Preprint ISSN 1473 – 4273

Weighted particle methods for the Smoluchowski coagulation equation

Robert I A Patterson, and Markus Kraft [1]

released: 28 June 2007

[1] Department of Chemical Engineering
University of Cambridge
Pembroke Street
Cambridge CB2 3RA
UK
E-mail: mk306@cam.ac.uk

Preprint No. 47

Key words and phrases: coagulation, weights, Monte Carlo, variance reduction

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

The Smoluchowski coagulation with additional linear terms is formulated on an explicitly weighted space. The use of such a space extends the ideas behind the mass flow algorithm [A. Eibeck and W. Wagner, Ann. Appl. Prob. (2001) 11:1137-1165] and allows the derivation of a range of new stochastic particle algorithms. Two such algorithms, which seem particularly natural, are implemented and successfully validated. The weighted algorithms are then compared to each other and to a version direct simulation algorithm, which uses particle doubling as a variance reduction technique. One weighted algorithm is found to consistently outperform the other by a significant amount. Applications for which the better weighted algorithm outperforms direct simulation are demonstrated, along with situations where the converse is true.

Contents

1 Introduction

In the basic Direct Simulation Monte Carlo (DSMC) approach to the Smoluchowski and Boltzmann equations, every computational particle represents the same number of physical particles. The accuracy with which a quantity is computed depends on the number of computational particles used. In spatially resolved gas simulation, regions with few physical particles are not accurately simulated. To deal with this problem Rjasanow and Wagner [28] introduced a Stochastic Weighted Particle Method (SWPM) in which computational particles were tagged with a statistical weight—the number of physical particles they represent. Using this technique a larger number of computational particles with smaller statistical weights could be used in regions of low density than was the case with basic DSMC, without increasing the time spent simulating regions of higher density. As a result, computational accuracy could be controlled separately for each spatial cell (for example kept approximately equal at all locations) and independently of the values of physical properties being simulated.

A similar kind of difficulty can occur in spatially homogeneous particle coagulation problems. For these problems it is found that the resolution of the high end of the particle size distribution in a DSMC simulation can be inadequate because there are very few computational particles and thus the statistical noise is great. A particle weighting designed for such coagulation problems was proposed by Eibeck and Wagner [6], in which the statistical weight of a computational particle was a function of the physical particle it described and so there was no need to explicitly tag the computational particles with their statistical weights. The particular statistical weight used was the inverse of the physical particle mass, which has the useful consequence that each computational particle represents the same mass of physical particles per unit volume. In systems with constant total mass one would therefore expect the number of computational particles to remain constant in mean. By some judicious algebra the Mass Flow Algorithm (MFA) was derived so that the number of computational particles was constant in an absolute rather than just mean sense [6].

The MFA effectively redistributes computational particles over the size space, placing more computational particles at larger particle sizes and fewer at smaller sizes as compared with the constant weighting DSMC method. Consequently the high end of the particle size distribution is calculated more accurately at the expense of the low end calculations. A more general consideration of statistical weights as functions of the physical particle properties was undertaken by Kolodko and Sabelfeld [16], whose comparisons included the special case of the Mass Flow Algorithm (MFA) [6], but the greater generality came at the price of coagulation events that sometimes increased and sometimes decreased the number of computational particles.

Wells and Kraft [34] applied the MFA to a coagulation (and sintering) problem in nanoparticle dynamics. Consideration of systems of engineering interest inevitably led to a desire to simulate systems which exchange mass with their surroundings and thus a MFA cannot be expected to maintain a constant number of computational particles, even in mean, without continual resampling of the distribution. Nevertheless in [21, 22] MFA was successfully extended to systems with particle inception and where individual particle masses changed by interaction with the environment. The treatment of this last class of processes—surface growth—is set out in [22]. In essence, surface growth was simulated as two separate processes. In one, the mass of physical particles referred to by a computational particle is changed and in the other, new computational particles are introduced to the system to account for the increase in mass. Analogous processes which remove mass from the system would be treated by having the second process remove computational particles.

In [5], where the physical motivation was atmospheric aerosols, computational particles were assigned weights that were not simply functions of the physical particles they represented. Using operator splitting and deterministic integration, surface processes were treated by updating the statistical weight of a computational particle so that it represented the same number of physical particles before and after the surface events. Coupled with the MFA approach to coagulation this meant that the only change in the number of computational particles during simulations was due to processes of particle inception.

Maintaining a constant number of computational particles, or at least bounding their number below, is essential for controlling the variance of the Monte Carlo sample solutions, that is, it is essential for maintaining computational accuracy [18, 20, 31]. A simple DSMC algorithm for simulating a coagulation process does not have this property—the number of computational particles decreases at the same percentage rate as the number of physical particles—and thus simulation accuracy is quickly lost [16]. Direct approaches to solve this problem include resampling the particle population at every stop to maintain a constant number of computational particles [19, 31] and particle doubling [18, 20] with resampling to avoid overflow when processes that add particles are present. The work of Eibeck and Wagner [6], referred to above, took the more indirect and mathematical approach of formulating a new stochastic process that could solve the same coagulation problems as before but without the drop in computational particle numbers associated with basic direct simulation. Variance control was also addressed by Haibo et al. [12], who introduced statistical weights to arrive by a slightly more informal route at the 'w2' method (independently) derived in this paper. The same authors introduce a very similar method for the same purpose in [36]. Having firm bounds for the number of computational particles also enables simulation software to be made less complex and therefore more suitable for adaption to new problems.

The purpose of this paper is to describe two stochastic algorithms with explicitly weighted computational particles for physical particle coagulation and growth problems, principally soot formation in laminar premixed flames [3]. These algorithms are of interest because they provide ways of simulating coagulation and growth problems with constant numbers of computational particles, which is valuable for the reasons discussed above.

The remainder of this introduction will outline the structure of the problems to be solved and the general stochastic solution approach. The next part of the paper develops the explicit statistical weighting methods that enable coagulation and surface reactions to be simulated stochastically for the first time, without a change in the number of computational particles. In the final part of the paper two possible weighting schemes are compared to each other and a Direct Simulation Algorithm (DSA) with particle doubling.

1.1 Soot simulation

The population balance problems considered in this paper are taken from work on soot formation and growth although it is easy to see how other physical processes lead to equations with the same structure [17]. To put the algorithmic work in its context an overview of the soot simulation methodology will be useful. Consideration of soot formation is restricted to premixed laminar flames for this work (physically they present a promising basis for model development). In such a system a perfect mixture of carbon based fuel, oxidiser and an inert gas flow into a flame through a device similar to a school bunsen burner. Under suitable conditions a steady flame forms, which is approximately uniform across any horizontal planes, that is, the flame is a 1-dimensional system and a point can be adequately described by its height above the burner through which the gas enters the flame. Chemical species concentrations in the flame are found by solving a boundary value problem. The solution, which consists of species concentrations on a non-uniform grid, is stored in a text file which can be reused for every simulation of the flame with linear interpolation between the points.

The 1-dimensional structure of the laminar gas flow in the flame means that there is a one-to-one transformation between the residence time of a small volume of gas in the flame and the height of that volume above the burner through which it entered. The soot simulation is performed by calculating the time evolution of the soot population in such a closed, homogeneous volume as it moves through the flame. Initial results are therefore expressed as a function of time but are usually transformed back to be functions of height. Within such a volume, as it moves up the flame expanding and contracting the following random processes can occur (all particles are assumed to be spheres with density 1.8 g cm^{-3}):

1. Soot particles are incepted as spheres containing 32 carbon atoms at a rate I_t, which is a function of the precalculated chemical species concentrations.

