94   % Using cubic spline to find the distribution values at the newly
95   % scaled P vectors data points.
96
97   N_inter = interp1(r_scale,mew_scale,P,'pchip');
98
99
100  end
```

colloc

```
1   function [r, A, B, q]=colloc(n,left,right)
2   % colloc: Calculate collocation weights
3   %         [r, A, B, q] = colloc( n ['left'] ['right'])
4   % inputs:
5   %               n - number of interior node points
6   %          'left' - include left boundary
7   %         'right' - include right bounary also
8   % outputs:
9   %          r - vector of roots
10  %          A - Matrix of first derivative weights
11  %          B - Matrix of second derivative weights
12  %          q - Quadrature weights.
13  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
14  % Copyright (C) 1996, 1997 John W. Eaton
15  %
16  % This program is free software; you can redistribute it and/or modify
17  % it under the terms of the GNU General Public License as published by
18  % the Free Software Foundation; either version 2, or (at your option)
19  % any later version.
20  %
21  % This program is distributed in the hope that it will be useful, but
22  % WITHOUT ANY WARRANTY; without even the implied warranty of
23  % MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
24  % General Public License for more details.
```

```
%
% You should have received a copy of the GNU General Public License
% along with Octave; see the file COPYING.  If not, write to the Free
% Software Foundation, 59 Temple Place - Suite 330, Boston, MA
% 02111-1307, USA.
%
% Adapted from Octave's colloc.cc by Steve Swinnea.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

n0 = 0 ; n1 = 0;
if (nargin > 1)
  if (strcmp(left,'left') | strcmp(left,'l') )
    n0 = 1;
  elseif (left == 0 | left == 1 )
    n0 = left;
  else
    error('Second argument should be the string left or l')
  end
end
if (nargin > 2)
  if (strcmp(right,'right') | strcmp(right,'r') )
    n1 = 1;
  elseif ( right == 1 | right == 0 )
    n1 = right;
  else
    error('Third argument should be the string right or r')
  end
end

[dif1,dif2,dif3,r]=jcobi(n,n0,n1,0,0);
q = dfopr(n,n0,n1,0,3,dif1,dif2,dif3,r);
for i=1:(n+n0+n1)
    vect = dfopr(n,n0,n1,i,1,dif1,dif2,dif3,r);
    A(i,:) = vect';
end
for i=1:(n+n0+n1)
    vect = dfopr(n,n0,n1,i,2,dif1,dif2,dif3,r);
    B(i,:) = vect';
end

%%%%%%  jcobi  %%%%%%%
```

```
function [dif1,dif2,dif3,root]=jcobi(n,n0,n1,alpha,beta)
if (n0 ~= 0) & (n0 ~= 1)
    error('** VILERR : Illegal value %  N0 ');
end
if (n1 ~= 0) & (n1 ~= 1)
    error('** VILERR : Illegal value for N1 ');
end
if (n+n0+n1 < 1)
    error('** VILERR : Number of interpolation points less than 1');
end
%
% -- FIRST EVALUATION OF COEFFICIENTS IN RECURSION FORMULAS.
% -- RECURSION COEFFICIENTS ARE STORED IN DIF1 AND DIF2.
%
nt = n+n0+n1;
dif1=zeros(nt,1);
dif2=zeros(nt,1);
dif3=zeros(nt,1);
root=zeros(nt,1);
ab = alpha+beta;
ad = beta-alpha;
ap = beta*alpha;
dif1(1) = (ad/(ab+2)+1)/2;
dif2(1) = 0;

if (n >= 2)
    for i=2:n
        z1 = i-1;
        z = ab + 2*z1;
        dif1(i) = (ab*ad/z/(z+2)+1)/2;
        if (i == 2 )
            dif2(i) = (ab+ap+z1)/z/z/(z+1);
        else
            z = z*z;
            y = z1*(ab+z1);
            y = y*(ap+y);
            dif2(i) = y/z/(z-1);
        end
    end
end
%
```

```
% -- ROOT DETERMINATION BY NEWTON METHOD WITH SUPPRESSION OF
% -- PREVIOUSLY DETERMINED ROOTS
%
x = 0;
for i=1:n
  z = 1;
  while ( abs(z) > 1e-9 )
    xd = 0;
    xn = 1;
    xd1 = 0;
    xn1 = 0;
    for j=1:n
        xp = (dif1(j)-x)*xn - dif2(j)*xd;
        xp1 = (dif1(j)-x)*xn1 - dif2(j)*xd1 - xn;
        xd = xn;
        xd1 = xn1;
        xn = xp;
        xn1 = xp1;
    end
    zc = 1;
    z = xn/xn1;
    if ( i ~= 1 )
        for j = 2:i
            zc = zc - z/(x-root(j-1));
        end
    end
    z = z/zc;
    x = x-z;
  end
  root(i) = x;
  x = x +.0001;
end
%
% -- ADD INTERPOLATION POINTS AT X = 0 AND/OR X = 1
%
if (n0 ~= 0)
    root = [ 0 ; root(1:nt-1) ];
end
if (n1 == 1)
    root(nt) = 1;
end
```

```
148 [dif1 dif2 dif3] = dif( root );
149
150 %%%%% dfopr %%%%%%
151 function vect = dfopr(n,n0,n1,i,id,dif1,dif2,dif3,root)
152 nt = n+n0+n1;
153 vect = zeros(nt,1);
154 if (n0 ~= 0) & (n0 ~= 1)
155     error('** VILERR : Illegal value %  N0 ');
156 end
157 if (n1 ~= 0) & (n1 ~= 1)
158     error('** VILERR : Illegal value for N1 ');
159 end
160 if (nt < 1)
161     error('** VILERR : Number of interpolation points less than 1');
162 end
163 if (id ~= 1 & id ~= 2 & id ~= 3 )
164     error('** VILERR : Illegal ID in DFOPR ')
165 end
166 if ( id ~= 3 )
167     if ( i < 1 )
168         error('** VILERR : Index less than zero in DFOPR ')
169     end
170     if ( i > nt )
171         error('** VILERR : Index greater than NTOTAL in DFOPR ')
172     end
173 end
174
175 %
176 % -- EVALUATE DISCRETIZATION MATRICES AND GAUSSIAN QUADRATURE
177 % -- WEIGHTS.  QUADRATURE WEIGHTS ARE NORMALIZED TO SUM TO ONE.
178 %
179 if ( id ~= 3 )
180     for j = 1:nt
181         if (j == i)
182             if (id == 1)
183                 vect(i) = dif2(i)/dif1(i)/2;
184             else
185                 vect(i) = dif3(i)/dif1(i)/3;
186             end
187         else
188             y = root(i)-root(j);
```

