Chapter 1
Numerical Methods for Coupled Population Balance Systems Applied to the Dynamical Simulation of Crystallization Processes

Robin Ahrens, Zahra Lakdawala, Andreas Voigt, Viktoria Wiedmeyer, Volker John, Sabine Le Borne, Kai Sundmacher

Abstract Uni- and bi-variate crystallization processes are considered that are modeled with population balance systems (PBSs). Experimental results for uni-variate processes in a helically coiled flow tube crystallizer are presented. A survey on numerical methods for the simulation of uni-variate PBSs is provided with the emphasis on a coupled stochastic-deterministic method. In

Robin Ahrens
Hamburg University of Technology, Faculty of Electrical Engineering, Informatics and Mathematics, Institute of Mathematics, Am Schwarzenberg-Campus 3, 21073 Hamburg, Germany, e-mail: robin.ahrens@tuhh.de

Zahra Lakdawala
Weierstrass Institute for Applied Analysis and Stochastics (WIAS), Mohrenstr. 39, 10117 Berlin, Germany, e-mail: lakdawala@wias-berlin.de

Andreas Voigt,
Otto-von-Guericke-University Magdeburg, Department Process Systems Engineering, Universitätsplatz 2, 39106 Magdeburg, Germany, e-mail: andreas.voigt@ovgu.de

Viktoria Wiedmeyer
ETH Zurich, Institute of Process Engineering, Sonneggstrasse 3, 8092 Zurich, Switzerland, e-mail: wiedmeyv@ipe.mavt.ethz.ch

Volker John
Weierstrass Institute for Applied Analysis and Stochastics (WIAS), Mohrenstr. 39, 10117 Berlin, Germany, and Freie Universität Berlin, Department of Mathematics and Computer Science, Arnimallee 6, 14195 Berlin, Germany, e-mail: john@wias-berlin.de

Sabine Le Borne
Hamburg University of Technology, Faculty of Electrical Engineering, Informatics and Mathematics, Institute of Mathematics, Am Schwarzenberg-Campus 3, 21073 Hamburg, Germany, e-mail: leborne@tuhh.de

Kai Sundmacher
Otto-von-Guericke-University Magdeburg, Department Process Systems Engineering, Universitätsplatz 2, 39106 Magdeburg, Germany, and Max Planck Institute for Dynamics of Complex Technical Systems, Process Systems Engineering, Sandtorstr. 1, 39106 Magdeburg, Germany, e-mail: sundmacher@mpi-magdeburg.mpg.de

this method, the equations of the PBS from computational fluid dynamics are solved deterministically and the population balance equation is solved with a stochastic algorithm. With this method, simulations of a crystallization process in a fluidized bed crystallizer are performed that identify appropriate values for two parameters of the model such that considerably improved results are obtained than reported so far in the literature. For bi-variate processes, the identification of agglomeration kernels from experimental data is briefly discussed. Even for multi-variate processes, an efficient algorithm for evaluating the agglomeration term is presented that is based on the fast Fourier transform (FFT). The complexity of this algorithm is discussed as well as the number of moments that can be conserved.

1.1 Introduction: Modeling of Crystallization Processes with Population Balance Systems

Solid state processing is an important part of the industrially relevant production as about 70% of products of the chemical and pharmaceutical industry are sold as solids. An important part of this processing is crystallization of solid materials from liquid solutions. Fundamental and applied research in this area of crystallization will lead to improved process performance with less energy consumption as well as more efficient material utilization. Also the product quality and specifications like size and its distribution, shape, and agglomeration degree have to be considered in more detail, as many process steps are dependent on such characteristics [44]. The DFG priority programme 1679 "Dynamic simulation of inter-connected solid processes" addressed many of the current issues and our particular contribution has been the investigation of different important aspects of continuous crystallization processes. As solid-liquid systems are complex and challenging in many ways and fluid flow and particles interact in a variety of fashions, the numerical methods had to be extended and new tools had to developed to simulate crystallization in a better way. We focus here on relevant phenomena of crystal growth of multi-faceted crystals as well as on crystal agglomeration with two specifically developed model experiments working with selected well-understood model substances.

Crystallization processes are often modeled in terms of a crystal population instead of considering the behavior of each individual crystal. Utilizing macroscopic conservation laws, one derives a system of coupled equations for the population, a so-called population balance system (PBS), that describes an averaged behavior of the crystals.