2. Each soot particle of type x coagulates with particles of type y at rate $K(x, y)\, n(y)$, where $n(y)$ is the concentration of particles of type y. The form of K used is the 'transition regime kernel', see [24] for details.

3. Pyrene molecules (16 carbon atoms, hydrogen is ignored) condense on the surface of a soot particle of type x at rate $\beta_t^{(pyr)}(x)$, the new particle type is $g^{(pyr)}(x)$.

4. Acetylene molecules react with the surface of a soot particle of type x at rate $\beta_t^{(acet)}(x)$ causing the soot particle to grow by 2 carbon atoms, the new particle type is $g^{(acet)}(x)$.

5. Oxygen molecules react with the surface of a soot particle of type x at rate $\beta_t^{(oxy)}(x)$ causing the soot particle to lose 2 carbon atoms, the new particle type is $g^{(oxy)}(x)$.

6. OH radicals react with the surface of a soot particle of type x at rate $\beta_t^{(OH)}(x)$ causing the soot particle to lose 1 carbon atom, the new particle type is $g^{(OH)}(x)$.

This description is intended to explain the numerical problems studied in this paper. Readers interested in the soot model in its own right are advised to consult papers such as [2, 24, 30, 33] from where the information was originally taken.

1.2 Linear Process Deferment

For most flames surface reactions occur far more frequently than coagulation and inception events. This led to very large amounts of computer time being spent simulating them, an issue which was addressed in [25] by the introduction of the Linear Process Deferment Algorithm (LPDA); the work reported there was based on the DSA but is equally applicable to weighted particle methods. Before formulating the population balance equation with deferment let

1. the particle type space be E, addition on this space corresponds to particle coagulation,
2. $\mathcal{E}$ be an $\sigma-\mathrm{algebra}$ on E,
3. $\phi : E \to \mathbb{R}$ be drawn from an appropriate class of measurable test functions,
4. $\mathcal{U} = \{$acet, oxy, OH, pyr$\}$ be the set of surface processes,
5. $\mathcal{U}' = \{$acet, oxy, OH$\} \subset \mathcal{U}$ be the subset of deferred processes,
6. R_t be the deferred process operator. For any particle type x, $R_t x$ is distributed as the value at time t of the Markov chain defined by jump, rate pairs $\left\{\left(g^{(l)}, \beta_t^{(l)}\right) : l \in \mathcal{U}'\right\}$ that started from x at the time the particle was last updated. Computationally this is achieved by a time stamp on the particles, see [25] for details.

The (unweighted) weak form of the population balance equation to be solved is

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t} \int_{x \in E} \phi(x)\, \lambda_t(\mathrm{d}x) = & \int_{x \in E} \phi(x)\, I_t(\mathrm{d}x) \\ & + \sum_{l \in \mathcal{U} \setminus \mathcal{U}'} \int_{x, \xi \in E} \left[\phi\left(g^{(l)}(\xi)\right) - \phi(x)\right] \beta_t^{(l)}(\xi)\, \mathbb{P}(R_t x = d\xi)\, \lambda_t(\mathrm{d}x) \\ & + \frac{1}{2} \int_{x, y, \xi, \zeta \in E} \left[\phi(\xi + \zeta) - \phi(x) - \phi(y)\right] K_t(\xi, \zeta) \\ & \qquad \mathbb{P}(R_t x = \mathrm{d}\xi)\, \mathbb{P}(R_t y = \mathrm{d}\zeta)\, \lambda_t(\mathrm{d}x)\, \lambda_t(\mathrm{d}y)\end{aligned} \tag{1}$$

for λ_t, a time indexed family of measures over $[0, T]$ for some $T > 0$ and with $\lambda_0 = 0$. Solutions to this equation need any outstanding deferred events performing before they can be properly be related to the physical problem. The update operator $\mathcal{P}_t$ [25], which maps one measure on E to another is defined by

$$\mathcal{P}_t(\lambda)(A) = \int_{x \in E} \mathbb{P}(R_t x \in A)\, \lambda(\mathrm{d}x) \quad \forall A \in \mathcal{E} \tag{2}$$

so that if λ_t is a solution of (1) at time t then $\mathcal{P}_t(\lambda_t)$ is a solution to the undeferred population balance equation. That is, the mean concentration, at time t, of particles with types in A in an infinite, random physical system with the processes defined above.

1.3 Majorants

As it stands (1) and any weighted version of it would be difficult to simulate by a jump type Markov process because the jump process rates depend on functions of unknowns like ξ and ζ. A second complication is the infinite number of possible outcomes of a jump from x to $g^{(l)}(x)$. These issues are considered in [25] and the solution is to introduce majorants $\widehat{K}_t$ and $\widehat{\beta}_t$ such that

$$\widehat{K}_t(x,y) \geq K_t(R_t x, R_t y) \tag{3}$$

$$\widehat{\beta}_t(x) \geq \beta_t(x) \tag{4}$$

and to perform fictitious jumps when the inequalities are strict. In fact, (3) and (4) can rarely be satisfied with probability 1 but only a small error is introduced if they are satisfied with probability close to 1. This error may be controlled and monitored by breaking the calculation of λ_T into smaller steps and solving (1) for each step and then applying $\mathcal{P}_t$ to the solution before continuing.

2 Weighting

Let W be a set and $\Omega : W \to \mathbb{R}^+$ a map that defines how many physical particles an element of W indicates. In practice $W = \mathbb{R}^+$ and $\Omega = id$ would seem to be the natural choices. Let $\mathcal{W}$ be a $\sigma-\mathrm{algebra}$ on W which makes Ω measurable. Note that it is not possible to talk about statistical weights at this stage since there are no statistics—(1) is a deterministic equation. However, one can look for maps ν analogous to λ, but taking values that are measures on $W \times E$ and satisfying

$$\int_{W\times E} \Omega(w)\,\phi(x)\,\nu_t(\mathrm{d}w,\mathrm{d}x) = \int_E \phi(x)\,\lambda_t(\mathrm{d}x) \tag{5}$$

for all $t \in [0,T]$.

It is convenient to extend the R_t to act on $W \times E$ and to extend the I_t to be measures on the product $\sigma-\mathrm{algebra}$ on $W \times E$. The extensions, $\tilde{R}_t$ and $\tilde{I}_t$ must have the same physical interpretation as the original forms, so one requires

$$\int_E \phi(x)\,I_t(\mathrm{d}x) = \int_{W\times E} \phi(x)\,\Omega(u)\,\tilde{I}_t(\mathrm{d}u,\mathrm{d}x) \tag{6}$$

and

$$\Omega(u)\int_E \phi(\xi)\,\mathbb{P}(R_t x = \mathrm{d}\xi) = \int_{W\times E} \Omega(w)\,\phi(\xi)\,\mathbb{P}\left(\tilde{R}_t(u,x) = (\mathrm{d}w,\mathrm{d}\xi)\right) \qquad \forall (u,x) \in W \times E. \tag{7}$$

The simplest form for $\tilde{R}_t$ is

$$\tilde{R}_t(u,x) = (u, R_t x) \tag{8}$$

and this from will be used throughout this work. It is clearly the most natural extension—R represents the effects of processes that act on individual particles independently and therefore would not be expected to alter the number of physical particles being modelled.

