Cambridge Centre for Computational Chemical Engineering

University of Cambridge

Department of Chemical Engineering

Preprint ISSN 1473 – 4273

Stochastic simulation of coalescence and breakage processes: a practical study

Mike Goodson, Markus Kraft[1]

submitted: 16th January 2003

[1] Department of
Chemical Engineering
University of Cambridge
Pembroke Street
Cambridge CB2 3RA
UK
E-Mail: markus_kraft@cheng.cam.ac.uk

Preprint No. 9

c4e

Key words and phrases. population balance equation, coalescence, breakage, stochastic simulation, direct simulation algorithm, mass flow algorithm.

Edited by
Cambridge Centre for Computational Chemical Engineering
Department of Chemical Engineering
University of Cambridge
Cambridge CB2 3RA
United Kingdom.

Fax: + 44 (0)1223 334796
E-Mail: c4e@cheng.cam.ac.uk
World Wide Web: http://www.cheng.cam.ac.uk/c4e/

Abstract

In this paper, a practical study is made of two stochastic solution methods for the population balance equation, simulating coalescence and binary breakage. The first algorithm studied is the existing Direct Simulation Algorithm (DSA), proposed in (Eibeck and Wagner, *Stoch. Anal. App.*, 18(6):921-948,2000). The second is an extension of the Mass Flow Algorithm (MFA), which was proposed in (Eibeck and Wagner, *Ann. Appl. Probab.*, 11(4):1137-1165,2001) for coagulation only. MFA is extended to include breakage and a binary search method of distribution generation is introduced, leading to improved efficiency. Numerical investigation of the performance of the two algorithms is carried out by applying them both to a test case, for which an analytical solution is calculated. For both algorithms, convergence of the predicted moments to the analytical solution goes as the inverse of the number of stochastic particles, N, except for the zeroth moment predicted by MFA. This exhibits large fluctuations, due to the presence of very small particles, and converges approximately as $N^{-1/3}$. The new algorithm, MFA, exhibits significant variance reduction—and therefore improved simulation efficiency—for the prediction of higher moments, but for our test case the zeroth moment (the total number of particles) is predicted with better efficiency by DSA. In many breakage models for liquid-liquid systems however, the introduction of a minimum particle size reduces the advantage held by DSA for predicting the zeroth moment. Depending on the minimum particle size, MFA can perform comparably with DSA for predicting the zeroth moment.

Contents

1 Modelling Coalescence and Breakage

Mathematical modelling of liquid-liquid extraction can be categorised into three basic types. **Empirical fitting** is used, based on experimental results, to predict mean drop size or hold-up based on liquid properties, column geometry and operating conditions (e.g. [20] for a rotating disc contactor). These models are generally simple and efficient, but cannot be extrapolated beyond the stipulated range of application. In **Stagewise modelling** (e.g. [1]), an extraction column is described as a series of perfectly mixed stages; effectively, each stage is treated as a CSTR. These stages may be real stages, e.g. as in a sieve-plate column, or cascades to approximate a differentially varying system (e.g. a packed extraction column). **Differential models** involve formulation of differential conservation equations for the two liquid phases. Within this category, there are two further distinct mathematical treatments of the liquid phases. The dispersed phase can be treated as **pseudo-homogeneous** [37], thus in effect there are two continuous phases in the model. When coalescence and breakage[1] cannot be neglected, and the dispersed phase continually undergoes changes, the pseudo-homogeneous treatment breaks down, and a **population balance model** [34] must be applied. Here a differential balance is formulated for the number of drops in the dispersed phase, taking into account birth and death rates due to such phenomena as coalescence, breakage and convective transport.

Further description of the models mentioned above, as well as information on many types of liquid-liquid extraction equipment, can be found in the extensive review by Mohanty [28].

While each of the above methods of modelling liquid-liquid extraction has its advantages and disadvantages, we concentrate on the population balance approach. Population balance modelling has a long history, being used as far back as 1916 to describe the pure coagulation case [43]. A statistical mechanical description was constructed by Hulbert and Katz [18], while the application to particulate processes was specified by Randolph and Larson [35]. The first application of the population balance to liquid-liquid extraction was in 1966, by Valentas *et al.* [41] for breakage and by Valentas and Amundson [40] for both breakage and coalescence. A very thorough study of the field of population balances and their applications to particulate systems has been made by Ramkrishna [34].

The **population balance equation** (pbe) gives, when solved, a complete statistical description of the system, i.e. a number density function for the dispersed phase bubbles in terms of size, concentration or other internal co-ordinates. The pbe is frequently an integro-differential equation that cannot be solved analytically, and hence a numerical approach must be taken. Analytical approximations can sometimes be derived for steady state solutions at the ends of the distribution [4] but a

[1] In the literature, many terms are used to describe the processes of coalescence (coagulation, aggregation, agglomeration) and breakage (breakup, fragmentation). In this paper we, in line with the liquid-liquid literature, use coalescence and breakage. Occasionally (such as in the case of Smoluchowski's coagulation equation) it is more pertinent to use another term. These are, however, intended to be synonymous, and no distinction should be made between them.

complete solution can only be found for the simplest (and generally non-physical) cases (e.g. [31]). The method of moments (e.g. [9, 10] for coagulation only) is an efficient method of solution, but gives no information about the shape of the distribution. In more complicated situations, there can also be problems with closure of the set of equations to be solved [5]. Efforts have been made to reconstruct the distribution from the moment solution [5], but there are difficulties, given that there is no obvious orthogonal set of functions from which a distribution can be constructed. Alternatively, an assumed shape can be imposed on the distribution to fit with a limited number of the moments. This can include monodisperse, log-normal [14] or Rosin-Rammler [2].

Sectional methods are a well established solution technique in population balance modelling. Discretisation of the size distribution can be linear [16] or geometric [11, 17, 25]; a coarser grid is possible with the latter method. To recover functionals of the size distribution, a pivotal particle size must be chosen for each interval.

Recent work includes efforts to use discretisation as a tool for recovering specific functionals of the distribution, rather than merely to approximate the system [21] and to use a varying pivotal particle size for each discrete interval in order to better account for non-uniformities in number density [22]. Several sectional approaches have been examined and reviewed by Vanni [42].

An alternative strategy is to employ finite-element methods, in which the solution is approximated as linear combinations of basis functions over a finite number of subdomains. Finite element methods include the method of weighted residuals, the method of orthogonal collocation and Galerkin's method. Nicmanis and Hounslow [30] applied a finite-element method to various cases of the steady state population balance equation, finding more accurate solution than using the discretisation method and using less computational power.

Liu and Cameron [27], in a similar vein, used a wavelet-based method. They found particular success in cases where steep fronts in the distribution are present, whereas discretisation in other solution methods would have to address this problem directly.

A more detailed review of numerical techniques for solution of the population balance equation was performed by Ramkrishna [33].

Stochastic (or Monte Carlo) methods are an attractive way of tackling the population balance equation due to the discrete nature of the mechanisms being modelled: breakage and coalescence [32]. They can also provide a feasible solution method for multi-dimensional (dependent on more than just size) population balance problems, where standard numerical techniques can become prohibitively computationally expensive. It is beyond the scope of this investigation to compare stochastic solution methods with every available numerical technique; instead we restrict ourselves to a thorough numerical investigation of two stochastic algorithms with the aim of making the step from existing theoretical formulations towards a practicable method of simulating coalescence-breakage systems.

Broadly speaking, stochastic simulation involves generating fictitious realisations of the behaviour of a set of particles. This particle ensemble is an approximation of the real life system being examined. As the process relies on random number generation to choose the nature and timing of the coalescence and breakage events, many trajectories are generated, and an average (which converges to the solution of the population balance equation [6]) is calculated. For the one-dimensional case (size dependency only), direct stochastic simulation (e.g. [12]) can be outperformed by conventional numerical techniques. Hence, there are several ways of improving the efficiency of simulation. To avoid the problems of decreasing particle number (and therefore variance increase) or increasing particle number (and thereby reduced efficiency), a particle combining technique can be employed, in which parallel arrays of particles are periodically combined to give an array with the desired number of particles [36]. A natural extension of this technique is the constant number simulation method of Matsoukas and co-workers [38, 23, 24]. One of the major inefficiencies of simulation comes from the coalescence process. A direct consideration of $n(n-1)$ possible particle pairs (out of n particles) can be avoided by introducing the fictitious jump technique of Eibeck and Wagner [7, 6]. Here a majorant kernel is used that enables independent generation of the two coalescing particles. To ensure that the simulation remains exact, null events (fictitious jumps) are added to the simulation space. Eibeck and Wagner [8] have introduced further variance reduction—and hence improved efficiency—by considering mass (rather than number) concentration. Their mass flow technique enables accurate simulation with far fewer stochastic particles in the simulation array.