```
189             vect(j) = dif1(i)/dif1(j)/y;
190             if (id == 2 )
191                 vect(j)=vect(j)*(dif2(i)/dif1(i)-2/y);
192             end
193         end
194     end
195 else
196     y=0;
197     for j = 1:nt
198         x = root(j);
199         ax = x*(1-x);
200         if (n0 == 0)
201             ax = ax/x/x;
202         end
203         if (n1 == 0)
204             ax = ax/(1-x)/(1-x);
205         end
206         vect(j) = ax/dif1(j)^2;
207         y = y + vect(j);
208     end
209     vect = vect/y;
210 end
211
212 %%%%% dif %%%%%
213 function [dif1,dif2,dif3] = dif( root )
214 nt = length( root );
215 dif1 = zeros(nt,1);
216 dif2 = zeros(nt,1);
217 dif3 = zeros(nt,1);
218 if ( nt < 1 )
219     error('** VILERR : Number of interpolation points less than 1');
220 end
221 for i = 1:nt
222     x = root(i);
223     dif1(i) = 1;
224     dif2(i) = 0;
225     dif3(i) = 0;
226     for j = 1:nt
227         if ( j ~= i)
228             y = x - root(j);
229             dif3(i) = y*dif3(i) + 3*dif2(i);
```

```
            dif2(i) = y*dif2(i) + 2*dif1(i);
            dif1(i) = y*dif1(i);
        end
    end
end
```

alphacalc

```
% Function that takes in the initial distribution, P, the weights and the
% vector of elements and calculates the initial conditions C0 and G0(mass
% balance).
function [C0,G0,alfa0] = alphacalc(fn,P,wts,ele_vec)
global Nibc Gmax K Cinf Ne phi Nd0 Vmax

% Surface to volume ratio and its derivative
fv0 = zeros(1,length(P));
for i = 1:length(P)
    fv0(i) = fn(i)*Nd0*Vmax*P(i)^3;
end

alfa0 = 0;
for i = 1:Ne
    for j = 1:Nibc(i)
        Nx = indexmapper(i,j);
        if Nx ==1;
        else
        alfa0 = alfa0 + (3/P(Nx))*fv0(Nx)*wts(Nx)*ele_vec(i);
        end
    end
end

alfa0 = (1/phi)*alfa0;

C0 = (1/(2*K))*(-1-((alfa0*K*Gmax)/(1+phi))+K*Cinf+sqrt(1+2*K*Cinf+K^2 ...
        *Cinf^2+((alfa0^2*K^2*Gmax^2)/(1+phi)^2)+((2*alfa0*K*Gmax)...
        /(1+phi))-((2*alfa0*K^2*Gmax*Cinf)/(1+phi))));
G0 = ((K*Gmax*C0)/(1+K*C0));


end
```

indexmapper

```
% Function that takes in element number i and point number in that element
% j and returns the overal indices of the radius domain.
function [x] = indexmapper(i,j)
global Nibc
    if i == 1
        x = j;
    else
        x = sum(Nibc(1:i-1))+j;
    end
end
```

differentialeq

```
% Function that sets up the system of differential equations to be solved
function dfn = differentialeq(z,fn,P,wts,ele_vec)
global Cinf K G F
lenP = length(P);
[BrC,DrC,BrB,DrB,alfa,dalphadz] = interp(fn,P,wts,ele_vec);
z
% Storing birth and death rate integrals
 dfn = zeros(lenP+2,1);
for i = 1:length(P)
    dfn(i) = BrC(i) -DrC(i) + BrB(i) - DrB(i);
end
% Calculating mass balance differentials
n1 = lenP +1;
n2 = lenP+2;
dfn(n1) = -alfa*G*(fn(n1)-((fn(n2))/(Cinf*K*(1-fn(n2)))));

dfn(n2) = F*(fn(n1)-(fn(n2)/(Cinf*K*(1-fn(n2)))))-(fn(n2)/alfa)...
    *dalphadz;

end
```

interp

```
% Function that calculates the birth and death rates of coalescence and
% breakage, the surface to volume ratio and the derivative of the surface
```

```
% to volume ratio.

function [BrC,DrC,BrB,DrB,alfa,dalphadz] = interp(fn,P,wts,ele_vec)
global Ne phi U L Di rho_d rho_c mu_c...
    gamma k_o k_p k_g1 k_g2 ems Rmax Nibc Nd0 R Gmax T K Cinf Vmax
% Defining gatherings of constants
ems = 0.01766*((U^3)/Di)*((mu_c)/(rho_c*U*Di))^(3/8);
fn1 = fn(1:length(P));
A = (k_o*4*nthroot(2,3)*(ems^(1/3))*(Rmax^(7/3))*Nd0*L)/(U*(1+phi));
B = (16*mu_c*rho_c*ems*(Rmax^4))/(k_p*(gamma^2)*((1+phi)^3));
C = (k_g1*(ems^(1/3))*L)/(nthroot(4,3)*(1+phi)*(Rmax^(2/3))*U);
D = (k_g2*gamma*((1+phi)^2))/(nthroot(32,3)*rho_d*(ems^(2/3))*...
    (Rmax^(5/3)));
Clim = fn(length(P)+2)/(K*(Gmax-fn(length(P)+2)));
gamma1 = gamma - R*T*Gmax*log(1+K*Cinf*Clim);
%Making empty matrices to fill in function values from the distribution
%at the values of the parrent droplets. This is to be used in the
%coalescence birth rate integral.
f_1bc = zeros(length(P));
f_2bc = zeros(length(P));
r_1bc = zeros(length(P));
r_2bc = zeros(length(P));

% Creating and storring the new function values
for i = 2:length(P)
    r_1bc(i,:) = (P(i)/(nthroot(2,3)))*P;
    for j = 1:length(P)
        r_2bc(i,j) = (P(i)/(nthroot(2,3)))*(2-P(j)^3)^(1/3);
    end
    f_1bc(i,:) = interp1(P,fn1,r_1bc(i,:),'pchip');
    f_2bc(i,:) = interp1(P,fn1,r_2bc(i,:),'pchip');
end

% Finding the values of the distribution used in the solving of the
% birth rate due to breakage integral. First creating emprty matrix
% to store values.