Extensions of[1] β and K so that their final arguments can be measures on $W \times E$ rather than just E are also needed. If ν is a measure on $W \times E$ then let λ be the measure on E given by

$$\lambda(A) = \int_{W\times A} \Omega(u,x)\,\nu(\mathrm{d}u,\mathrm{d}x) \quad \forall A \in \mathcal{E} \tag{9}$$

and

$$\tilde{\beta}(x;\nu) := \beta(x;\lambda), \tag{10}$$

$$\tilde{K}(x,y;\nu) := K(x,y;\lambda). \tag{11}$$

A simple extension of I_t is also possible:

$$\tilde{I}_t(\mathrm{d}u,\mathrm{d}x) := \frac{\delta_w(\mathrm{d}u)}{\Omega(u)} I_t(\mathrm{d}x). \tag{12}$$

This approach will be used in this paper but it should be noted that u may vary with time and with x [29].

2.1 Dynamics of the New Measure

To proceed to a stochastic particle algorithm one requires the dynamics of ν from (5). Consider

$$\psi : W \times E \to \mathbb{R};\ (w,x) \mapsto \Omega(w)\,\phi(x) \tag{13}$$

so

$$\int_{W\times E} \psi(w,x)\,\nu_t(\mathrm{d}w,\mathrm{d}x) = \int_E \phi(x)\,\lambda_t(\mathrm{d}x) \quad \forall\, t \tag{14}$$

and in particular

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{W\times E} \psi(w,x)\,\nu_t(\mathrm{d}w,\mathrm{d}x) = \frac{\mathrm{d}}{\mathrm{d}t}\int_E \phi(x)\,\lambda_t(\mathrm{d}x). \tag{15}$$

By expanding the right hand side of (15) according to (1), repeatedly substituting (14) and assuming $\nu_t(\{(u,x) : \Omega(u) = 0\}) = 0\,\forall\, t$ one has

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{W\times E} \psi(u,x)\,\nu_t(\mathrm{d}u,\mathrm{d}x) = J_1 + J_2 + J_3 \tag{16}$$

[1]The labels $^{(l)}$ on β and g will be omitted from the following presentation to avoid the notation becoming too cluttered.

where

$$J_1 := \int_{(u,x)\in W\times E} \psi(u,x)\, \tilde{I}_t(\mathrm{d}u,\mathrm{d}x)\,, \tag{17}$$

$$J_2 := \int_{(W\times E)^2} \left[\frac{\psi(w,g(\xi))}{\Omega(w)} - \frac{\psi(u,x)}{\Omega(u)}\right] \beta_t(\xi;\nu_t)\,\Omega(u) \\ \mathbb{P}\left(\tilde{R}_t(u,x) = (\mathrm{d}u',\mathrm{d}\xi)\right)\nu_t(\mathrm{d}u,\mathrm{d}x) \tag{18}$$

and

$$J_3 := \frac{1}{2}\int_{(W\times E)^4} \left[\frac{\psi(w,\xi+\zeta)}{\Omega(w)} - \frac{\psi(u,x)}{\Omega(u)} - \frac{\psi(v,y)}{\Omega(v)}\right] \\ \tilde{K}(\xi,\zeta,\nu_t)\,\Omega(u)\,\Omega(v)\,\mathbb{P}\left(\tilde{R}_t(u,x) = (\mathrm{d}u',\mathrm{d}\xi)\right) \\ \mathbb{P}\left(\tilde{R}_t(v,y) = (\mathrm{d}v',\mathrm{d}\zeta)\right)\nu_t(\mathrm{d}u,\mathrm{d}x)\,\nu_t(\mathrm{d}v,\mathrm{d}y)\,. \tag{19}$$

In (18) & (19) the variables w appear as free parameters so (16) holds whatever values of w (provided $\Omega(w) > 0$) are substituted into J_2 and J_3. Therefore one may choose the rules for calculating the w from the u' and v' to optimise the stochastic particle algorithm that is being derived. To derive such an algorithm one wants to express all the integrands in (17)–(19) in the form $\sum_i \psi(u_i,x_i) - \sum_j \psi(v_j,y_j)$ multiplied by some rate expression and to integrate over all the (v_j,y_j). The differences of sums become the definitions of the jumps of the particle algorithm and the rest of the integral becomes the jump rate. The variables u' and v' found in (17)–(19) are dummy variables, which represent the weight component of $\tilde{R}_t(u,x)$ and $\tilde{R}_t(v,y)$ respectively.

As stated in (8) $u' = u$ and $v' = v$ with probability 1 so (18) can be simplified by choosing $w = u' = u$ to give

$$J_2 = \int_{E\times(W\times E)} [\psi(u,g(\xi)) - \psi(u,x)]\,\beta_t(\xi,\nu_t)\,\mathbb{P}(R_t x = \mathrm{d}\xi)\,\nu_t(\mathrm{d}u,\mathrm{d}x)\,. \tag{20}$$

There is more than one way to proceed from (19), two approaches that lead to different simulation procedures are given here. Both, by exploiting symmetry in (u,x) and (v,y), yield, for their respective definitions of w,

$$J_3 = \int_{E^2\times(W\times E)^2} [\psi(w,\xi+\zeta) - \psi(u,x)]\,\tilde{K}(\xi,\zeta,\nu_t)\,\Omega(v) \\ \mathbb{P}(R_t x = \mathrm{d}\xi)\,\mathbb{P}(R_t y = \mathrm{d}\zeta)\,\nu_t(\mathrm{d}u,\mathrm{d}x)\,\nu_t(\mathrm{d}v,\mathrm{d}y)\,. \tag{21}$$

The first approach is to choose w such that

$$\frac{1}{\Omega(w)} = \frac{1}{\Omega(u)} + \frac{1}{\Omega(v)}. \tag{22}$$

(This is a possible choice for at least the simplest choices of W and Ω—consider $W = \mathbb{R}^+$ with Ω a positive multiple of the identity then $w = \left(u^{-1} + v^{-1}\right)^{-1}$.) The second approach is to choose w such that

$$\Omega(w) = \frac{\Omega(u)}{2}, \tag{23}$$

which is also possible if $W = \mathbb{R}^+$ and Ω is a positive multiple of the identity. The idea of halving the statistical weight has previously been used by Haibo et al. [12] as mentioned in §1.

2.2 Simulation Algorithms

All the equations considered above can be thought of as deterministic, mean field equations [1]. Stochastic algorithms are derived (see, for example, [27, §4.6]) by taking (16) as defining the generator of a Markov process [13, ch 19], after introducing a scaling parameter to control the level of discretisation. The relationship of pure jump Markov processes to the mean field equations has been extensively studied in papers such as [4, 7–9, 11, 23] and their references. These and other papers typically prove that the trajectories of a sequence of pure jump Markov process converge in some sense to a solution of the deterministic mean field equation. The present paper was conceived as a more practical attempt to improve a computer program already in use for solving soot problems in chemistry and engineering. As such investigation of convergence is confined to numerical tests.