We consider crystallization processes within a moving incompressible fluid, which occur, e.g., in pipes or batch crystallizers. It is assumed that the suspension of the crystals is dilute such that the impact of the crystals on the fluid flow is negligible. Then, the first two conservation laws are the balance

of the linear momentum and the conservation of mass for the fluid flow, which are modeled by the incompressible Navier–Stokes equations

$$\begin{aligned} \partial_t \boldsymbol{u} - \nabla \cdot \left(\frac{\rho}{\eta} \nabla \boldsymbol{u} \right) + (\boldsymbol{u} \cdot \nabla)\, \boldsymbol{u} + \nabla p &= \boldsymbol{f} \text{ in } (0, t_{\text{end}}) \times \Omega, \\ \nabla \cdot \boldsymbol{u} &= 0 \text{ in } (0, t_{\text{end}}) \times \Omega. \end{aligned} \tag{1.1}$$

In (1.1), t_{end} [s] is a final time, $\Omega \subset \mathbb{R}^3$ is a bounded domain, which is assumed to be constant in the whole time interval, $\boldsymbol{u}$ [m/s] is the velocity field, p [Pa] is the pressure, $\boldsymbol{f}$ [m/s²] represents forces acting on the fluid, ρ [kg/m³] is the density of the fluid, and μ [kg/m s] is the dynamic viscosity of the fluid. Often, the body forces possess the form $\boldsymbol{f} = (0, 0, g)^T$ with g [m/s²] being the gravitational acceleration.

The other equations of a PBS are usually coupled. These are equations for the energy balance, where the unknown quantity is the temperature T [K], for the balance of the molar concentration c [mol/m³] of dissolved species, and for the balance of the particle population density f [1/kg m³] (the unit is for a particle population density with the only internal coordinate mass, it is different in other situations).

The energy balance of the PBS has the form

$$\partial_t T - D_T \Delta T + \boldsymbol{u} \cdot \nabla T = F_{\text{ener,growth}}(c, T, f) \quad \text{in } (0, t_{\text{end}}) \times \Omega, \tag{1.2}$$

where D_T [m²/s] is a diffusion coefficient, $\boldsymbol{u}$ is the velocity from (1.1), and the right-hand side $F_{\text{ener,growth}}(c, T, f)$ [K/s] models the energy consumption or production in the growth process of the crystals. Since the velocity is divergence-free, it holds that $\boldsymbol{u} \cdot \nabla T = \nabla \cdot (T\boldsymbol{u})$.

In a crystallization process, the dissolved material in the fluid is used in the growth process of the crystals. The corresponding balance equation has the form

$$\partial_t c - D_c \Delta c + \boldsymbol{u} \cdot \nabla c = F_{\text{conc,growth}}(c, T, f) \quad \text{in } (0, t_{\text{end}}) \times \Omega. \tag{1.3}$$

Here, D_c [m²/s] is again a diffusion coefficient and $F_{\text{conc,growth}}(c, T, f)$ [mol/s m³] represents the consumption or production of dissolved material. We like to mention that there are PBSs with a coupled system of equations of type (1.3) for several concentrations, like in the modeling of precipitation processes, e.g., see [36].

The final part of a PBS is an equation for the particle population density. Assuming that the number of internal or property coordinates is $d_{\text{int}} \geq 1$, then this equation might read as follows

$$\begin{aligned} &\partial_t f + (\boldsymbol{u} + \boldsymbol{u}_{\text{sed}}) \cdot \nabla f + \nabla_{\text{int}} \cdot (G(c, T) f) \\ &= F_{\text{agg}}(\boldsymbol{u}, c, T, f) + F_{\text{break}}(\boldsymbol{u}, c, T, f) \quad \text{in } (0, t_{\text{end}}) \times \Omega \times \Omega_{\text{int}}. \end{aligned} \tag{1.4}$$

Here, Ω_{int} is the d_{int}-dimensional domain for the internal coordinate and $\boldsymbol{u}_{\text{sed}}$ [m/s] is the sedimentation velocity, which is assumed to be divergence-free. The growth term is assumed to be linear with the growth rate $G(c,T)$ [kg/s], and ∇_{int} is the nabla operator with respect to the internal coordinates. Nucleation is included via appropriate boundary conditions with respect to the internal coordinates. The right-hand side of (1.4) describes the agglomeration (aggregation, coalescence) of crystals and their breakage (fragmentation).