f_1bb = zeros(length(P));
r_1bb = zeros(length(P));

for i = 1:length(P)
```

```
    r_1bb(i,:) = P(i) + (1-P(i))*P;
    f_1bb(i,:) = interp1(P,fn1,r_1bb(i,:),'pchip');
end

% Creating function that calculates the coalescence rate
    function [cf] = coalrate(r1,r2)
        cf = A*(r1+r2)^2*sqrt(r1^(2/3)+r2^(2/3))*exp(-B*(gamma/gamma1)...
            ^2*((r1*r2)/(r1+r2))^4);
    end
% creating function that calculates rate of breakage
    function rb = breakrate(r1)
        rb = (C/(r1^(2/3)))*exp((-D/(r1^(5/3)))*(gamma1/gamma));
    end
% Creating function that calculates daughtersize distribution
    function dsd = daughter(r1,r2)
        dsd = (45/nthroot(2,3))*((r1^2)/(r2^3))*(((r1^3)/(r2^3))^2)...
        *(1-((r1^3)/(r2^3))^2);
    end

% Calculating birth and death rates of coalescence and breakage
lenP = length(P);
BrC = zeros(1,lenP);
DrC = zeros(1,lenP);
BrB = zeros(1,lenP);
DrB = zeros(1,lenP);

for k = 1:lenP
    for i = 1:Ne
        for j = 1:Nibc(i)
        Nx = indexmapper(i,j);
        x1 = r_1bc(k,Nx);
        x2 = r_2bc(k,Nx);
        x3 = P(k);
        x4 = P(Nx);
        x5 = P(k);
        x6 = r_1bb(k,Nx);
        rc1 = coalrate(x1,x2);
        rc2 = coalrate(x3,x4);
        rb1 = breakrate(x6);
        drd1 = daughter(x5,x6);
        BrC(k) = BrC(k) + rc1*f_1bc(k,Nx)*f_2bc(k,Nx)*wts(Nx)*ele_vec(i)...
```

```
85          *P(k)/nthroot(2,3);
86          DrC(k) = DrC(k) + rc2*fn1(k)*fn1(Nx)*wts(Nx)*ele_vec(i);
87          BrB(k) = BrB(k) + 2*rb1*drd1*f_1bb(k,Nx)*wts(Nx)*ele_vec(i)...
88          *(1-P(k));
89          DrB(k) = breakrate(P(k))*fn1(k);
90          end
91      end
92  end
93  % Seting first element of each rate  vector to 0 since program returns NaN
94  BrC(1) = 0;
95  DrC(1) = 0;
96  BrB(1) = 0;
97  DrB(1) = 0;
98
99  % Adding the mass balance equation to the mix.
100 % First calculating volumetric distribution and then the surface to
101 % volume ratio of each drop and its derivative
102 %fv = zeros(1,lenP);
103 fvdz = zeros(1,lenP);
104 for i = 1:lenP
105     fv(i) = fn1(i)*Nd0*Vmax*P(i)^3;
106     fvdz(i) = (BrC(i)-DrC(i)+BrB(i)-DrB(i))*Nd0*Vmax*P(i)^3;
107 end
108
109 % Surface to volume ratio and its derivative
110 alfa = 0;
111 dalphadz = 0;
112 for i = 1:Ne
113     for j = 1:Nibc(i)
114         Nx = indexmapper(i,j);
115         if Nx ==1;
116         else
117         alfa = alfa + (3/P(Nx))*fv(Nx)*wts(Nx)*ele_vec(i);
118         dalphadz = dalphadz + (3/P(Nx))*fvdz(Nx)*wts(Nx)*ele_vec(i);
119         end
120     end
121 end
122
123 alfa = (1/phi)*alfa;
124 dalphadz = (1/phi)*dalphadz;
125
```

```
end
```

Plotting Functions

```
function PlotInitialConditions( param, exper, sim, files )
%PLOTINITIALCONDITIONS
%   Load figure properties
    properties = PlotProperties ( );

%   Create the figure object and set properties
    hFigure = figure();
    set(hFigure, properties.figures);

%   Create the axes object and set properties
    hAxes = axes();
    set(hAxes, properties.axes);
    xlim([0.1 param.Rmax*1e6])
    hold(hAxes, 'all');

%   Plot the initial experimental number distribution and set the
%   plot properties
    nColor = 4;
    nMarker = 3;
    sMarker = 4;
    plot(exper.rRaw*1e6, exper.fn, 'Color', properties.plots.Colors{nColor}, ...
        'LineWidth', properties.plots.LineWidth, ...
        'Marker', properties.plots.Markers{nMarker}, ...
        'MarkerEdgeColor', properties.plots.Colors{nColor}, ...
        'MarkerFaceColor', properties.plots.Colors{nColor}, ...
        'MarkerSize', sMarker, ...
        'DisplayName', '{\itf}_{n,exp}');

%   Plot the initial experimental volume distribution and set the
%   plot properties
    nColor = 9;
    nMarker = 3;
    sMarker = 4;
    plot(exper.rRaw*1e6, exper.fv, 'Color', properties.plots.Colors{nColor}, ...
        'LineWidth', properties.plots.LineWidth, ...
        'Marker', properties.plots.Markers{nMarker}, ...
```

```
37          'MarkerEdgeColor', properties.plots.Colors{nColor}, ...
38          'MarkerFaceColor', properties.plots.Colors{nColor}, ...
39          'MarkerSize', sMarker, ...
40          'DisplayName', '{\itf}_{v,exp}');
41
42  %   Plot the initial simulation number distribution and set the
43  %   plot properties
44      nColor = 4;
45      nMarker = 1;
46      sMarker = 8;
47      plot(sim.r, sim.fnInitial, 'Color', properties.plots.Colors{nColor}, ...
48          'LineStyle', 'none', ...
49          'LineWidth', properties.plots.LineWidth, ...
50          'Marker', properties.plots.Markers{nMarker}, ...
51          'MarkerEdgeColor', properties.plots.Colors{nColor}, ...
52          'MarkerFaceColor', 'none', ...
53          'MarkerSize', sMarker, ...
54          'DisplayName', '{\itf}_{n0}');
55
56  %   Plot the initial simulation volume distribution and set the
57  %   plot properties
58      nColor = 9;
59      nMarker = 1;
60      sMarker = 8;
61      plot(sim.r, sim.fvInitial, 'Color', properties.plots.Colors{nColor}, ...
62          'LineStyle', 'none', ...
63          'LineWidth', properties.plots.LineWidth, ...
64          'Marker', properties.plots.Markers{nMarker}, ...
65          'MarkerEdgeColor', properties.plots.Colors{nColor}, ...
66          'MarkerFaceColor', 'none', ...
67          'MarkerSize', sMarker, ...
68          'DisplayName', '{\itf}_{v0}');
69
70  %   Define figure title and axis labels and set properties
71      hTitle = title('Comparison of the experimental and interpolated initial..
72      number and volume distributions, {\itf}_{n0} and {\itf}_{v0}');
73      set(hTitle, properties.titles)
74      hXlabel = xlabel('Drop radius, r (\mum)');
75      set(hXlabel, properties.labels)
76      hYlabel = ylabel('{\itf}_{n0} and {\itf}_{v0}');
77      set(hYlabel, properties.labels)
```