The **purpose of this paper** is to examine the feasibility of using a stochastic algorithm, either Direct Simulation Algorithm (DSA) or Mass Flow Algorithm (MFA), to simulate solutions to the population balance equation, where coalescence and binary breakage both occur. DSA is examined as described in [6], and MFA (formulated for coagulation only in [8]) is extended to include binary breakage. The unchanging number of stochastic particles used by MFA lends itself to using a binary search method of probability distribution generation with an associated reduction in simulation CPU time. The two algorithms are applied to a simple test case, for which an analytical solution can be calculated. The two methods are compared for speed of simulation, systematic and statistical errors with the aim of determining which method can predict a solution with a given error in a shorter time. With real liquid-liquid coalescence-breakage models in mind, the effect of imposing a minimum particle size on the system is also investigated.

The paper is organised as follows. In **Section 2**, we extend the well-known Smoluchowski coagulation equation to include breakage. Discrepancies between representations of this process in the literature are discussed. **Section 3** describes the Direct Simulation Algorithm as applied to the coalescence-breakage case; **Section 4** does the same for the Mass Flow Algorithm, extending it to include breakage. The analytical solutions for a suitable test case are derived in **Section 5**. The implications of different methods of data storage are examined in **Section 6**, and numerical comparisons between the two stochastic algorithms are presented in **Section 7**. The effects of imposing a minimum particle size are studied in **Section 8**. Finally, in

Section 9 we present our conclusions.

2 Breakage Process

Mathematical consideration of the coagulation process is well-established [43], and the particle size evolution in the continuous case is given as:

$$\frac{\partial c(x,t)}{\partial t} = \frac{1}{2}\int_0^x K(x-y,y)c(x-y,t)c(y,t)dy - \int_0^\infty K(x,y)c(x,t)c(y,t)dy. \quad (2.1)$$

Here, the number concentration, $c(x,t)$, of particles of size x can increase by coagulation of two particles smaller than x and decrease by coagulation of a particle of size x with any other particle. The rate kernel, $K(x,y)$ gives the rate at which particles of sizes x and y coagulate.

The mathematical treatment of binary breakage is less well-established, with contradictory formulations proposed in the literature. There are two ways to consider the breakage characteristics of a population of particles. One is to describe the breakage by a breakage frequency, $g(x)$, and a probability density function for the sizes of the daughter particles, $\beta(x,y)$. $g(x)$ is the frequency at which a particle of size x undergoes breakage and $\beta(x,y)dy$ gives the probability that the size of a particle formed from breakage of a particle of size x is in the range $[y, y+dy]$. The evolution of the particle size density function (PSDF) is:

$$\frac{\partial c(x,t)}{\partial t} = -c(x,t)g(x) + 2\int_x^\infty c(y,t)g(y)\beta(y,x)dy. \quad (2.2)$$

This representation is commonly used in the literature (e.g. [3]) when it is required to match experimental observations with theoretical predictions of breakage rates. The breakage rate, $g(x)$, is measurable, although it can be harder to do so for the daughter particle size distribution function. An assumed distribution is sometimes used instead of a theoretical model, e.g. uniform [29], Beta [15] or normal [3].

The second way of looking at the breakage rate is to take the frequency of production of the daughter particles, so that the function $f(x,y)$ is the rate at which a particle of size x and a particle of size y are formed from the breakage of a particle of size $x+y$. This representation can sometimes be more convenient when a solution to the population balance equation is being sought. It should be noted here that there is some disagreement in the literature as to the exact form of the pbe when $f(x,y)$ is used. Some authors (e.g. [45]) write the evolution of the PSDF as:

$$\frac{\partial c(x,t)}{\partial t} = -c(x,t)\int_0^x f(y,x-y)dy + 2\int_x^\infty c(y,t)f(x,y-x)dy, \quad (2.3)$$

while others (e.g. [6]) write:

$$\frac{\partial c(x,t)}{\partial t} = -\frac{1}{2}c(x,t)\int_0^x f(y,x-y)dy + \int_x^\infty c(y,t)f(x,y-x)dy. \quad (2.4)$$

The discrepancy between these two equations results from confusion over the exact relationship between $f(x,y)$, $g(x)$ and $\beta(x,y)$, and whether double counting is taken into account. The two different equations arise from using either:

$$f(y, x-y) = g(x)\beta(x,y), \tag{2.5}$$

or

$$f(y, x-y) = 2g(x)\beta(x,y), \tag{2.6}$$

to describe the formation rate of the daughter particles.

It is perhaps easiest to formulate the correct relationship using a discrete example; that of depolymerization, as described in [45]. Consider a polymer chain of k units. The chain has $k-1$ bonds that could each be broken on a binary breakage step. Assuming a uniformly distributed daughter chain length distribution, we can write $\beta(k,j) = \frac{1}{k-1}$. If, for simplicity, we set the breakage rate of this chain as $g(k) = (k-1)\ s^{-1}$ then we can work out the production rate of the daughter chains, $f(k-j, j)$: on average, in one second, we would expect one breakage to occur at each of the $k-1$ available breakage sites. Thus, in one second, we produce two chains of length one, two chains of length two and so on. We can write $f(k,j) = 2\ s^{-1}$ for $j = 1, 2, \ldots, k-1$, which is clearly twice the product of $g(k)$ and $\beta(k,j)$.

In general, we can write the relationship between $f(x,y)$, $g(x)$ and $\beta(x,y)$ in the continuous case as:

$$\begin{aligned} f(y, x-y) &= 2g(x)\beta(x,y) \\ \int_0^x f(y, x-y)dy &= 2\int_0^x g(x)\beta(x,y)dy \\ &= 2g(x)\int_0^x \beta(x,y)dy = 2g(x). \end{aligned} \tag{2.7}$$

Here we have made use of the fact that $\beta(x,y)$ is a density distribution normalised on the interval $[0,x]$.

Combining the two processes, coalescence and breakage, gives us the **population balance equation** that we wish to solve:

$$\begin{aligned} \frac{\partial c(x,t)}{\partial t} = \frac{1}{2}\int_0^x K(x-y,y)c(x-y,t)c(y,t)dy - \int_0^\infty K(x,y)c(x,t)c(y,t)dy \\ - c(x,t)g(x) + 2\int_x^\infty c(y,t)g(y)\beta(y,x)dy. \end{aligned} \tag{2.8}$$

3 Direct Simulation Algorithm

The general method for direct simulation of solutions to the coalescence-breakage equation has been give by Eibeck and Wagner [6]. Their paper concentrates on proving existence of a solution, and proposes a simulation algorithm for the coalescence-breakage case. This method can be described as direct simulation because a real particle ensemble is approximated by a stochastic particle system, where one stochastic particle, of size x, approximates a certain number concentration of real particles of that size.

For a stochastic system of n particles with sizes labelled:

$$x_i, \qquad i = 1, 2, \ldots, n, \tag{3.1}$$

the state of the system at any point can be represented by a measure-valued function, $p(x)$:

$$p(x) = \frac{1}{N} \sum_{i=1}^{n} \delta(x - x_i), \tag{3.2}$$

where N is a normalisation parameter that can be considered to represent a sample volume. N is referred to as the particle number. Simulation proceeds by performing coalescence or breakage jumps separated by waiting times, with functionals approximated by:

$$\begin{aligned} \int_0^\infty \phi(x)c(x)dx \ &\sim \ \int_0^\infty \phi(x) \frac{1}{N} \sum_{i=1}^{n} \delta(x - x_i)dx \\ &\sim \ \frac{1}{N} \sum_{i=1}^{n} \phi(x_i). \end{aligned} \tag{3.3}$$

We do not go into the details of the derivation of this method here, but refer to [6] and quote the algorithm.

1. Generate the initial state $U^N(0) = p \in \mathcal{S}^N$.

2. Wait an exponentially distributed time step τ with parameter

   $$\rho(p) = \rho_K(p) + \rho_g(p) = \frac{1}{2N} \sum_{1 \leq i \neq j \leq n} K(x_i, x_j) + \sum_{i=1}^{n} g(x_i),$$

   i.e. the waiting time, τ is chosen according to:

   $$\text{Prob}\{\tau(p) \geq s\} = \exp(-\rho(p) \cdot s), \qquad s \geq 0.$$

3. With probability:

   $$\frac{\rho_K(p)}{\rho_K(p) + \rho_g(p)},$$

   choose coalescence and hence go to step 4. Otherwise, choose breakage and go to step 5.