In this work the discretisation level or scaling may be controlled through $\tilde{I}_t$ as follows: Choose a sequence (w_N) in W such that $\Omega(w_N) > 0$ and $\Omega(w_N) \searrow 0$; define

$$\tilde{I}_t^N(\mathrm{d}u, \mathrm{d}x) := \frac{\delta_{w_N}(\mathrm{d}u)}{\Omega(w_N)} I_t(\mathrm{d}x). \tag{24}$$

Then replacing $\tilde{I}$ with $\tilde{I}_N$ and substituting (21)&(20) for J_1 and J_2 in (16) leads to a sequence of equations

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\int_{W\times E} \psi(u,x)\,\nu_t(\mathrm{d}u,\mathrm{d}x) &= \int_{(u,x)\in W\times E} \psi(u,x)\,\tilde{I}_t^N(\mathrm{d}u,\mathrm{d}x) \\
&\quad + \int_{(W\times E)^2} \left[\psi(u,g(\xi)) - \psi(u,x)\right] \beta_t(\xi,\nu_t) \\
&\qquad\qquad \mathbb{P}\left(\tilde{R}_t(u,x) = (\mathrm{d}u',\mathrm{d}\xi)\right) \nu_t(\mathrm{d}u,\mathrm{d}x) \\
&\quad + \int_{(W\times E)^4} \left[\psi(w,\xi+\zeta) - \psi(u,x)\right] \tilde{K}(\xi,\zeta,\nu_t)\,\Omega(v)\,\mathbb{P}\left(\tilde{R}_t(u,x) = (\mathrm{d}u',\mathrm{d}\xi)\right) \\
&\qquad\qquad \mathbb{P}\left(\tilde{R}_t(v,y) = (\mathrm{d}v',\mathrm{d}\zeta)\right) \nu_t(\mathrm{d}u,\mathrm{d}x)\,\nu_t(\mathrm{d}v,\mathrm{d}y).
\end{aligned} \tag{25}$$

A sequence of generators for pure jump Markov processes can then be derived from (25) in the standard way. The rates of the events which make up the Markov processes are given in table 1. In table 1, and for the rest of the paper, $W = \mathbb{R}^+$ and Ω is the identity mapping. Under reasonable conditions, the trajectories of these processes can be expected, as $N \to \infty$, to converge to solutions of the mean field problem. Throughout the remainder

Table 1: *Process summary*

$\longrightarrow (w_N, x)$	$\tilde{I}_t^N (\{(w_N, x)\})$
$(u, x) \longrightarrow (u, g(\xi))$	$\mathbb{P}\left(\tilde{R}_t (u, x) = \xi\right) \beta(\xi) \nu(\{(u, x)\})$
$(u, x) \longrightarrow (w, \xi + \zeta)$	$\tilde{K}(\xi, \zeta, \nu) \mathbb{P}(R_t x = \xi) \mathbb{P}(R_t y = \zeta)$ $\times \nu(\{(u, x)\}) \nu(\{(v, y)\})$

of the paper the two rules for calculating w for coagulation products will be denoted ‘w1’ and ‘w2’ as specified in table 2.

Table 2: *Coagulation weight rules*

w1	$w^{-1} = u^{-1} + v^{-1}$
w2	$w = u/2$

3 Numerical Tests

3.1 Initial Validation

Validation of the new weighted algorithms began using problems from [24], for which the entire particle size distribution can be calculated using an ODE solver. Direct solution of the population balance equations is possible because all but the first few thousand size classes can be neglected. Initial tests simulated only inception and coagulation processes, specifically the conditions were

1. physical particle inception rate $I_t = 2.63 \times 10^{13} \times \left(\frac{0.05-t}{0.05}\right)^2$ cm^{-3} s^{-1},
2. constant temperature of 500 K
3. constant pressure of 600 bar.

These are the same test conditions that were used in [24]. The particle size distributions are presented in figure 1 and show good agreement between the ODE solver results and all the stochastic weighted particle methods. Both rules for the weight of the coagulation