```
78
79  %   Create the legend object and set properties
80      hLegend = legend(hAxes, 'show');
81      %set(hLegend, properties.legends);
82
83  %   Save the figure to the specified file in the case output folder
84      fileName = [files.OutputDir '/F01IniDistComp.tiff'];
85      figImage = getframe(gcf);
86      imwrite (figImage.cdata, fileName, 'tiff');
87
88  end
```

```
1   function PlotLengthVariables( sim, files, annote )
2   %PLOTLENGTHVARIABLES
3
4   %   Load figure properties
5       properties = PlotProperties ( );
6
7   %   Create the figure object and set properties
8       hFigure = figure();
9       set(hFigure, properties.figures);
10
11  %   Add annotations to figure
12      hTitle = annotation(hFigure, 'textbox', [0.135 0.755 0.1 0.16], ...
13          'String', (annote));
14      set(hTitle, properties.text);
15
16  %   Create the axes object and set properties
17      hAxes = subplot(2, 1, 1);
18      set(hAxes, properties.axes);
19      set(hAxes, 'Position', [0.1300    0.5838    0.7750    0.3412])
20      hold(hAxes, 'all');
21
22  %   Plot the average number radius and set the plot properties
23      plot(sim.z, sim.MunZ, 'Color', properties.plots.Colors{4}, ...
24          'LineWidth', properties.plots.LineWidth, ...
25          'Marker', properties.plots.Markers{1}, ...
26          'MarkerEdgeColor', properties.plots.Colors{4}, ...
27          'MarkerFaceColor', 'none', ...
28          'DisplayName', '\mu_{n}(z)');
```

```
29
30  %   Plot the average volume radius and set the plot properties
31      plot(sim.z, sim.MuvZ, 'Color', properties.plots.Colors{9}, ...
32          'LineWidth', properties.plots.LineWidth, ...
33          'Marker', properties.plots.Markers{1}, ...
34          'MarkerEdgeColor', properties.plots.Colors{9}, ...
35          'MarkerFaceColor', 'none', ...
36          'DisplayName', '\mu_{v}(z)');
37
38  %   Define figure title and axis labels and set properties
39      hTitle = title('Average radii of the number and volume distributions');
40      set(hTitle, properties.titles)
41      hXlabel = xlabel('Axial position, z (m)');
42      set(hXlabel, properties.labels)
43      hYlabel = ylabel('\mu_{n}(z) and \mu_{v}(z) (m)');
44      set(hYlabel, properties.labels)
45
46  %   Create the legend object and set properties
47      hLegend = legend(hAxes, 'show');
48      %set(hLegend, properties.legends);
49
50  %   Create the axes object and set properties
51      hAxes = subplot(2, 1, 2);
52      set(hAxes, properties.axes);
53      set(hAxes, 'Position', [0.1300    0.1100    0.7750    0.3412])
54      hold(hAxes, 'all');
55
56  %   Plot the volume fraction and set the plot properties
57      plot(sim.z, sim.PhiZ, 'Color', properties.plots.Colors{1}, ...
58          'LineWidth', properties.plots.LineWidth, ...
59          'Marker', properties.plots.Markers{1}, ...
60          'MarkerEdgeColor', properties.plots.Colors{1}, ...
61          'MarkerFaceColor', 'none', ...
62          'DisplayName', '\phi(z)');
63
64  %   Plot the coalescence mass balance and set the plot properties
65      plot(sim.z, sim.VbcZ, 'Color', properties.plots.Colors{5}, ...
66          'LineWidth', properties.plots.LineWidth, ...
67          'Marker', properties.plots.Markers{1}, ...
68          'MarkerEdgeColor', properties.plots.Colors{5}, ...
69          'MarkerFaceColor', 'none', ...
```

```
        'DisplayName', 'M_{C}(z)');

%   Plot the coalescence mass balance and set the plot properties
    plot(sim.z, sim.VbbZ, 'Color', properties.plots.Colors{6}, ...
        'LineWidth', properties.plots.LineWidth, ...
        'Marker', properties.plots.Markers{1}, ...
        'MarkerEdgeColor', properties.plots.Colors{6}, ...
        'MarkerFaceColor', 'none', ...
        'DisplayName', 'M_{B}(z)');

%   Define figure title and axis labels and set properties
    hTitle = title('Volume fraction and coalescence and birth mass balance...
    ratios');
    set(hTitle, properties.titles)
    hXlabel = xlabel('Axial position, z (m)');
    set(hXlabel, properties.labels)
    hYlabel = ylabel('\phi(z) , M_{C}(z) , M_{B}(z)');
    set(hYlabel, properties.labels)

%   Create the legend object and set properties
    hLegend = legend(hAxes, 'show');
    %set(hLegend, properties.legends);

%   Save the figure to the specified file in the case output folder
    fileName = [files.OutputDir '/F02MeanRadiiMassBal.tiff'];
    figImage = getframe(gcf);
    imwrite (figImage.cdata, fileName, 'tiff');
end
```

```
function properties = PlotProperties ( )
%PLOTPROPERTIES
%   Set the font
    fonts.Name = 'Calibri';
    fonts.Size = 18;
    fonts.FontWeight = 'bold';

%   Define a property structure for the figure objects
    properties.figures.Color = [1 1 1];
    properties.figures.OuterPosition = [170, 170, 1280, 960];

```