4. Perform a coalescence jump, i.e.

   (a) Choose a pair i, j according to the index distribution

$$\frac{K(x_i, x_j)}{2N\rho_K(p)}, \qquad 1 \leq i \neq j \leq n.$$

   (b) Remove the particles x_i and x_j and add a particle of size $x_i + x_j$.

   (c) Go to step 2.

5. Perform a breakage jump, i.e.

   (a) Choose an index i according to the distribution

$$\frac{g(x_i)}{\rho_g(p)}, \qquad 1 \leq i \leq n.$$

   (b) Choose a breakage part y according to the distribution

$$\beta(x_i, y),$$

   and remove the particle x_i and add particles of sizes y and $x_i - y$.

   (c) Go to step 2.

Note that for the purposes of this work we have simplified the Eibeck and Wagner DSA algorithm. The test case that we introduce in section 5 is simple enough that no majorant kernels [6, 7] need be introduced. Extension of the algorithm above to include majorant kernels and fictitious jumps is, however, straightforward. For mathematical rigour, we should also include a number truncation parameter, c_N. As the breakage process results in an increase in number of particles, an upper limit must be set on this number to ensure that the system remains within the solution space defined in the existence proof. For this paper however, we have chosen a test case with a particle number that tends to a finite steady state value, so we can, by effectively setting this truncation parameter arbitrarily large, ignore its effects.

4 Mass Flow Algorithm

Eibeck and Wagner [8] introduced the so-called **Mass Flow Algorithm** for stochastic solution of the Smoluchowski coagulation equation. In contrast to **direct simulation**, where one stochastic particle can be considered to represent one real particle in a theoretical sample space, the **mass flow** case uses one stochastic particle to represent mass concentration rather than number concentration.

More specifically, if $P(t, dx)$ is a measure valued solution of the population balance equation, then in the mass flow algorithm, we seek the solution:

$$Q(t, dx) = xP(t, dx), \tag{4.1}$$

where $Q(t,dx)$ is called the mass flow. Now, in approximating a functional of the concentration, $\int_0^\infty \phi(x)c(x)dx$, we use:

$$\begin{aligned}\int_0^\infty \phi(x)c(x)dx &\sim \int_0^\infty \frac{\phi(x)}{x}\frac{1}{N}\sum_{i=1}^n \delta(x-x_i)dx \\ &\sim \frac{1}{N}\sum_{i=1}^n \frac{\phi(x_i)}{x_i}. \end{aligned} \tag{4.2}$$

We now wish to extend the mass flow algorithm to include the breakage process. Following [44], we can write for the breakage case the weak integral version of the breakage equation:

$$\frac{d}{dt}\int_0^\infty \phi(x)Q(t,dx) = \int_0^\infty \int_0^\infty [\phi(y)-\phi(x)]\frac{2y}{x}g(x)\beta(x,y)dyQ(t,dx). \tag{4.3}$$

This leads to writing the Breakage Infinitesimal Generator:

$$\mathcal{A}_g^N \Phi(p) = \sum_{i=1}^n \int_0^\infty \left[\Phi(J_g(p,i,y)) - \Phi(p)\right]\frac{2y}{x_i}g(x_i)\beta(x_i,y)dy, \tag{4.4}$$

where the breakage operator is given by:

$$J_g(p,i,y) = p + \frac{1}{N}(\delta(x-y)-\delta(x-x_i)). \tag{4.5}$$

We note here that the symmetrical nature of $\beta(x,y)$ means that:

$$\int_0^x \phi(y)\beta(x,y)dy = \int_0^x \phi(x-y)\beta(x,x-y)dy, \tag{4.6}$$

and that:

$$\beta(x,y) = 0 \quad \text{for } x<y. \tag{4.7}$$

Thus, in constructing a simulation algorithm, we make use of the fact that:

$$\begin{aligned}\int_0^\infty y\beta(x,y)dy = \int_0^x y\beta(x,y)dy &= \int_0^x (x-y)\beta(x,y)dy \\ &= \frac{1}{2}\int_0^x x\beta(x,y)dy \\ &= \frac{x}{2}. \end{aligned} \tag{4.8}$$

We now expect to use as the distribution for selecting a breakage part, y, (as opposed to using $\beta(x,y)$ in the algorithm described above):

$$\frac{y\beta(x,y)}{\int y\beta(x,y)dy} = \frac{y\beta(x,y)}{\frac{x}{2}} = \frac{2y\beta(x,y)}{x}. \tag{4.9}$$

This suggests the following **simulation algorithm**:

1. Generate the initial state $U^N(0) = p \in \mathcal{S}^N$.

2. Wait an exponentially distributed time step τ with parameter

$$\begin{aligned}\rho(p) = \rho_K(p) + \rho_g(p) = \frac{1}{N}\sum_{i,j=1}^{n}\frac{K(x_i, x_j)}{x_j} + \sum_{i=1}^{n}\int_0^{x_i}\frac{2y}{x_i}g(x_i)\beta(x_i, y)dy \\ = \frac{1}{N}\sum_{i,j=1}^{n}\frac{K(x_i, x_j)}{x_j} + \sum_{i=1}^{n} g(x_i),\end{aligned}$$

   i.e. the waiting time, τ is chosen according to:

$$\text{Prob}\{\tau(p) \geq s\} = \exp(-\rho(p) \cdot s), \qquad s \geq 0.$$

3. With probability:

$$\frac{\rho_K(p)}{\rho_K(p) + \rho_g(p)},$$

   choose coalescence and hence go to step 4. Otherwise, choose breakage and go to step 5.

4. Perform a coalescence jump, i.e.

   (a) Choose a pair i, j according to the index distribution

$$\frac{K(x_i, x_j)}{N\rho_K(p)x_j}, \qquad 1 \leq i, j \leq n.$$

   (b) Remove the particle x_i and replace it with a particle of size $x_i + x_j$.

   (c) Go to step 2.

5. Perform a breakage jump, i.e.

   (a) Choose an index, i, according to the distribution

$$\frac{g(x_i)}{\sum_{i=1}^{n} g(x_i)}.$$

   (b) Choose a breakage part y according to the distribution

$$\frac{2y\beta(x_i, y)}{x_i},$$

   and replace the particle of size x_i with one of size y.

   (c) Go to step 2

Note that, in a similar fashion to the algorithm presented in section 3, we have simplified this algorithm. Eibeck and Wagner, in their original presentation of the mass flow algorithm, include a maximum size truncation parameter, b_N. Similar to the particle number truncation parameter, c_N, this parameter exists to prevent any individual particle growing too large and thus growing larger than the solution space defined in the existence proof. In coagulation only problems, where the particles grow larger without limit, this parameter can be useful in predicting gelation properties, but in our simple coalescence-breakage problem, we can arbitrarily set b_N large enough that it does not affect our simulation.

5 Test Case

Consider the case where coalescence and breakage occur according to:

$$\begin{aligned} K(x,y) &= 1 && (5.1)\\ f(x,y) &= \frac{2}{x+y}, && (5.2)\\ \text{i.e.}\quad g(x)=1 & && (5.3)\\ \beta(x,y)=\frac{1}{x}. & && (5.4) \end{aligned}$$

The population balance then takes the form:

$$\begin{aligned} \frac{\partial c(x,t)}{\partial t} &= -c(x,t) + 2\int_x^\infty c(y,t)\frac{1}{y}dy \\ &\quad - c(x,t)\int_0^\infty c(y,t)dy + \frac{1}{2}\int_0^x c(x-y,t)c(y,t)dy. \qquad (5.5) \end{aligned}$$

We use the initial condition:

$$c(x,0) = \begin{cases} 1 & : \ x=1 \\ 0 & : \ \text{otherwise,} \end{cases} \qquad (5.6)$$

and use the method of moments (see Appendix A) to find:

$$\begin{aligned} m_0(t) &= \frac{2}{1+e^{-t}} \\ m_1(t) &= 1 \\ m_2(t) &= 3-2e^{-t/3} \\ m_3(t) &= 18-36e^{-t/3}+19e^{-t/2} \\ m_4(t) &= 165-69e^{-3t/5}+760e^{-t/2}-675e^{-t/3}-180e^{-2t/3} \\ m_5(t) &= \frac{4095}{2}-\frac{21853}{2}e^{-2t/3}-5175e^{-3t/5}+26220e^{-t/2} \\ &\qquad -14445e^{-t/3}-180te^{-2t/3}+2280e^{-5t/6}. \qquad (5.7) \end{aligned}$$

This is a convenient analytical solution to use for testing the stochastic algorithm. Note that we can treat all quantities as dimensionless in this case.