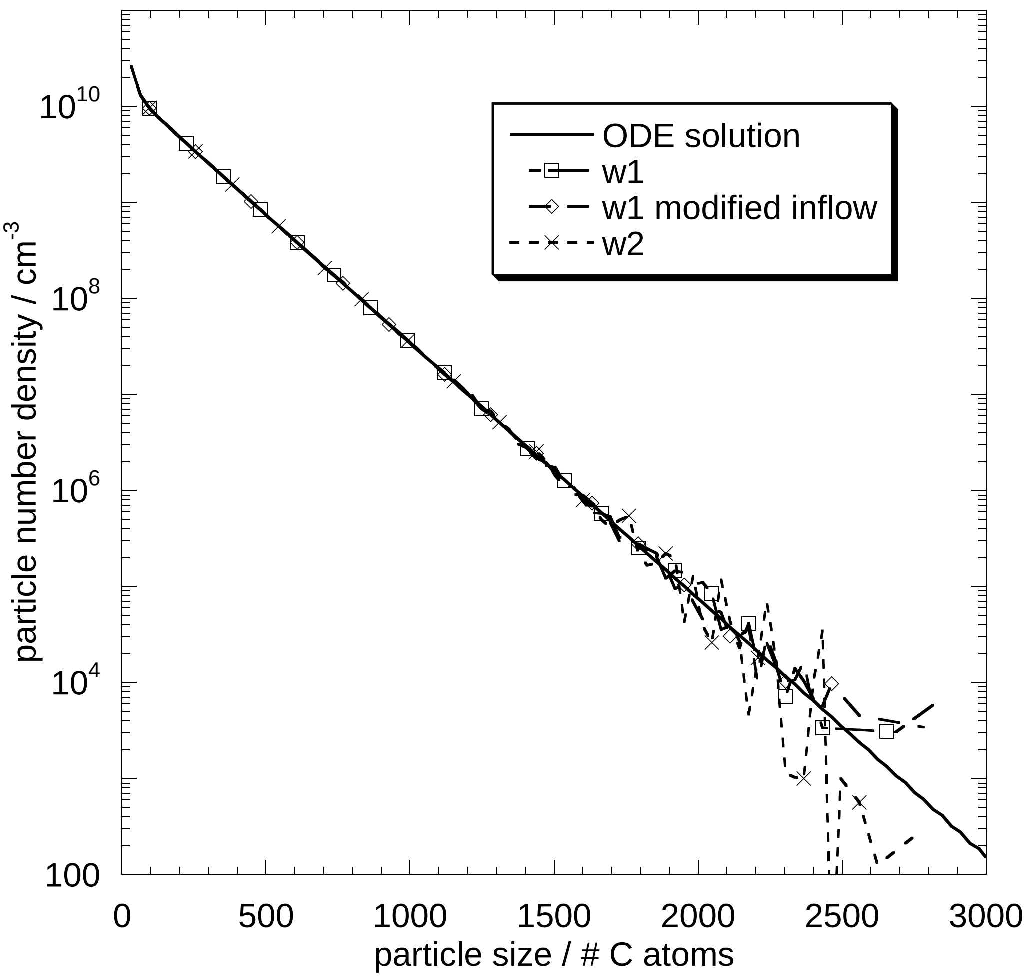

Figure 1: *PSD for coagulation and inception test problem*

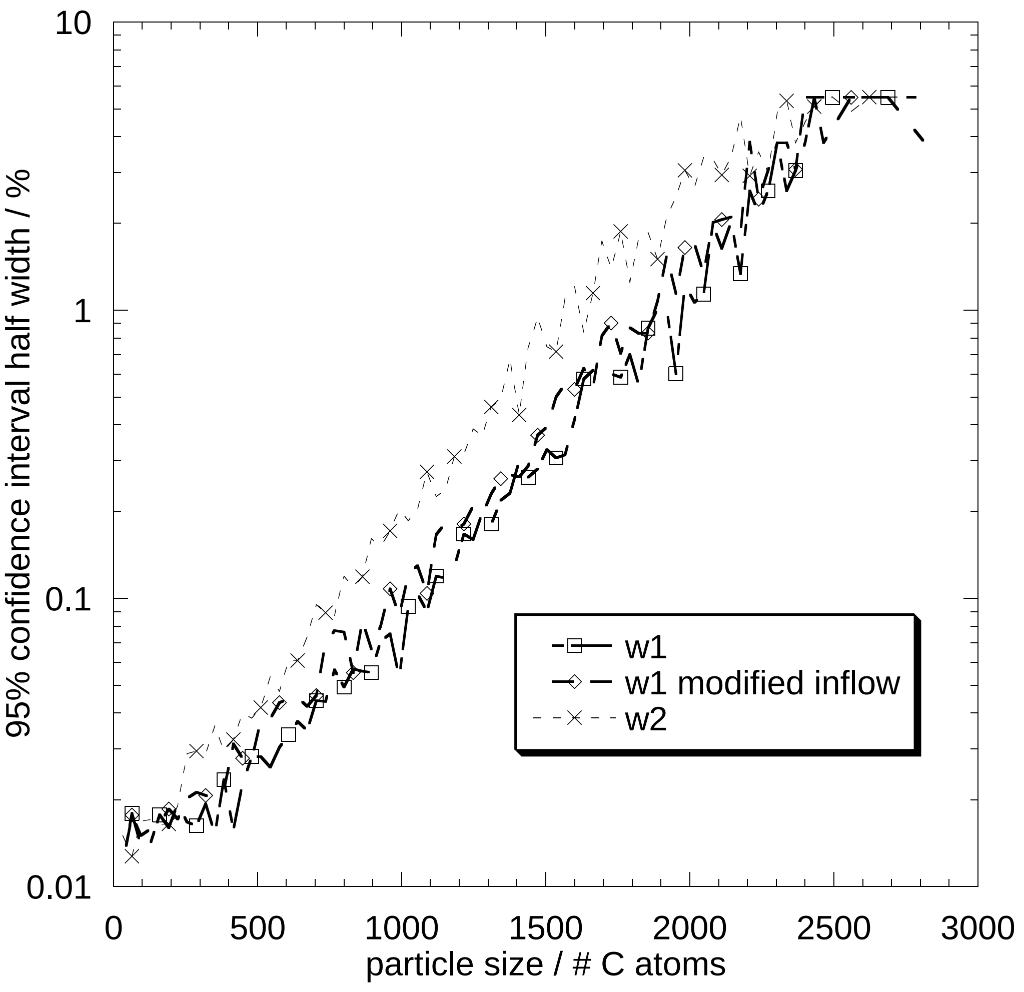

Figure 2: *Statistical noise for coagulation and inception test problem*

product (see §2.2, table 2) were used with computational particle inflow proportional to the inception rate for physical particles, that is, with w from (12) constant. These two data sets are labelled 'w1' and 'w2' in the legend of figure 1. In the nucleation and coagulation only case considered here, w1 corresponds to the mass flow algorithm of [7]. The w1 coagulation rule was also tested with a modified inflow rule; in this case the rate of inflow of computational particles was constant throughout the simulation, but the statistical weight of the new computational particles was automatically adjusted to simulate the physical particle inception rate. Adjustment of the statistical weight of particles entering a system has previously been used in simulations of the Boltzmann equation [29].

The statistical noise associated with the methods was also considered. Figure 2 summarises the results. The 95% confidence interval sizes were calculated from a central limit theorem estimate based on 30 realisations of the Markov chain for each simulation method, each realisation used just under 2^{16} computational particles. The results in figure 2 indicate that w2 is generally more noisy than w1 and that there is little difference between the two inflow methods used with w1. The data is an initial indication that w1 is to be preferred to w2 since fewer realisations of the w1 Markov chain than of the w2 Markov chain would be needed to obtain the same size of confidence interval.

Further comparisons to the particle size distribution produced by the ODE solver were performed for a test problem including a surface reaction (pyrene condensation). Good agreement was found between the weighted particle methods and the deterministic solution of the size distribution for the limited case for which this was possible.

3.2 LPDA and real flames

Having established that the algorithms and their implementations worked correctly, in limited cases for which the population balance equations could be solved directly, testing moved on to premixed laminar flames. For these tests the LPDA as described in §1.2 was used throughout, including for the DSA calculation used to provide a reference solution. The first flame based test compared the accuracy of the moments of the soot particle size distribution calculated with the w1 and w2 weightings for the flame JW10.68 [15]. The second moment of the mass distribution is shown in figure 3, 95% confidence intervals for the DSA and w1 data are within $\pm 2\%$ of the plotted values and so confidence intervals are only shown for w2. The calculations for the weighted algorithms were performed with 30

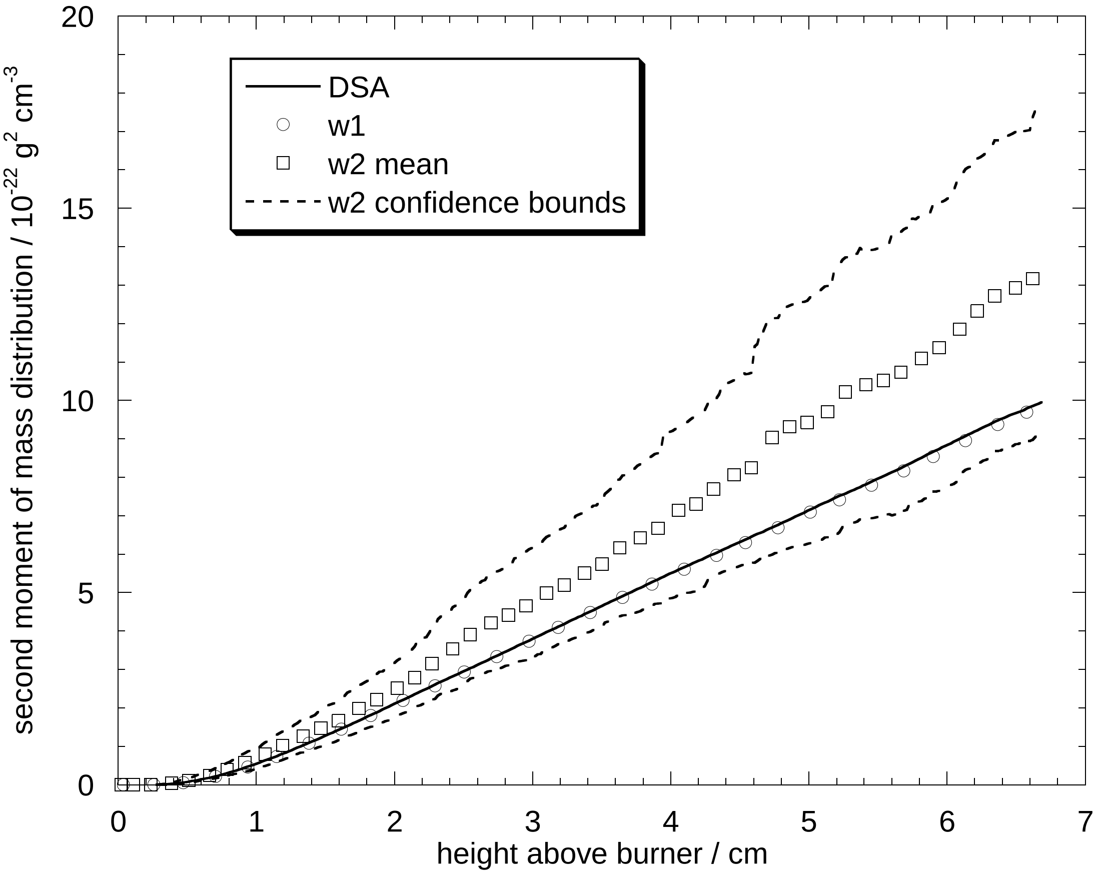

Figure 3: *Second moment of JW10.68 mass distribution*

runs using around 2000 computational particles from the end of the inception peak until the end of each simulation. To ensure no error was introduced by the deferral of processes all computational particles were updated every time the simulation covered 5×10^{-4} s of real time. The large difference in the statistical variability generated by the two weighting methods should be observed. The w2 method leads to a variance for the second moment of the size distribution that is more than 10 times larger than that obtained with the w1 method. The situation with the zeroth and first moments is similar.