```
12  %   Define a property structure for the axes
13      properties.axes.FontName = fonts.Name;
14      properties.axes.FontSize = fonts.Size;
15      properties.axes.FontWeight = fonts.FontWeight;
16      properties.axes.Layer = 'top';
17      properties.axes.Color = 'none';
18      properties.axes.LineWidth = 1.5;
19      properties.axes.XScale = 'log';
20      properties.axes.Box = 'on';
21      properties.axes.XGrid = 'on';
22      properties.axes.YGrid = 'on';
23      properties.axes.XMinorTick = 'on';
24      properties.axes.YMinorTick = 'on';
25
26  %   Define a property structure for the legend
27      properties.legends.FontName = fonts.Name;
28      properties.legends.FontSize = fonts.Size-6;
29      properties.legends.FontWeight = 'normal';
30      properties.legends.Location = 'NorthEastOutside';
31  % %     properties.legends.YColor = [1 1 1];
32  % %     properties.legends.XColor = [1 1 1];
33
34  %   Define a color array for the plots
35      properties.plots.LineWidth = 2;
36      properties.plots.Colors{1} = [0 0 0];
37      properties.plots.Colors{2} = [0.600000023841858 0.200000002980232 0];
38      properties.plots.Colors{3} = [0.47843137383461 0.062745101749897...
39      0.894117653369904];
40      properties.plots.Colors{4} = [0 0 1];
41      properties.plots.Colors{5} = [0 0.749019622802734 0.749019622802734];
42      properties.plots.Colors{6} = [0 0.498039215803146 0];
43      properties.plots.Colors{7} = [0.749019622802734 0.749019622802734 0];
44      properties.plots.Colors{8} = [0.87058824300766 0.490196079015732 0];
45      properties.plots.Colors{9} = [1 0 0];
46      properties.plots.Markers{1} = 'o';
47      properties.plots.Markers{2} = 'square';
48      properties.plots.Markers{3} = 'diamond';
49      properties.plots.Markers{4} = '^';
50      properties.plots.Markers{5} = 'pentagram';
51      properties.plots.Markers{6} = 'hexagram';
52      properties.plots.Markers{7} = '+';
```

```
    properties.plots.Markers{8} = '*';
    properties.plots.Markers{9} = '>';

%   Define a property structure for plot titles
    properties.titles.FontName = fonts.Name;
    properties.titles.FontSize = fonts.Size+2;
    properties.titles.FontWeight = fonts.FontWeight;

%   Define a property structure for axis labels
    properties.labels.FontName = fonts.Name;
    properties.labels.FontSize = fonts.Size;
    properties.labels.FontWeight = fonts.FontWeight;

% Define a property structure for text objects
    properties.text.BackgroundColor = [1, 1, 1];
    properties.text.Color = [0, 0, 0];
    properties.text.EdgeColor = [1, 1, 1];
    properties.text.Editing = 'off';
    properties.text.FontName = fonts.Name;
    properties.text.FontSize = fonts.Size-6;
    properties.text.FontWeight = 'light';
    properties.text.HorizontalAlignment = 'left';
    properties.text.VerticalAlignment = 'middle';
    properties.text.FitBoxToText = 'off';
end
```

```
function PlotRadiusVariables ( z, r, f1, f2, zi, titles, xlabels, ylabels...
, legtext, annote, filename)
%PLOTDENSITYDISTRIBUTIONS

%   Load figure properties
    properties = PlotProperties ( );

%   Create the figure object and set properties
    hFigure = figure();
    set(hFigure, properties.figures);

%   Add annotations to subplot 1
    hTitle1 = annotation(hFigure, 'textbox', [0.135 0.755 0.1 0.16], ...
        'String', (annote));
```

```
15      set(hTitle1, properties.text);
16
17  %   Add annotations to subplot 2
18      hTitle2 = annotation(hFigure, 'textbox', [0.135 0.281 0.1 0.16], ...
19          'String', (annote));
20      set(hTitle2, properties.text);
21
22  %   Loop for each subplot
23      axesPosition{1} = [0.1300    0.5838    0.7750    0.3412];
24      axesPosition{2} = [0.1300    0.1100    0.7750    0.3412];
25      for p=1:2
26      %   Create the axes object and set properties
27          hAxes = subplot(2, 1, p);
28          set(hAxes, properties.axes);
29          set(hAxes, 'Position', axesPosition{p})
30          hold(hAxes, 'all');
31
32      %   Plot the input functions at the selected z points and set the
33      %   plot properties
34          for i=1:9
35          %   Select function set to plot
36              if (p == 1)
37                  f = f1(zi(i),:);
38              elseif (p == 2)
39                  f = f2(zi(i),:);
40              end
41          %   Plot the set of functions at each z point
42              plot(r, f, 'Color', properties.plots.Colors{i}, ...
43                  'LineWidth', properties.plots.LineWidth, ...
44                  'Marker', properties.plots.Markers{i}, ...
45                  'MarkerEdgeColor', properties.plots.Colors{i}, ...
46                  'MarkerFaceColor', 'none', ...
47                  'DisplayName', [legtext{p} num2str(z(zi(i)),'%8.3f') ')']);
48          end
49
50      %   Define figure title and axis labels and set properties
51          hTitle = title(titles{p});
52          set(hTitle, properties.titles)
53          hXlabel = xlabel(xlabels{p});
54          set(hXlabel, properties.labels)
55          hYlabel = ylabel(ylabels{p});
```

```
        set(hYlabel, properties.labels)

    %   Create the legend object and set properties
        hLegend = legend(hAxes, 'show');
        %set(hLegend, properties.legends);
    end

%   Save the figure to the specified file in the case output folder
    figImage = getframe(gcf);
    imwrite (figImage.cdata, filename, 'tiff');
end
```

```
function PlotTransientData( z, r, fn, fv, Cbr, Cdr, Bbr, Bdr, dfdt )
%PLOTTRANSIENTDATA
%   Close all open figures
    close all;

%   Load figure properties
    properties = PlotProperties ( );

%   Plot the current number and volume distribution
    PlotCurrentDistributions (properties, z, r, fn, fv)

%   Plot the current coalescence birth and death functions
    PlotCurrentCoalescenceRates (properties, z, r, Cbr, Cdr, dfdt)

%   Plot the current coalescence birth and death functions
    PlotCurrentBreakageRates (properties, z, r, Bbr, Bdr, dfdt)

%   Pause the program for 10s to allow for the plots to render.
    pause(10)
end

%--------------------------------------------------------------------------
%   Density distribution plotting function
function PlotCurrentDistributions (properties, z, r, fn, fv)

%   Create the figure object and set properties
    hFigure = figure();
    set(hFigure, properties.figures)
```