6 Data Storage

The method used to store the stochastic particle sizes and associated functionals has a profound influence on the efficiency of generation of required probability distributions. Often there is a trade-off between economy of storage and efficiency of generation, but we generally value low CPU times over low storage requirements.

6.1 Bins

Arranging the stochastic particles by size into bins is a method favoured by Eibeck and Wagner [7, 8] for the case of coagulation only. In this method, the bin size bounds,

$$0 =: b_0 < b_1 < \ldots, \tag{6.1}$$

are given by

$$b_k = \beta^{k-1}, \qquad \beta > 1, \quad k = 1, 2, \ldots. \tag{6.2}$$

Then the bin index is chosen by linear search, and the particle index within a bin is chosen by acceptance-rejection, with a minimum efficiency of:

$$\frac{1}{\beta^{|\varepsilon|}}, \tag{6.3}$$

when the required distribution to be generated is:

$$\frac{x_i^{\varepsilon}}{\sum_{i=1}^{n} x_i^{\varepsilon}}. \tag{6.4}$$

The result is that, for n particles, the time taken for the index generation step is proportional to $\log(n)$. However, after each stochastic jump, the bins must be updated. The process in which an empty space is removed from an array to reflect a particle being removed from that bin takes time of order n. In this case, for large particle numbers, the overall process can take time of order n^2 [13].

A further problem encountered when the bin method is used is a consideration of the range of bin sizes. Sensible selection of β is essential to optimise CPU time, but this often results in larger storage requirements. Furthermore, for the coagulation-only case, there is a well-defined minimum particle size (the smallest particle in the initial distribution). This minimum particle size might not exist in a breakage problem, in which case a lowest bin must be set, which then contains all particles smaller than some cut-off size. The selection of β and the minimum bin-size in order to minimise the CPU time of the simulation is an extra and undesirable problem and so we look for a system of storage/index generation that is more appropriate for the coalescence-breakage process.

6.2 Binary Search

A binary search method of generating distributions is particularly suitable for use within the mass flow algorithm. The number of stochastic particles remains constant (unless a particle grows larger than b_N, but this is rare, especially if b_N is chosen large enough), so if a binary tree is constructed, it remains the same height throughout the simulation.

Figure 1 shows an example of the construction of a binary tree. For simplicity, it is of height 3, so the particle number is $2^3 = 8$. We are interested in generating an index, i, according to the distribution:

$$\frac{\phi(x_i)}{\sum_{i=1}^{n} \phi(x_i)}. \tag{6.5}$$

The function $\phi(x)$ is often simply the value x raised to some power, but it need not be restricted to such cases.

Note that in figure 1 the particles (on the lowest row of the tree) are labelled $i = 0, 1, \ldots, 7$, i.e. from 0 to $n - 1$, so from now on in this section, the numbering of particle indices will reflect this.

To choose a particle index, $i = 0, 1, \ldots, 7$, we generate a uniform random deviate on [0,1] and multiply it by $\sum_{i=0}^{7} \phi(x_i)$. We now check if this quantity is greater than the value of node$_1$ on the tree. If it is, we move down the tree to the right (1). If not, we move down to the tree to the left (0). If we move left, we repeat the process. If we move right, then we need to update our comparison quantity, by subtracting the value of node$_1$. Then we repeat the process down to the bottom. Note that by recording the value of 0 or 1 each time we move down a level, we end up with a binary representation of the particle index we are interested in (hence the renumbering of the particles). Note also that this binary representation shows which nodes on the tree need to be updated after a stochastic jump. Whenever a 0 is recorded as we move down the tree, the node that has just been passed will need updating.

The index generation and system update steps both now take CPU time of order $\log(n)$, so the overall CPU time required increases as $n \log(n)$ regardless of particle number.

A comparison of CPU time for the two methods is shown in **Figure 2**. It can be seen that, despite similar runtimes for low particle numbers, the CPU time for the bins method increases slightly faster, and the two methods diverge at around $n = 10^4$. Even for particle numbers up to 10^6, the CPU time for the binary search method increases just faster than linearly with particle number.

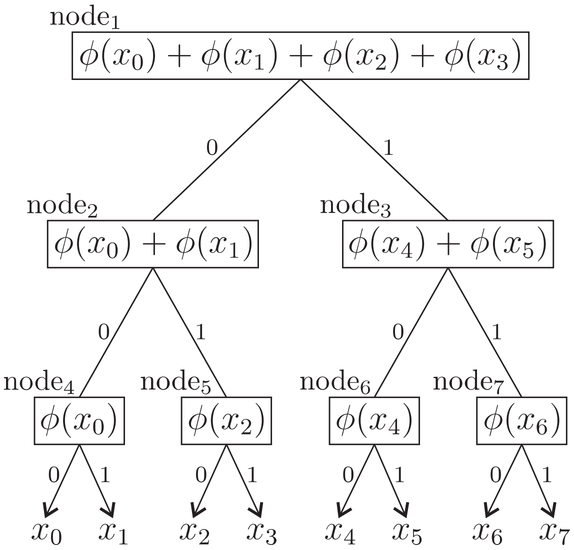

Figure 1: *A binary tree, of height 3.*

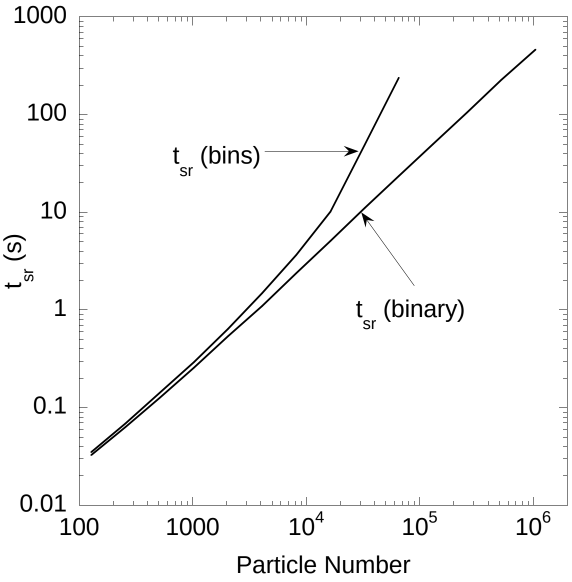

Figure 2: *Single run CPU time comparison for the bins and binary methods.*

7 Numerical Results

Figure 3 and **Figure 4** show sample results of stochastic simulation of the coalescence-breakage equation for the test case described above. The results are averaged over a number of repeated runs, L, and confidence bounds are derived (e.g. [13]) within which we can be 99.9% sure that the correct solution lies.

It is immediately evident that DSA and MFA have varying success when it comes to predicting different moments of the particle size distribution. For the zeroth moment, despite a much larger particle number (2^{17} as opposed to 2^{11}), MFA has very wide confidence intervals that do not always encompass the analytical solution. DSA on the other hand, even for this moderate particle number, gives a good prediction of the zeroth moment. For the fifth moment, the situation is reversed. Now (with the same particle number, 2^{11}) MFA gives a much better prediction of the fifth moment, while the confidence bands for DSA are large and fluctuate widely.

This behaviour can be explained by considering the way the functionals of the particle size distribution are approximated (recall Equations 3.3 and 4.2):

$$\int_0^\infty \phi(x)c(x)dx \sim \underbrace{\frac{1}{N}\sum_{i=1}^{n}\phi(x_i)}_{\text{for DSA}} \quad \text{or} \quad \underbrace{\frac{1}{N}\sum_{i=1}^{n}\frac{\phi(x_i)}{x_i}}_{\text{for MFA}}. \tag{7.1}$$

When approximating the zeroth moment using MFA, it is actually the sum of the -1th powers of the stochastic particle sizes that is calculated and therefore the smallest particles have most influence over the result. The appearance, due to breakage, or disappearance, due to coalescence, of a very small particle will result in a large step change in the zeroth moment. To some extent, MFA is self compensating in this regard. The coalescence step is based on a rate $\frac{K(x_i,x_j)}{x_j}$; the $\frac{1}{x_j}$ factor means that small particles are more likely to be chosen for coalescence than they are in DSA. Also, in choosing a new particle size in a breakage step, the $y\beta(x_i,y)$ distribution (as opposed to $\beta(x_i,y)$) means that a small particle is less likely to be produced than it is in DSA. But even with these compensatory mechanisms, the varying presence of very small particles causes large fluctuations in the prediction of the zeroth moment. This is a problem that was not encountered by Eibeck and Wagner when they studied the mass flow algorithm for the coagulation-only case. Without breakage, there is no opportunity for the production of very small particles, so no large fluctuations in zeroth moment are observed.