The real attraction of stochastic particle methods is that they provide an explicit estimate of the particle distribution. As a test case for the distribution the flame JW1.69 [15] was used. This flame is known to have a bimodal particle size distribution [3] and therefore to present an interesting test case for the way in which the w1 weighting method transfers computational effort to larger particle sizes. The w2 method was not used for this comparison, because the results above suggest that far more realisations of the w2 Markov chain would be required than of the w1 Markov chain in order to achieve the same precision. Therefore, to meet any particular error tolerance, less computer time would be required using the w1 method.

Densities were calculated using the statistical computation package R [26] by performing Gaussian blurring of the observations with a bandwidth of 0.0245. The densities presented here were calculated in logarithmic size space, that is, they are (estimates of)

$$\frac{\mathrm{d}}{\mathrm{d}\log_{10} x} N\left(\log_{10} x\right) \tag{26}$$

where $N\left(\log_{10} x\right)$ is the number of particles per cubic centimetre comprising no more than x carbon atoms. Data calculated from 50 repetitions of the w1 Markov chain with just under 2^{13} computational particles are compared to data from high precision DSA (without LPDA) calculations which used 30 repetitions with between 2^{16} and 2^{16} computational particles.The results in figure 4 show a very high degree of agreement between the two algorithms, these particular data apply to the top of the flame—about 4.2 cm above the burner.

In figure 4 large oscillations in the density generated from the w1 data can be seen for particle sizes between 300 and 10,000 carbon atoms. These oscillations are a symptom of the way the weighted algorithm transfers computational resolution to large particle sizes as discussed in the next paragraph and illustrated in figure 5. The weighted method clearly would be rather unsuitable for this flame if the number of particles containing 300–10,000 carbon atoms was the main quantity of interest. In such a case a DSA method with particle doubling [18, 20] as a variance reduction technique should be used if possible. However, as will be seen for other measures of solution accuracy, the weighted method can offer as good or better performance than the un-weighted alternative.

It is also interesting to look at the distribution of computational particles on the size spectrum since the number of particles is what controls the precision of the calculation. In figure 5 the normalised densities of the computational particle distribution for the calculations used for figure 4 are plotted. The normalisation ensures that the area under both the w1 and the DSA curve is 1 (when integrated against $\mathrm{d}\left(\log_{10} x\right)$) and so there are no effects due to the different numbers of computational particles used with the two algorithms. Figure 5 gives a very clear view of the way in which w1 and DSA concentrate

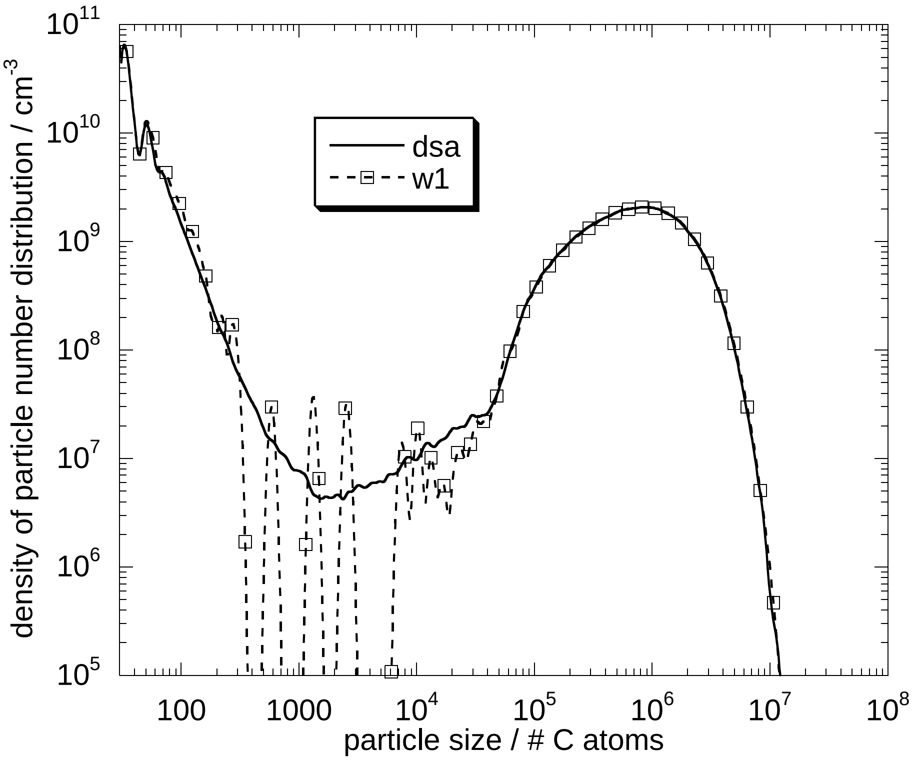

Figure 4: *Physical particle size distribution for JW1.69*

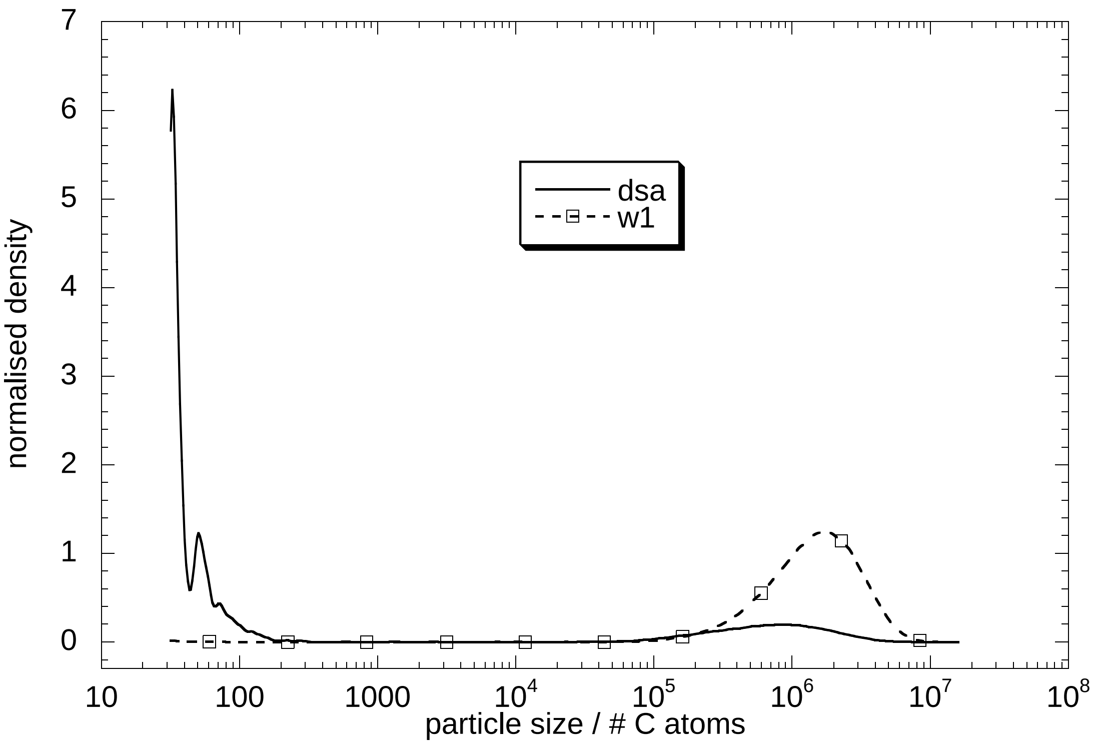

Figure 5: *Relative computational particle distribution for JW1.69*

computational effort on different parts of the size range. This shows that the choice of algorithm will depend, to some extent, on the problem that is being solved. Problems that mainly concern the largest particles are likely to be addressed best using a weighted algorithm, problems concerning the smallest particles should be tackled with a DSA. The remainder of this paper attempts to investigate this choice in a quantitative way.