```
29
30  %   Create the axes object and set properties
31      hAxes = axes('Parent', hFigure);
32      set(hAxes, properties.axes)
33      hold(hAxes, 'all');
34
35  %   Plot the number distribution and set its properties
36      plot(r, fn, 'Color', properties.plots.Colors{2}, ...
37          'LineWidth', properties.plots.LineWidth, ...
38          'Marker', properties.plots.Markers{3}, ...
39          'MarkerEdgeColor', properties.plots.Colors{2}, ...
40          'MarkerFaceColor', 'none', ...
41          'DisplayName', '{\itf}_n(r,z)');
42
43  %   Plot the number distribution and set its properties
44      plot(r, fv, 'Color', properties.plots.Colors{4}, ...
45          'LineWidth', properties.plots.LineWidth, ...
46          'Marker', properties.plots.Markers{1}, ...
47          'MarkerEdgeColor', properties.plots.Colors{4}, ...
48          'MarkerFaceColor', 'none', ...
49          'DisplayName', '{\itf}_v(r,z)');
50
51  %   Define figure title and axis labels and set properties
52      hTitle = title(['Normalized volume and number distribution at z...
53      =' num2str(z,'%8.6f')]);
54      set(hTitle, properties.titles)
55      hXlabel = xlabel('R_d (\mum)');
56      set(hXlabel, properties.labels)
57      hYlabel = ylabel('{\itf}_n and {\itf}_v');
58      set(hYlabel, properties.labels)
59
60  %   Create the legend object and set properties
61      hLegend = legend(hAxes, 'show');
62      %set(hLegend, properties.legends);
63  end
64
65  %--------------------------------------------------------------------------
66  %   Coalescence birth and death rate plotting function
67  function PlotCurrentCoalescenceRates (properties, z, r, Cbr, Cdr, dfdt)
68
69  %   Create the figure object and set properties
```

```
70      hFigure = figure();
71      set(hFigure, properties.figures)
72
73  %   Create the axes object and set properties
74      hAxes = subplot(2, 1, 1, 'Parent', hFigure);
75      set(hAxes, properties.axes)
76      hold(hAxes, 'all');
77
78  %   Plot the number distribution and set its properties
79      plot(r, Cbr, 'Color', properties.plots.Colors{2}, ...
80          'LineWidth', properties.plots.LineWidth, ...
81          'Marker', properties.plots.Markers{3}, ...
82          'MarkerEdgeColor', properties.plots.Colors{2}, ...
83          'MarkerFaceColor', 'none', ...
84          'DisplayName', 'R_{Cb}(r,z)');
85
86  %   Plot the number distribution and set its properties
87      plot(r, Cdr, 'Color', properties.plots.Colors{4}, ...
88          'LineWidth', properties.plots.LineWidth, ...
89          'Marker', properties.plots.Markers{1}, ...
90          'MarkerEdgeColor', properties.plots.Colors{4}, ...
91          'MarkerFaceColor', 'none', ...
92          'DisplayName', 'R_{Cd}(r,z)');
93
94  %   Define figure title and axis labels and set properties
95      hTitle = title(['Normalized coalescence birth and death rates at...
96      z =' num2str(z,'%8.6f')]);
97      set(hTitle, properties.titles)
98      hXlabel = xlabel('R_d (\mum)');
99      set(hXlabel, properties.labels)
100     hYlabel = ylabel('R_{Cb}(r,z) and R_{Cd}(r,z)');
101     set(hYlabel, properties.labels)
102
103 %   Create the legend object and set properties
104     hLegend = legend(hAxes, 'show');
105     %set(hLegend, properties.legends);
106
107 %   Create the axes object for the total rate and differential and
108 %   set properties
109     hAxes = subplot(2, 1, 2, 'Parent', hFigure);
110     set(hAxes, properties.axes)
```

```
111     hold(hAxes, 'all');
112 
113 %   Plot the number distribution and set its properties
114     Ctr = Cbr-Cdr;
115     plot(r, Ctr, 'Color', properties.plots.Colors{2}, ...
116         'LineWidth', properties.plots.LineWidth, ...
117         'Marker', properties.plots.Markers{3}, ...
118         'MarkerEdgeColor', properties.plots.Colors{2}, ...
119         'MarkerFaceColor', 'none', ...
120         'DisplayName', 'R_{ct}(r,z)');
121 
122 %   Plot the number distribution and set its properties
123     plot(r, dfdt, 'Color', properties.plots.Colors{4}, ...
124         'LineWidth', properties.plots.LineWidth, ...
125         'Marker', properties.plots.Markers{1}, ...
126         'MarkerEdgeColor', properties.plots.Colors{4}, ...
127         'MarkerFaceColor', 'none', ...
128         'DisplayName', 'df_n/dz(r,z)');
129 
130 %   Define figure title and axis labels and set properties
131     hTitle = title(['Comparison between the coalescence rate and the...
132     total system differential at z =' num2str(z,'%8.6f')]);
133     set(hTitle, properties.titles)
134     hXlabel = xlabel('R_d (\mum)');
135     set(hXlabel, properties.labels)
136     hYlabel = ylabel('R_{Ct}(r,z) and df_n/dz(r,z)');
137     set(hYlabel, properties.labels)
138 
139 %   Create the legend object and set properties
140     hLegend = legend(hAxes, 'show');
141     %set(hLegend, properties.legends);
142 end
143 
144 %--------------------------------------------------------------------------
145 %   Coalescence birth and death rate plotting function
146 function PlotCurrentBreakageRates (properties, z, r, Bbr, Bdr, dfdt)
147 
148 %   Create the figure object and set properties
149     hFigure = figure();
150     set(hFigure, properties.figures)
151 
```

```
152 %   Create the axes object and set properties
153     hAxes = subplot(2, 1, 1, 'Parent', hFigure);
154     set(hAxes, properties.axes)
155     hold(hAxes, 'all');
156
157 %   Plot the number distribution and set its properties
158     plot(r, Bbr, 'Color', properties.plots.Colors{2}, ...
159         'LineWidth', properties.plots.LineWidth, ...
160         'Marker', properties.plots.Markers{3}, ...
161         'MarkerEdgeColor', properties.plots.Colors{2}, ...
162         'MarkerFaceColor', 'none', ...
163         'DisplayName', 'R_{Bb}(r,z)');
164
165 %   Plot the number distribution and set its properties
166     plot(r, Bdr, 'Color', properties.plots.Colors{4}, ...
167         'LineWidth', properties.plots.LineWidth, ...
168         'Marker', properties.plots.Markers{1}, ...
169         'MarkerEdgeColor', properties.plots.Colors{4}, ...
170         'MarkerFaceColor', 'none', ...
171         'DisplayName', 'R_{Bd}(r,z)');
172
173 %   Define figure title and axis labels and set properties
174     hTitle = title(['Normalized breakage birth and death rates at...
175     z =' num2str(z,'%8.6f')]);
176     set(hTitle, properties.titles)
177     hXlabel = xlabel('R_d (\mum)');
178     set(hXlabel, properties.labels)
179     hYlabel = ylabel('R_{Bb}(r,z) and R_{Bd}(r,z)');
180     set(hYlabel, properties.labels)
181
182 %   Create the legend object and set properties
183     hLegend = legend(hAxes, 'show');
184     %set(hLegend, properties.legends);
185
186 %   Create the axes object for the total rate and differential and
187 %   set properties
188     hAxes = subplot(2, 1, 2, 'Parent', hFigure);
189     set(hAxes, properties.axes)
190     hold(hAxes, 'all');
191
192 %   Plot the number distribution and set its properties
```