The higher moments are where the successful variance reduction of the mass flow algorithm begins to show. For the same particle number—and similar CPU times—, MFA exhibits much narrower confidence bands than for DSA. Thus, based on these initial observations, it would seem that DSA holds the advantage for the zeroth moment and MFA is more efficient at predicting higher (>1) moments.

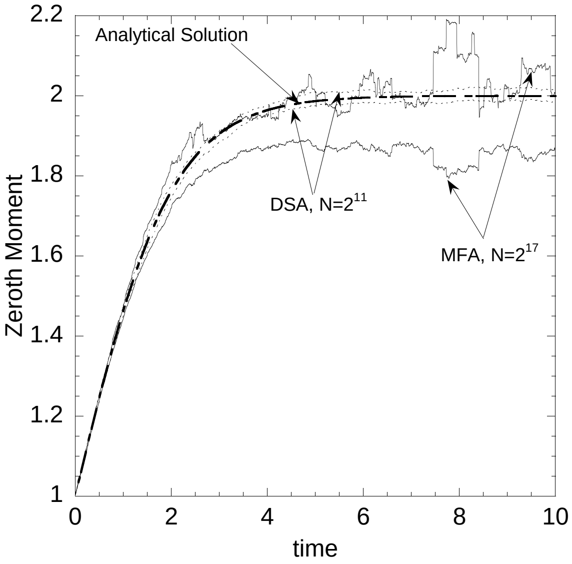

Figure 3: *Comparison of the zeroth moment predicted by Direct Simulation (DSA) and Mass Flow (MFA).*

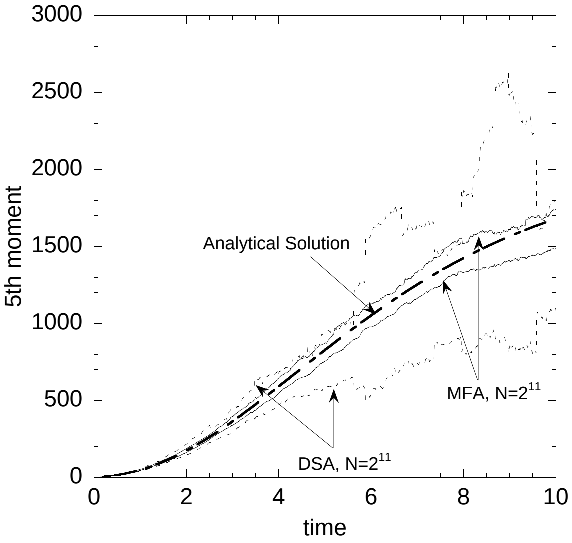

Figure 4: *Comparison of the fifth moment predicted by Direct Simulation (DSA) and Mass Flow (MFA).*

7.1 Convergence Properties

It is useful to consider the convergence properties of the two algorithms, i.e. how does the systematic error of the predicted functionals depend on the particle number, N? This will then give an idea of the CPU time required for each method to produce a fixed error and will indicate which is faster. For these studies, we approximate the systematic error by the average error over the simulation time, and the systematic error by the maximum width of the confidence interval within that time.

For the MFA case, **Figure 5** shows the convergence properties of the 2nd to 5th moments. As expected, the systematic error decreases as the inverse of the particle number. In order to produce this figure, it is necessary to perform simulations where the statistical error (denoted by the error bars) is small enough that the trend in systematic error can be detected. This, for the MFA case, involves a very small number of particles and a very large number of trajectories generated.

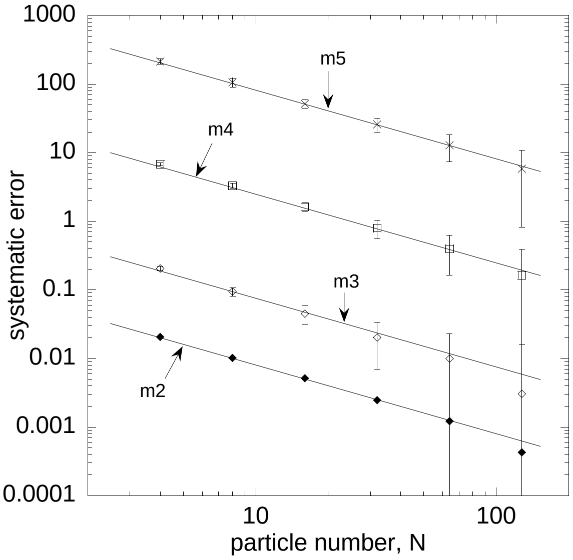

Figure 5: *Convergence to the analytical solution of the 2nd to 5th moments predicted by the mass flow algorithm. The solid lines show slopes of -1. In this study,* $N \times L$ *is kept constant at 104857600.*

A somewhat different result occurs when we examine the convergence properties of the zeroth moment predicted by the mass flow algorithm (**Figure 6**). Using the same product of number of particles and number of runs (104857600), the error bars are now so small as not to show up on the plot for low particle numbers. Furthermore, the convergence is much slower; the systematic error is decreasing approximately as $N^{-1/3}$.

An illustration of what this convergence behaviour actually means is shown in **Figure 7** for particle numbers of $2^5 \rightarrow 2^8$. Here, the large number of repeated runs removes the fluctuations in the predicted zeroth moment, thus showing the large systematic error in the results.

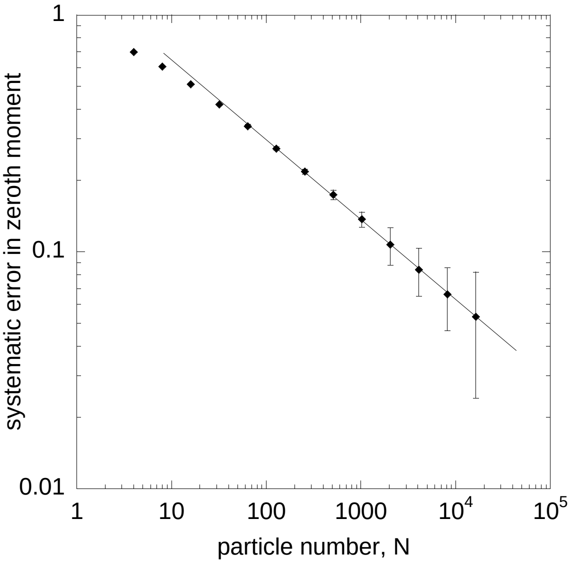

Figure 6: *Convergence to the analytical solution of the zeroth moment predicted by the mass flow algorithm. The solid line shows a slope of -1/3. In this study, $N \times L$ is kept constant at 104857600.*

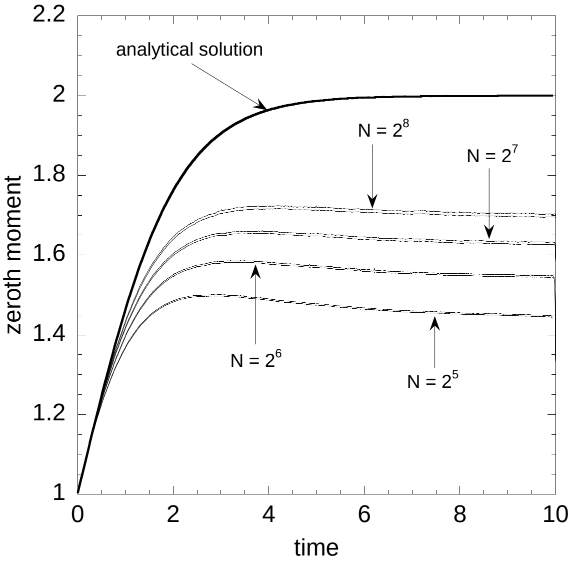

Figure 7: *Convergence to the analytical solution of the zeroth moment predicted by the mass flow algorithm. In this study, $N \times L$ is kept constant at 104857600. The number of repeated runs is sufficient to smooth out any fluctuations in the confidence intervals, but the systematic error is still significant.*

When we examine the same convergence properties for the direct simulation algorithm, we find that all the moments converge at the same rate, proportional to $\frac{1}{N}$ (**Figure 8**). From this study we can now make a comparison of the two algorithms: which algorithm takes less CPU time to predict a given moment with a given error?