3.3 Performance

Simulations of the flame HWA3 [35] were performed using DSA and the w1 weighting. The first set of tests reported ignored the acetylene, OH and O_2 surface reactions since simulation of these processes takes a significant amount of time and is the same whether or not weighted particles are used. The results obtained in this way give no information about the soot produced by the flame but provide a comparison that should focus a little more on the properties of the weighted algorithms. Simulation size is described by the maximum number of computational particles in the simulation, settings were chosen so that most of this capacity was used. The initial sample volume (DSA) and the weighting in I_t were chosen to use almost all the capacity of the binary tree, in which the computational particles were stored, at the point when the physical particle number peaked. For DSA particle doubling [25] was used so that the tree was never less than 50% full. For the weighted methods doubling was not needed as the number of computational particles did not decrease, this being one of the attractions of the weighted method, see (21).

Table 3 summarises performance on the simplified flame; run times were measured on the same desktop PC which has a 2 GHz Athlon XP CPU (2400+). The memory requirement of the simulations are low, only a few MB are required, even for the largest simulations. In table 3 the standard deviation of the population of samples for certain functionals of

Table 3: *Variability of algorithms for reduced HWA3*

tree	method	time per	population std. dev. /%				
size		run / s	m0	m1	m2	m3	5–6$\times 10^3$
2^{12}	w1	0.6	9.7	6.4	15.0	25.4	11.1
2^{14}	w1	2.5	4.4	3.6	7.7	12.0	6.5
2^{16}	w1	10.4	3.0	1.8	3.9	6.0	2.9
2^{12}	DSA	0.5	5.4	4.2	14.6	34.9	24.4
2^{14}	DSA	1.9	2.7	2.3	7.6	19.5	11.7
2^{16}	DSA	7.6	1.5	1.1	4.0	9.6	7.4

the solution are given for 1.34 cm above the burner, which is approximately the end of the flame. The functionals used are the zeroth, first, second, third moments of the mass distribution (denoted m0, m1, m2, m3 respectively) and the number of particles containing 5000–6000 carbon atoms.

From table 3 one sees that, for a given tree size, DSA simulations take around 75% of the computational time. For the zeroth moment DSA yields an estimate that is only half as variable as the w1 approach. However, the advantage of DSA drops as one moves to higher moments and by the third moment DSA is significantly inferior to w1. The value of the higher moments is primarily determined by the larger particles in the distribution and it is not surprising the w1 offers an advantage in this case since it increases the computational resolution for this part of the distribution. The variance in the estimates of the number of particles containing 5000–6000 carbon atoms is an even more extreme example of the way in which w1 resolves the larger sizes better (at the expense of the smaller ones) compared with DSA.

For all the functionals both algorithms appeared to have converged in mean to the true value for the largest tree sizes reported in table 3. This was verified by performing larger simulations on other hardware, which were not timed, and so are not reported in detail. These results suggest that, for some functionals, which heavily emphasize the distribution of larger particles w1 offers a faster way of getting good estimates than DSA.

3.4 Further Comparison

The same tests were carried out, with the same flame, HWA3, but including all reactions on the surfaces of soot particles by means of the LPDA (compare §1.2). A couple of results for w2 are included for interest but fit the pattern discussed above and will not receive any further attention. Stochastic simulation of this flame is of considerable interest because measured particle size distributions have been published in [35]. Computational times are not comparable to those from table 3 because different hardware was used. The simulations with the full flame model take much longer because of the high rates of surface reactions and so Opteron 252 processors running at 2.6 GHz in 64 bit mode were used for the computations. The results are summarised in table 4.

Table 4: *Variability of algorithms for physical system*

tree size	method	time per run / s	population std. dev. /%				
			m0	m1	m2	m3	9–10$\times 10^5$
2^{12}	w1	4.4	9.6	1.9	3.8	6.2	18.9
2^{14}	w1	18.0	5.2	1.1	2.3	3.6	9.8
2^{16}	w1	73.7	2.4	0.6	1.3	2.1	4.9
2^{18}	w1	274	1.1	0.2	0.4	0.7	2.1
2^{14}	w2	18.9	5.3	3.3	2.3	7.5	36.1
2^{16}	w2	75.1	2.2	1.5	3.4	5.7	16.8
2^{12}	DSA	3.0	4.6	1.1	2.9	7.1	39.1
2^{14}	DSA	12.2	3.0	0.5	1.4	3.3	19.6
2^{16}	DSA	49.5	1.3	0.3	0.6	1.8	9.3
2^{18}	DSA	213	0.7	0.1	0.4	1.0	5.3

In common with the results for the simplified problem DSA is seen to be around one third faster than the w1 approach for a given tree size. All the sets of simulations produced reasonably accurate estimates of the quantities considered in table 4: For the moments the mean from 80 repetitions with each tree size was within 1% of the values from extremely high precision calculations. For the number of particles containing 9–10$\times 10^5$ C atoms, the difference between the mean and the high precision solution just reached 4% in some cases, which is not statistically significant. As in the simplified case DSA generates less statistically noisy estimates for the first few moments of the size distribution but w1 becomes more attractive for functionals that place a greater stress on the largest sizes of particles. However, for the reduced case, w1 was significantly less noisy for the third moment (m3) than DSA for a given tree size, but for the full flame the crossover is only just beginning at m3. It can be seen that for the number of particles in the size range 9–10$\times 10^5$ the w1 algorithm produces an estimate with roughly the same variance as the DSA with 4 times the number of particles. Examination of the 'time per run' column of table 4 shows that w1 can therefore provide an estimate, of any given precision, of the number of particles in the size range 9–10$\times 10^5$ in roughly one third of the time of DSA.

3.5 Potential Applications

While the weighted particle algorithm presents an alternative to the DSA with particle doubling for simulations of Smoluchowski's coagulation equation for spatially homogeneous systems, it also has other potential applications where it might clearly distinguish itself from the DSA. One application which has already received considerable attention for the purposes of gas dynamics simulation is the simulation of particle populations in a grid of cells. For such problems that ability to explicitly control the weighting is important to capture effects in regions with low particle densities [14], when a small proportion of the population has very important effects [29]. Weighted particle methods would therefore seem attractive for simulations of spatially resolved coagulating systems, a purpose for which the MFA has already been used [10].

Explicit weights offer options for the implementation of particle transport between cells by accounting automatically for differences in the statistical weight assigned to computational particles in different cells and facilitating conservative resampling of particle populations [32]. It is also possible to exploit weighting to adjust computational resolution independently of the main kinetic simulation process by resampling the computational particle population. An application of resampling would be to construct a different computational resolution profile from the one shown in figure 5 in order to move the statistical noise seen at particle sizes in the range 300–10,000 carbon atoms in figure 4 to a different size range.