```
    Btr = Bbr-Bdr;
    plot(r, Btr, 'Color', properties.plots.Colors{2}, ...
        'LineWidth', properties.plots.LineWidth, ...
        'Marker', properties.plots.Markers{3}, ...
        'MarkerEdgeColor', properties.plots.Colors{2}, ...
        'MarkerFaceColor', 'none', ...
        'DisplayName', 'R_{ct}(r,z)');

%   Plot the number distribution and set its properties
    plot(r, dfdt, 'Color', properties.plots.Colors{4}, ...
        'LineWidth', properties.plots.LineWidth, ...
        'Marker', properties.plots.Markers{1}, ...
        'MarkerEdgeColor', properties.plots.Colors{4}, ...
        'MarkerFaceColor', 'none', ...
        'DisplayName', 'df_n/dz(r,z)');

%   Define figure title and axis labels and set properties
    hTitle = title(['Comparison between the breakage rate and the...
    total system differential at z =' num2str(z,'%8.6f')]);
    set(hTitle, properties.titles)
    hXlabel = xlabel('R_d (\mum)');
    set(hXlabel, properties.labels)
    hYlabel = ylabel('R_{Bt}(r,z) and df_n/dz(r,z)');
    set(hYlabel, properties.labels)

%   Create the legend object and set properties
    hLegend = legend(hAxes, 'show');
    %set(hLegend, properties.legends);
end
```

SurfPlots

```
% Script that provides surface plots of length, average volume and
% average radius against different parameters of the model
% at the point where the distribution is equilibriated.
close all
clc
% Function that plots the outputs taken from runs of the code with
% different parameter sets.
komegavec = [1e-3; 0.33e-2; 0.66e-2; 1e-2; 0.33e-1; 0.66e-1;...
    1e-1; 0.33; 0.66; 1];
```

```
kg1vec =    [1e-3; 0.33e-2; 0.66e-2; 1e-2; 0.33e-1; 0.66e-1;...
    1e-1; 0.33; 0.66; 1];
kpsivec = [1e-22; 0.33e-21; 0.66e-21; 1e-21; 0.33e-20; 0.66e-20;...
   1e-20; 0.33e-19; 0.66e-19; 1e-19];
kg2vec =  [1e-5; 0.33e-4; 0.66e-4; 1e-4; 0.33e-3; 0.66e-3;...
    1e-3; 0.33e-2; 0.66e-2; 1e-2];
%   Storing the length to equlibrium in aray.
Linfmat = zeros(10);
Muninfmat = zeros(10);
Muvinfmat = zeros(10);

for i = 1:10
    for j = 1:10
    n = (i-1)*10+j;
    m = 100 +(i-1)*10+j;
    Linfmat(j,i) = SimOutput{n}.Liv;
    Muninfmat(j,i) = SimOutput{m}.MunZ(SimOutput{m}.nZinf);
    Muvinfmat(j,i) = SimOutput{m}.MuvZ(SimOutput{m}.vZinf);
    end
end

% Making surface plot of the length to equilibrium vs diferent pairings
% of ko and kg1 paramters.

figure
surf(kg1vec,komegavec,Linfmat)
xlabel('\fontsize{15}\bf k_{g1}')
ylabel('\fontsize{15}\bf k_{\omega}')
zlabel('\fontsize{15}\bf L_{\infty}')
title('Linf for different paramter sets')
set(gca,'xscale','log','yscale','log')

figure
surf(kg2vec,kpsivec,Muninfmat)
xlabel('\fontsize{15}\bf k_{g2}')
ylabel('\fontsize{15}\bf k_{\psi}')
zlabel('\fontsize{15}\bf \mu_{n,\infty}')
set(gca,'xscale','log','yscale','log')

figure
surf(kg2vec,kpsivec,Muvinfmat)
```

```
xlabel('\fontsize{15}\bf k_{g2}')
ylabel('\fontsize{15}\bf k_{\psi}')
zlabel('\fontsize{15}\bf \mu_{v,\infty}')
set(gca,'xscale','log','yscale','log')
```

PlotSurfactantVariables

```
function PlotSurfactantVariables( sim, files, annote )
%PLOTLENGTHVARIABLES Summary of this function goes here
%   Detailed explanation goes here

%   Load figure properties
    properties = PlotProperties ( );

%   Create the figure object and set properties
    hFigure = figure();
    set(hFigure, properties.figures);

%   Add annotations to figure
    hTitle = annotation(hFigure, 'textbox', [0.132 0.715 0.1 0.196], ...
        'String', (annote));
    set(hTitle, properties.text);

%   Create the axes object and set properties
    hAxes = subplot(2, 1, 1);
    set(hAxes, properties.axes);
    set(hAxes, 'Position', [0.1300    0.5838    0.7750    0.3412])
    ylim([0 1]);
    hold(hAxes, 'all');

%   Plot the bulk surfactant concentration and set the plot properties
    plot(sim.z, sim.CbsZ, 'Color', properties.plots.Colors{4}, ...
        'LineWidth', properties.plots.LineWidth, ...
        'Marker', properties.plots.Markers{1}, ...
        'MarkerEdgeColor', properties.plots.Colors{4}, ...
        'MarkerFaceColor', 'none', ...
        'DisplayName', 'C(z)/C_\infty');

%   Plot the interfacial surfactant concentration and set the plot
%   properties
    plot(sim.z, sim.GbsZ, 'Color', properties.plots.Colors{9}, ...
```