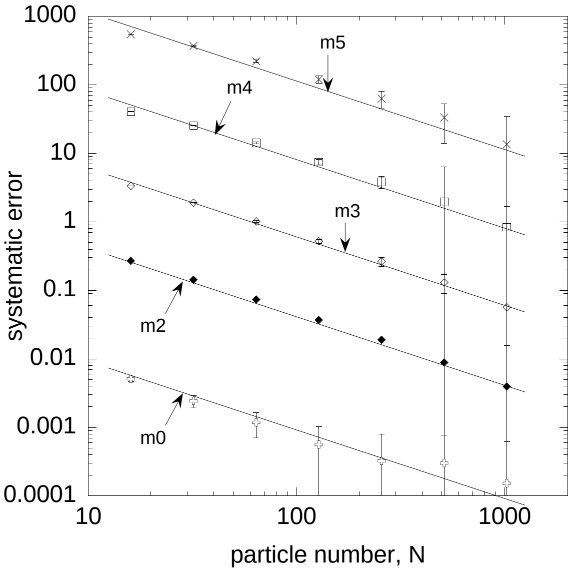

Figure 8: *Convergence to the analytical solution of the 2nd to 5th moments predicted by the direct simulation algorithm. The solid lines show slopes of -1. In this study, $N \times L$ is kept constant at 104857600.*

7.2 DSA or MFA?

Figures 9 − 11 give further evidence to that suggested by the initial inspection of the simulation results: MFA gives better predictions for higher moments, but DSA is better for predicting the zeroth moment. The difference in CPU time is up to two orders of magnitude in favour of MFA for the higher moments, and is considerably more than that in favour of DSA for the zeroth moment.

8 Minimum Particle Size

Frequently, a real system in which coalescence and breakage occur will have a minimum particle size. This can be a well-defined monomer unit as in the case of polymerisation-depolymerisation (e.g. [45]), or can be introduced due to stability considerations (e.g. [26]). As well as better approximating a real system, incorporating this minimum particle size into the simulation algorithm will have a positive effect on the viability of using the mass flow algorithm to predict the zeroth moment; having fewer very small particles will reduce the size of the fluctuations in the zeroth moment. To examine this effect, we impose a minimum stable particle size

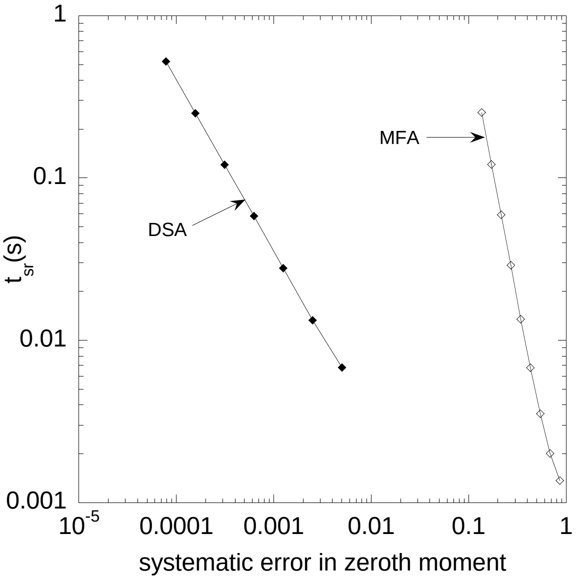

Figure 9: *Single run CPU time required to predict the zeroth moment with a given systematic error.*

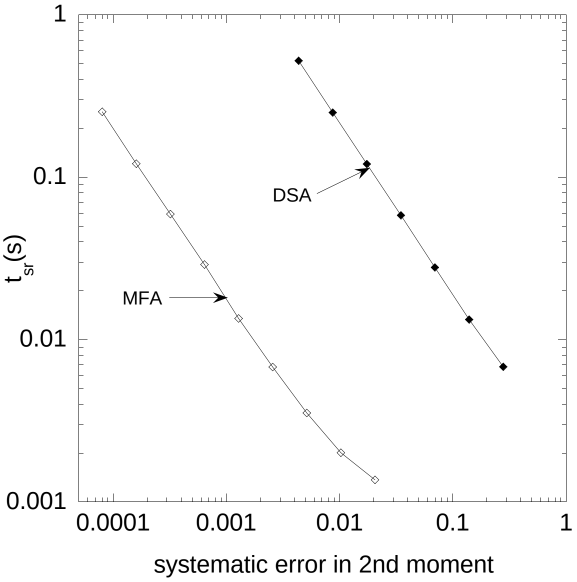

Figure 10: *Single run CPU time required to predict the second moment with a given systematic error.*

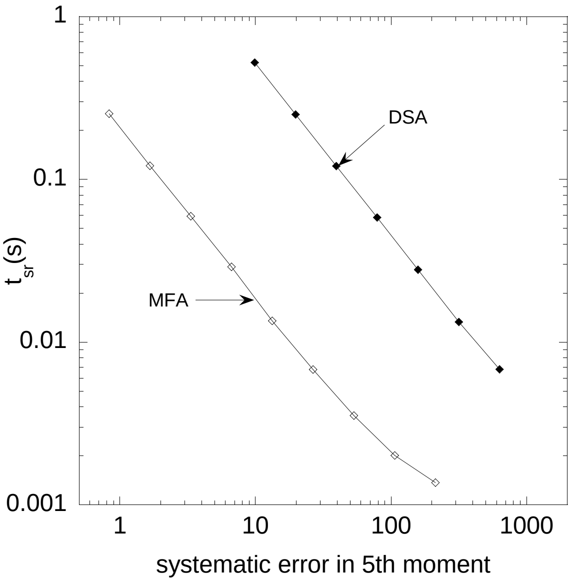

Figure 11: *Single run CPU time required to predict the fifth moment with a given systematic error.*

on the test case and simulate the zeroth moment using both DSA and MFA. There is no longer a simple analytical solution for the moments, so we instead compare the results predicted by MFA with the converged solution predicted by DSA (which has been shown to be reliable).

There are two ways of considering how the minimum particle size, x_{min}, affects the simulation:

1. No particle smaller than x_{min} can exist. Thus we can infer that breakage of a particle smaller than $2x_{min}$ cannot occur. This would be the case if there were a well-defined monomer unit (but note also that if each particle is made up of an integer number of monomer units the problem would be discrete rather than continuous).

2. A particle smaller than x_{min} can exist and can take part in coalescence events, but will not break into smaller particles. This is the approach used by e.g. Tsouris and Tavlarides [39]; they refer to this minimum size as a maximum stable drop diameter, i.e. the largest drop size for which no breakage can occur.

8.1 Case 1

For case 1 we need to alter the probability density function used to pick the sizes of daughter particles when a breakage event occurs. We no longer allow any particle smaller than x_{min} or larger than $x - x_{min}$ to be formed by breakage of a particle of size x. Clearly, what was previously a uniform distribution over the size range $[0, x]$

becomes a uniform distribution over the range $[x_{min}, x - x_{min}]$, so we have:

$$\beta(x,y) = \begin{cases} \frac{1}{x-2x_{min}} & : \; x_{min} < y < x - x_{min} \\ 0 & : \; \text{otherwise.} \end{cases} \tag{8.1}$$

This leads to, for MFA simulation:

$$\frac{y\beta(x,y)}{\int y\beta(x,y)dy} = \begin{cases} \frac{2y}{x(x-2x_{min})} & : \; x_{min} < y < x - x_{min} \\ 0 & : \; \text{otherwise.} \end{cases} \tag{8.2}$$

In both these cases, setting $x_{min} = 0$ returns us to the case where no minimum size is imposed.

Also, as mentioned above, we must only choose particles larger than $2x_{min}$ for breakage, so as not to produce particles smaller than x_{min}.

8.2 Case 2

For case 2, it is not necessary to alter the daughter particle size distribution. Our only restriction in this case is not to pick any particle smaller than x_{min} for breakage. This is the most common way a minimum particle size is imposed for liquid-liquid systems, so this is the case we examine. The only alteration that needs to be made to the simulation algorithm is to add a null event in the case that a particle smaller than x_{min} is chosen for breakage.