4 Conclusion

A general weighted form of the Smoluchowski coagulation equation with additional linear terms has been formulated. From this equation two new weighted particle simulation algorithms have been derived. Implementations of these algorithms have been successfully validated against direct solutions of the population balance equations of simple systems. Further results for complex laminar premixed flame systems show close agreement with data from calculations with more established stochastic methods. The statistical noise generated by the two weighted algorithms has been compared showing one weighted algorithm to be consistently better than the other. Finally, run times to achieve fixed error tolerances for a real flame system have been measured on a desktop PC and the better weighted algorithm has been found to be up to three times faster than direct simulation.

4.1 Acknowledgements

The authors are grateful to Wolfgang Wagner for his advice and helpful criticism.

RP would like to acknowledge support from the EPSRC under EP-C547241-1 and from Trinity College, Cambridge.

References

[1] D. J. Aldous. Deterministic and stochastic models for coalescence (aggregation and coagulation) : a review of the mean-field theory for probabilists. *Bernoulli*, 5, 1999.

[2] J. Appel, H. Bockhorn, and M. Frenklach. Kinetic modeling of soot formation with detailed chemistry and physics: Laminar premixed flames of C_2 hydrocarbons. *Combust. Flame*, 121:122–136, 2000.

[3] M. Balthasar and M. Kraft. A stochastic approach to solve the particle size distribution function of soot particles in laminar premixed flames. *Combust. Flame*, 133: 289–298, 2003.

[4] M. Deaconu, N. Fournier, and E. Tanré. Rate of Convergence of a Stochastic Particle System for the Smoluchowski Coagulation Equation. *Meth. Comput. App. Prob.*, 5 (2):131–158, 2003.

[5] E. Debry, B. Sportisse, and B. Jourdain. A stochastic approach for the numerical simulation of the general dynamics equation for aerosols. *J. Comput. Phys.*, 184: 649–669, 2003.

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

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

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

[9] A. Eibeck and W. Wagner. Stochastic interacting particle systems and nonlinear kinetic equations. *Ann. Appl. Probab.*, 13(3):845–889, 2003.

[10] F. Guiaş. A stochastic numerical method for diffusion equations and applications to spatially inhomogeneous coagulation processes. In H. Niederreiter and D. Talay, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2004*, pages 147–161. Springer, 2006. doi:10.1007/3-540-31186-6_10.

[11] F. Guiaş. A Monte Carlo approach to the Smoluchowski equations. *MCMA*, 3(4): 313–326, 1997.

[12] Z. Haibo, Z. Chuguang, and X. Minghou. Multi-Monte Carlo approach for general dynamic equation considering simultaneous particle coagulation and breakage. *Powder Tech.*, 154:164–178, 2005. doi:10.1016/j.powtec.2005.04.042.

[13] O. Kallenberg. *Foundations of modern probability*. Springer, New York, 2 edition, 2001.

[14] K. C. Kannenberg and I. D. Boyd. Strategies for efficient particle resolution in the direct simulation Monte Carlo method. *J. Comput. Phys.*, 157:727–745, 2000. doi:10.1006/jcph.1999.6397.

[15] A. Kazakov, H. Wang, and M. Frenklach. Detailed modeling of soot formation in laminar premixed ethylene flames at a pressure of 10 bar. *Combust. Flame*, 100: 111–120, 1995.

[16] A. Kolodko and K. Sabelfeld. Stochastic particle methods for Smoluchowski coagulation equation: variance reduction and error estimations. *MCMA*, 9(4):315–339, 2003.

[17] M. Kraft. Modelling of particulate processes. *KONA, Powder and Particle*, (23): 18–35, 2005.

[18] K. Liffman. A direct simulation Monte-Carlo method for cluster coagulation. *J. Comput. Phys.*, 100(1):116–127, 1992.

[19] Y. Lin, K. Lee, and T. Matsoukas. Solution of the population balance equation using constant-number Monte Carlo. *Chem. Eng. Sci.*, 57(12):2241–2252, 2002.

[20] A. Maisels, F. E. Kruis, and H. Fissan. Direct simulation Monte Carlo for simultaneous nucleation, coagulation and surface growth in dispersed systems. *Chem. Eng. Sci*, 59:2231–2239, 2004.

[21] N. Morgan, C. Wells, M. Kraft, and W. Wagner. Modelling nanoparticle dynamics: coagulation, sintering, particle inception and surface growth. *Combust. Theor. Model.*, 9(3):449–461, 2005.

[22] N. Morgan, C. Wells, M. Goodson, M. Kraft, and W. Wagner. A new numerical approach for the simulation of the growth of inorganic nanoparticles. *J. Comput. Phys.*, 211(2):638–658, 2006.

[23] J. R. Norris. Smouchowski's coagulation equation: uniqueness, nonuniqueness and a hydrodynamic limit for the stochastic coalescent. *Ann. Appl. Probab.*, 9(1):78–109, 1999.

[24] R. Patterson, J. Singh, M. Balthasar, M. Kraft, and W. Wagner. Extending stochastic soot simulation to higher pressures. *Combust. Flame*, 145(3):638–642, 2006.

[25] R. I. A. Patterson, J. Singh, M. Balthasar, M. Kraft, and J. R. Norris. The linear process deferment algorithm: A new technique for solving population balance equations. *SIAM J. Sci. Comput.*, 145(1):303–320, 2006.

[26] R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2005. URL `http://www.R-project.org`. ISBN 3-900051-07-0.

[27] D. Ramkrishna. *Population Balances*. Academic Press, 2000.

[28] S. Rjasanow and W. Wagner. A stochastic weighted particle method for the Boltzmann equation. *J. Comput. Phys.*, 124:243–253, 1996.

[29] S. Rjasanow and W. Wagner. Simulation of rare events by the stochastic weighted particle method for the Boltzmann equation. *Math. and Comput. Modelling*, 33: 926–907, 2001.

[30] J. Singh, R. Patterson, M. Balthasar, M. Kraft, and W. Wagner. Modelling soot particle size distribution: Dynamics of pressure regimes. Technical Report 25, c4e Preprint-Series, Cambridge, 2004.

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

[32] A. Vikhansky and M. Kraft. Conservative method for the reduction of the number of particles in the Monte Carlo simulation method for kinetic equations. *J. Comput. Phys.*, 203:371–378, 2005. doi:10.1016/j.jcp.2004.09.007.

[33] H. Wang and M. Frenklach. A detailed kinetic modeling study of aromatics formation in laminar premixed acetylene and ethylene flames. *Combust. Flame*, 110: 173–221, 1997.

[34] C. G. Wells and M. Kraft. Direct simulation and mass flow stochastic algorithms to solve a sintering-coagulation equation. *MCMA*, 11:175–197, 2005. doi:10.1163/156939605777585980.

[35] B. Zhao, Z. Yang, Z. Li, M. V. Johnston, and H. Wang. Particle size distribution function of incipient soot in laminar premixed ethylene flames: effect of flame temperature. *Proc. Combust. Inst.*, 30:1441–1448, 2005.

[36] H. Zhao, C. Zheng, and M. Xu. Multi-Monte Carlo approach for particle coagulation: description and validation. *Appl. Math. Comput.*, 167:1383–1399, 2005. doi:10.1016/j.amc.2004.08.014. This paper is by the same authors as [12]; the two published manuscripts just reverse the orders of the parts of the authors' names.