```
35          'LineWidth', properties.plots.LineWidth, ...
36          'Marker', properties.plots.Markers{1}, ...
37          'MarkerEdgeColor', properties.plots.Colors{9}, ...
38          'MarkerFaceColor', 'none', ...
39          'DisplayName', '\Gamma(z)/\Gamma_{max}');
40
41  %   Plot the average volume radius and set the plot properties
42      plot(sim.z, sim.IftZ, 'Color', properties.plots.Colors{6}, ...
43          'LineWidth', properties.plots.LineWidth, ...
44          'Marker', properties.plots.Markers{1}, ...
45          'MarkerEdgeColor', properties.plots.Colors{6}, ...
46          'MarkerFaceColor', 'none', ...
47          'DisplayName', '\gamma(z)/\gamma_0');
48
49  %   Define figure title and axis labels and set properties
50      hTitle = title('Surfactant concentration in the bulk and interface...
51      / Interfacial tension');
52      set(hTitle, properties.titles)
53      hXlabel = xlabel('Axial position, z (m)');
54      set(hXlabel, properties.labels)
55      hYlabel = ylabel('C(z)/C_\infty ; \Gamma(z)/\Gamma_{max}...
56      ; and \gamma(z)/\gamma_0');
57      set(hYlabel, properties.labels)
58
59  %   Create the legend object and set properties
60      hLegend = legend(hAxes, 'show');
61      set(hLegend, properties.legends);
62
63  %   Create the axes object and set properties
64      hAxes = subplot(2, 1, 2);
65      set(hAxes, properties.axes);
66      set(hAxes, 'Position', [0.1300    0.1100    0.7750    0.3412])
67      hold(hAxes, 'all');
68
69  %   Plot the surface to volume ratio and set the plot properties
70      plot(sim.z, sim.SvrZ, 'Color', properties.plots.Colors{5}, ...
71          'LineWidth', properties.plots.LineWidth, ...
72          'Marker', properties.plots.Markers{1}, ...
73          'MarkerEdgeColor', properties.plots.Colors{5}, ...
74          'MarkerFaceColor', 'none', ...
75          'DisplayName', '\alpha(z)ÂůR_{max}');
```

```
76
77  %   Define figure title and axis labels and set properties
78      hTitle = title('Surface to volume ratio of the emulsions versus...
79      pipe length');
80      set(hTitle, properties.titles)
81      hXlabel = xlabel('Axial position, z (m)');
82      set(hXlabel, properties.labels)
83      hYlabel = ylabel('\alpha(z)ÂůR_{max}');
84      set(hYlabel, properties.labels)
85
86  %   Create the legend object and set properties
87      hLegend = legend(hAxes, 'show');
88      set(hLegend, properties.legends);
89
90  %   Save the figure to the specified file in the case output folder
91      fileName = [files.OutputDir '/F07SurfactantPlot.tiff'];
92      figImage = getframe(gcf);
93      imwrite (figImage.cdata, fileName, 'tiff');
94  end
```

ConPlots

```
1   close all
2
3   G1 = SimOutput{1}.GbsZ;
4   G2 = SimOutput{2}.GbsZ;
5   G3 = SimOutput{3}.GbsZ;
6   C1 = SimOutput{1}.CbsZ;
7   C2 = SimOutput{2}.CbsZ;
8   C3 = SimOutput{3}.CbsZ;
9   I1 = SimOutput{1}.IftZ;
10  I2 = SimOutput{2}.IftZ;
11  I3 = SimOutput{3}.IftZ;
12  S1 = SimOutput{1}.SvrZ;
13  S2 = SimOutput{2}.SvrZ;
14  S3 = SimOutput{3}.SvrZ;
15  Z1 = SimOutput{1}.z;
16  Z2 = SimOutput{2}.z;
17  Z3 = SimOutput{3}.z;
18  Z4 = ans.z;
19  M1 = SimOutput{1}.MuvZ;
```

```
M2 = SimOutput{2}.MuvZ;
M3 = SimOutput{3}.MuvZ;
M4 = ans.MuvZ;


figure
semilogx(Z1,C1,'-o b',Z2,C2,'-o y',Z3,C3,'-o r','LineWidth',2);
axis([1e-4 1e2 0 1])
xlabel('\fontsize{20}\bf z[m]')
ylabel('\fontsize{20}\bf C/C_{\infty}')
set(gca,'fontsize',18)
title('Surfactant concentration in the bulk phase')
legend('\fontsize{17}C_\infty = 0.05','\fontsize{17}C_\infty = 0.1',...
    '\fontsize{17}C_\infty = 0.15')

figure
semilogx(Z1,G1,'-o b',Z2,G2,'-o y',Z3,G3,'-o r','LineWidth',2);
axis([1e-4 1e2 0 1])
xlabel('\fontsize{20}\bf z[m]')
ylabel('\fontsize{20}\bf \Gamma/\Gamma_{max}')
set(gca,'fontsize',18)
title('Surfactant adsorbed on the interface')
legend('\fontsize{17}C_\infty = 0.05','\fontsize{17}C_\infty = 0.1',...
    '\fontsize{17}C_\infty = 0.15')

figure
semilogx(Z1,I1,'-o b',Z2,I2,'-o y',Z3,I3,'-o r','LineWidth',2);
axis([1e-4 1e2 0 1])
xlabel('\fontsize{20}\bf z[m]')
ylabel('\fontsize{20}\bf \gamma/\gamma_0')
set(gca,'fontsize',18)
title('Interfacial tension')
legend('\fontsize{17}C_\infty = 0.05','\fontsize{17}C_\infty = 0.1',...
    '\fontsize{17}C_\infty = 0.15')

figure
semilogx(Z1,S1,'-o b',Z2,S2,'-o y',Z3,S3,'-o r','LineWidth',2);
axis([1e-4 1e2 0 80])
xlabel('\fontsize{20}\bf z[m]')
ylabel('\fontsize{20}\bf \alpha')
set(gca,'fontsize',18)
```

```
61 title('Surface to volume ratio')
62 legend('\fontsize{17}C_\infty = 0.05','\fontsize{17}C_\infty = 0.1',...
63     '\fontsize{17}C_\infty = 0.15')
64
65 figure
66 semilogx(Z1,M1,'-o b',Z2,M2,'-o y',Z3,M3,'-o r',Z4,M4,'-o...
67 g','LineWidth',2);
68 axis([1e-4 1e2 20 50])
69 xlabel('\fontsize{20}\bf z[m]')
70 ylabel('\fontsize{20}\bf \mu_{v}')
71 set(gca,'fontsize',18)
72 title('Average drop size evolution')
73 legend('\fontsize{17}C_\infty = 0.05','\fontsize{17}C_\infty = 0.1',...
74     '\fontsize{17}C_\infty = 0.15','Without mass balance')
```