Figure 12 shows a comparison of the zeroth moment predicted by MFA with the converged solution predicted by DSA for various minimum sizes. In this study, MFA is performed with particle number $N = 2^{15}$ and only 50 repetitions. The large systematic error has now decreased to the point where the converged solution lies within the confidence bounds, even though the fluctuations are quite large. As the minimum particle size increases, the fluctuations decrease, and for $x_{min} = 1$, MFA gives a good prediction of the zeroth moment. This is consistent with the results of Eibeck and Wagner [8] for the coagulation only case, where no particles smaller than $x = 1$ occur. The single run simulation CPU time, t_{sr}, is now of the order of a minute.

9 Conclusions and Discussion

In this paper, two stochastic algorithms have been used to simulate solutions to the population balance equation in which coalescence and binary breakage are the processes involved. Direct Simulation Algorithm (DSA) was used as proposed by Eibeck and Wagner [6]. Mass Flow Algorithm (MFA), which was first used by the same authors [8] for the coagulation only case, has been extended to include

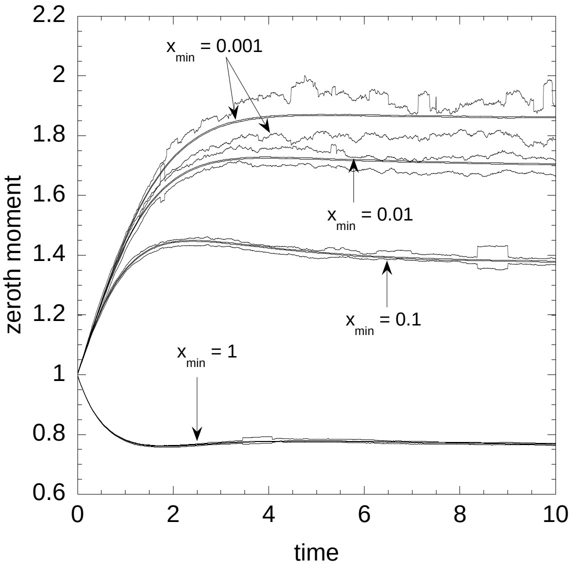

Figure 12: *Behaviour of the Mass Flow Algorithm with the imposition of a minimum particle size, x_{min}. The confidence bounds are for the solutions given by MFA. The solid lines are the converged solutions given by DSA.*

the breakage process. We also introduced a binary search method of probability distribution generation, leading to improved efficiency of simulation.

The two algorithms were applied to a test case—for which an analytical solution was calculated—to determine the relative merits of each. For the coagulation only case, Eibeck and Wagner found excellent improvement in efficiency and good variance reduction for MFA as compared to DSA. With breakage included, MFA does not predict the zeroth moment so well. As it involves summing the -1th powers of the sizes of the stochastic particles, the presence of very small particles can lead to large fluctuations in the predicted zeroth moment.

This zeroth moment predicted by MFA exhibits markedly different convergence characteristics to the other moments studied. For every moment predicted by DSA and for every moment greater than the 1st (the 1st moment is by definition constant for this mass-preserving process) predicted by MFA, the systematic error decreases as the inverse of the particle number, N. This behaviour is common to stochastic simulation of many systems (e.g. [19, 13]). In contrast, the systematic error in the zeroth moment predicted by MFA decreases as $N^{-1/3}$. For moderate particle numbers, this systematic error, along with large fluctuations, precludes the use of MFA for stochastic simulation of the zeroth moment.

The higher moments (2nd and higher) are predicted well by MFA, and with a significant decrease in CPU time required—for a given error—as compared with DSA. For a given particle number—and therefore comparable simulation times—the systematic errors in the MFA-predicted moments are much smaller than those predicted by DSA. Hence, if the higher moments of the particle size distribution are the required functionals, MFA is clearly the better option.

The effect of the imposition of a minimum particle size, x_{min}, on the system was also investigated. In common with several models of liquid-liquid systems (e.g. [39, 26]), the minimum particle size is taken to be the size below which no further breakage can take place. The result of this modification to the algorithm is that there are fewer very small particles contained in the stochastic particle system. Hence, the problems associated with MFA are reduced and, for large enough x_{min}, MFA can perform as well as DSA.

A Solution of Test Case

To solve the equation:

$$\frac{\partial c(x,t)}{\partial t} = -c(x,t) + 2\int_x^\infty c(y,t)\frac{1}{y}dy \\ - c(x,t)\int_0^\infty c(y,t)dy + \frac{1}{2}\int_0^x c(x-y,t)c(y,t)dy, \quad \text{(A.1)}$$

we use the method of moments (e.g. [9]), i.e. we multiply by x^k and integrate over all x:

$$\int_0^\infty x^k \frac{\partial c(x,t)}{\partial t}dx = \frac{d}{dt}m_k(t) = -\int_0^\infty x^k c(x,t)dx + 2\int_0^\infty x^k \int_x^\infty c(y,t)\frac{1}{y}dy\ dx \\ - \int_0^\infty x^k c(x,t)\int_0^\infty c(y,t)dy\ dx + \frac{1}{2}\int_0^\infty x^k \int_0^x c(x-y,t)c(y,t)dy\ dx. \quad \text{(A.2)}$$

Treating each term on the RHS in turn:

$$\int_0^\infty x^k c(x,t)dx = m_k(t). \quad \text{(A.3)}$$

$$\begin{aligned} \int_{x=0}^\infty x^k \int_{y=x}^\infty c(y,t)\frac{1}{y}dy\ dx &= \int_{y=0}^\infty c(y,t)\frac{1}{y}\int_{x=0}^y x^k dx\ dy \\ &= \int_0^\infty c(y,t)\frac{1}{y}\frac{y^{k+1}}{k+1}dy \\ &= \int_0^\infty c(y,t)\frac{y^k}{k+1}dy = \frac{m_k(t)}{k+1}. \quad \text{(A.4)} \end{aligned}$$

$$\int_{x=0}^\infty x^k c(x,t)\int_{y=0}^\infty c(y,t)dy\ dx = m_k(t)m_0(t). \quad \text{(A.5)}$$

$$\begin{aligned} \int_{x=0}^\infty x^k \int_{y=0}^x c(x-y,t)c(y,t)dy\ dx &= \int_{y=0}^\infty c(y,t)\int_{x=y}^\infty x^k c(x-y,t)dy\ dx \\ &= \int_{y=0}^\infty c(y,t)\int_{z=0}^\infty (z+y)^k c(z,t)dy\ dz \\ &= \int_{y=0}^\infty c(y,t)\int_{z=0}^\infty \sum_{r=0}^k \binom{k}{r} z^{k-r}y^r c(z,t)dy\ dz \\ &= \sum_{r=0}^k \binom{k}{r}\int_{y=0}^\infty c(y,t)y^r dy \int_{z=0}^\infty c(z,t)z^{k-r}dz \\ &= \sum_{r=0}^k \binom{k}{r} m_r(t)m_{k-r}(t). \quad \text{(A.6)} \end{aligned}$$

We then have differential equations for $m_k(t)$:

$$\begin{aligned}
\frac{d\,m_0(t)}{dt} &= m_0(t) - \frac{m_0^2(t)}{2} && \text{(A.7)}\\
\frac{d\,m_1(t)}{dt} &= 0 && \text{(A.8)}\\
\frac{d\,m_k(t)}{dt} &= \frac{1-k}{1+k} m_k(t) + \frac{1}{2}\sum_{r=1}^{k-1}\binom{k}{r} m_r(t) m_{k-r}(t) \quad k = 2, 3, \ldots . && \text{(A.9)}
\end{aligned}$$

For the initial condition:

$$c(x,0) = \begin{cases} 1 & : \ x = 1 \\ 0 & : \ \text{otherwise,} \end{cases} \tag{A.10}$$

i.e. $m_k(0) = 1, \quad k = 0, 1, 2, \ldots$, we have a series of first order differential equations that are separable (A.7), trivial (A.8) or have constant coefficients once the lower order moments have been found (A.9). These can be solved to give:

$$\begin{aligned}
m_0(t) &= \frac{2}{1+e^{-t}}\\
m_1(t) &= 1\\
m_2(t) &= 3 - 2e^{-t/3}\\
m_3(t) &= 18 - 36e^{-t/3} + 19e^{-t/2}\\
m_4(t) &= 165 - 69e^{-3t/5} + 760e^{-t/2} - 675e^{-t/3} - 180e^{-2t/3}\\
m_5(t) &= \frac{4095}{2} - \frac{21853}{2}e^{-2t/3} - 5175e^{-3t/5} + 26220e^{-t/2}\\
&\qquad -14445e^{-t/3} - 180te^{-2t/3} + 2280e^{-5t/6}. && \text{(A.11)}
\end{aligned}$$

References

[1] I. Alatiqi, G. Aly, F. Mjalli, and C. J. Mumford. Mathematical modeling and steady-state analysis of a Scheibel extraction column. *Can. J. Chem. Eng.*, 73:523–533, 1995.

[2] P. Angeli and G. F. Hewitt. Drop size distributions in horizontal oil-water dispersed flows. *Chem. Eng. Sci.*, 55(16):3133–3143, 2000.

[3] C. A. Coulaloglou and L. L. Tavlarides. Description of interaction processes in agitated liquid-liquid dispersions. *Chem. Eng. Sci.*, 32:1289–1297, 1977.

[4] R. B. Diemer and J. H. Olson. A moment methodology for coagulation and breakage problems: Part 1 – Analytical solution of the steady-state population balance. *Chem. Eng. Sci.*, 57(12):2193–2209, 2002.

[5] R. B. Diemer and J. H. Olson. A moment methodology for coagulation and breakage problems: Part 2 – Moment models and distribution reconstruction. *Chem. Eng. Sci.*, 57(12):2211–2228, 2002.

[6] A. Eibeck and W. Wagner. Approximative solution of the coagulation-fragmentation equation by stochastic particle systems. *Stoch. Anal. App.*, 18(6):921–948, 2000.

[7] A. Eibeck and W. Wagner. An efficient stochastic algorithm for studying coagulation dynamics and gelation phenomena. *SIAM J. Sci. Comput.*, 22(3):802–821, 2000.

[8] A. Eibeck and W. Wagner. Stochastic particle approximations for Smoluchowski's coagulation equation. *Ann. Appl. Probab.*, 11(4):1137–1165, 2001.

[9] M. Frenklach. Dynamics of discrete distribution for Smoluchowski coagulation model. *J. Coll. Int. Sci.*, 108(1):237–242, 1985.

[10] M. Frenklach and S. Harris. Aerosol dynamics modeling using the method of moments. *J. Coll. Int. Sci.*, 118(1):252–262, 1986.

[11] F. Gelbard and J. H. Seinfeld. Numerical solution of the dynamic equation for particulate systems. *J. Comp. Phys.*, 28:357–, 1978.

[12] D. T. Gillespie. The stochastic coalescence model for cloud droplet growth. *J. Atmospheric Sci.*, 29:1496–1510, 1972.

[13] M. J. Goodson and M. Kraft. An efficient stochastic algorithm for simulating nano-particle dynamics. *J. Comp. Phys.*, In Press.

[14] D. A. Grosschmidt, H. Bockhorn, M. J. Goodson, and M. Kraft. Two approaches to the simulation of silica particle synthesis. *Proc. Comb. Inst.*, 29, In Press.

[15] H. Haverland, A. Vogelpohl, C. Gourdon, and G. Casamatta. Simulation of fluid dynamics in a pulsed sieve-plate column. *Chem. Eng. Technol.*, 10:151–157, 1987.

[16] G. M. Hidy and J. R. Brock. *The Dynamics of Aerocolloidal Systems.* Pergamon, Oxford, 1970.

[17] M. J. Hounslow, R. L. Ryall, and V. R. Marshall. A discretized population balance for nucleation, growth and aggregation. *AIChE J.*, 34(11):1821–1832, 1988.

[18] H. M. Hulbert and S. Katz. Some problems in particle technology. A statistical mechanical formulation. *Chem. Eng. Sci.*, 19:555–574, 1964.

[19] M. Kraft and W. Wagner. Numerical study of a stochastic particle method for homogeneous gas phase reactions. Technical Report 570, Weierstrass Institute for Applied Analysis and Stochastics, 2000.

[20] A. Kumar and S. Hartland. Prediction of drop size in rotating disc extractors. *Can. J. Chem. Eng.*, 64:915–924, 1986.

[21] S. Kumar and D. Ramkrishna. On the solution of population balance equations by discretization – I. a fixed pivot technique. *Chem. Eng. Sci.*, 51(8):1311–1332, 1996.

[22] S. Kumar and D. Ramkrishna. On the solution of population balance equations by discretization – II. a moving pivot technique. *Chem. Eng. Sci.*, 51(8):1333–1342, 1996.

[23] K. Lee and T. Matsoukas. Simultaneous coagulation and break-up using constant-N Monte-Carlo. *Powder Tech.*, 110:82–89, 2000.

[24] Y. Lin, K. Lee, and T. Matsoukas. Solution of the population balance equation using constant-number Monte Carlo. *Chem. Eng. Sci.*, 57:2241–2252, 2002.

[25] J. D. Litster, D. J. Smit, and M. J. Hounslow. Adjustable discretized population balance for growth and aggregation. *AIChE J.*, 41(3):591–603, 1995.

[26] S. P. Liu and D. M. Li. Drop coalescence in turbulent dispersions. *Chem. Eng. Sci.*, 54(23):5667–5675, 1999.

[27] Y. Liu and I. T. Cameron. A new wavelet-based method for the solution of the population balance equation. *Chem. Eng. Sci.*, 56(18):5283–5294, 2001.

[28] S. Mohanty. Modeling of liquid-liquid extraction column: A review. *Rev. Chem. Eng.*, 16(3):199–248, 2000.

[29] G. Narsimhan, J. P. Gupta, and D. Ramkrishna. A model for transitional breakage probability of droplets in agitated lean liquid-liquid dispersions. *Chem. Eng. Sci.*, 34:257–264, 1979.

[30] M. Nicmanis and M. J. Hounslow. Finite-element methods for steady-state population balance equations. *AIChE J.*, 44(10):2258–2272, 1998.

[31] D.P. Patil and J. R. G. Andrews. An analytical solution to continuous population balance model describing floc coalescence and breakage – A special case. *Chem. Eng. Sci.*, 53(3):599–601, 1998.

[32] D. Ramkrishna. Analysis of population balance – IV. The precise connection between Monte Carlo simulation and population balances. *Chem. Eng. Sci.*, 36:1203–1209, 1980.

[33] D. Ramkrishna. The status of population balances. *Rev. Chem. Eng.*, 3:49–95, 1985.

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

[35] A. D. Randolph and M. A. Larson. *Theory of Particulate Processes.* Academic Press, London, 1988.

[36] K. K. Sabelfeld, S. V. Rogansinsky, A. A. Kolodko, and A. I. Levykin. Stochastic algorithms for solving Smoluchovsky coagulation equation and applications to aerosol growth simulation. *Monte Carlo Meth. App.*, 2(1):41–87, 1996.

[37] C. A. Sleicher. Axial mixing and extraction efficiency. *AIChE J.*, 5:145–149, 1959.

[38] M. Smith and T. Matsoukas. Constant-number Monte Carlo simulation of population balances. *Chem. Eng. Sci.*, 53(9):1777–1786, 1998.

[39] C. Tsouris and L. L. Tavlarides. Breakage and coalescence models for drops in turbulent dispersions. *AIChE J.*, 40(3):395–406, 1994.

[40] K. J. Valentas and N. R. Amundson. Breakage and coalescence in dispersed phase systems. *Ind. Eng. Chem. Fund.*, 5(4):533–542, 1966.

[41] K. J. Valentas, O. Bilous, and N. R. Amundson. Analysis of breakage in dispersed phase systems. *Ind. Eng. Chem. Fund.*, 5(2):271–279, 1966.

[42] M. Vanni. Approximate population balance equations for aggregation-breakage processes. *J. Coll. Int. Sci.*, 221:143–160, 2000.

[43] M. von Smoluchowski. Drei Vorträge über Diffusion, Brownsche Molekularbewegung und Koagulation von Kolloidteilchen. *Phys. Z.*, 17:557–571, 585–599, 1916.

[44] W. Wagner. Stochastic, analytic and numerical aspects of coagulation processes. *Technical Report 697, Weierstraß-Institut für Angewandte Analysis und Stochastik, Berlin*, 2001.

[45] R. M. Ziff and E. D. McGrady. The kinetics of cluster fragmentation and depolymerisation. *J. Phys. A: Math. Gen.*, 18:3027–3037, 1985.