To simplify the presentation below, the case $d_{\text{int}} = 1$ will be considered in this section, i.e., a so-called univariate population. Then, Ω_{int} is just an interval, e.g., an interval with respect to the mass of the crystals $\Omega_{\text{int}} = [m_{\min}, m_{\max}]$ in kg and it is $\nabla_{\text{int}} = \partial_m$. In this case, the agglomeration term for every time-space point $(t, \boldsymbol{x})$ has the form

$$\begin{aligned} F_{\text{agg}}(\boldsymbol{u}, T, f) = \frac{1}{2} \int_{m_{\min}}^{m_{\max}} \kappa_{\text{agg}}(\boldsymbol{u}, T, m-m', m') f(m-m') f(m') \, dm' \\ - \int_{m_{\min}}^{m_{\max}} \kappa_{\text{agg}}(\boldsymbol{u}, T, m-m', m') f(m) f(m') \, dm', \end{aligned} \tag{1.5}$$

where κ_{agg} [m^3/s] is the agglomeration kernel. The first term, which is the source term, models the amount of crystals of mass m that are created by the agglomeration of two crystals with masses m' and $m - m'$, where $m' \in (m_{\min}, m_{\max})$. The corresponding sink term accounts for the crystals of mass m that vanish because they are consumed by agglomeration with other crystals of mass m'. The breakage term might be of the form

$$\begin{aligned} F_{\text{break}}(\boldsymbol{u}, c, T, f) = \int_{m_{\min}}^{m_{\max}-m} \kappa_{\text{break}}(\boldsymbol{u}, T, m, m') f(m+m') \, dm' \\ -\frac{1}{2} \int_{m_{\min}}^{m} \kappa_{\text{break}}(\boldsymbol{u}, T, m-m', m') f(m') \, dm', \end{aligned} \tag{1.6}$$

where κ_{break} [1/kg s] is the breakage kernel. The first term on the right-hand side describes the appearance of crystals of mass m and the second term describes the disappearance of such crystals due to breakage events.

1.2 Uni-variate Process

1.2.1 Benchmark Problem

Different phenomena such as nucleation, growth, breakage, and agglomeration occur during crystallization. It depends on the particular crystallization process, which phenomena are dominant. They have to be identified and integrated in the population balance system (PBS) as shown in (1.4), while other

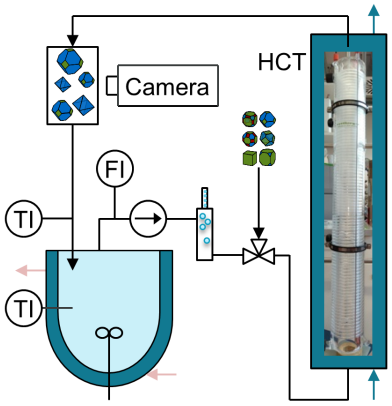

Fig. 1.1 Schematic of the benchmark experiment in the helically coiled flow tube (HCT) crystallizer.

terms may be neglected. The resulting coupled PBS needs to be parameterized. For that, benchmark problems are required. Here, a growth dominated crystallizer is selected.

As mentioned in the previous section, the crystal mass can be used as internal property coordinate of the PBS. The goal of the presented benchmark problem is to intensify a process to grow faceted crystals shape-selectively. Hence, a measure of crystal size is applied as internal coordinate. To determine the crystal size distribution (CSD), 3d-crystal shapes are estimated from 2d-projections of the observed particles following the methods by [9, 10, 11]. The shape is described by the perpendicular distances of the crystal faces to the crystal center. It is sufficient to consider one perpendicular distance for each face type to describe the full symmetry of an ideal crystal. Potassium aluminum sulfate dodecahydrate, also called potash alum, crystallizes predominantly as octahedron in aqueous solution. Hence, its shape can be characterized by one face type. The resulting crystal distribution is univariate.

The benchmark problem is of high dimension. There are four dimensions in time and space and one internal coordinate. Further, the solid and liquid phase are coupled.

1.2.2 Helically Coiled Flow Tube Crystallizer

1.2.2.1 Setup and Process

Growth-dominated experiments are realized in a helically coiled flow tube (HCT) crystallizer. The crystallization is temperature controlled. For the

experiments, solution is pumped from a reservoir to the HCT, as depicted in Figure 1.1. The solution passes a degasifier before crystal seeds are added, where the seeds are of a defined size fraction. The suspension is cooled in the HCT to grow. At the outlet of the HCT, the crystal population is imaged by a flow-through microscope. Finally, the crystals are dissolved in a reservoir.

Seeds are sieved in different size fractions. The seed fractions are applied for residence time experiments without growth and for growth experiments. In the experiments, several process parameters can be varied systematically: helix orientation, average fluid flow rate, crystal seed fraction, feed concentration, and temperature [66, 67]. Selected results are shown for an HCT crystallizer with a coil diameter of 0.11 m and an inner tube diameter of 0.006 m at laminar flow rates.

1.2.2.2 Residence Time Distribution

In residence time experiments for the dispersed phase, a sieved crystal size fraction was added within 10 s at the inlet. The solution was saturated and isothermal conditions were applied to avoid crystal growth. The residence time was estimated from the crystal projections, which were recorded at the tube outlet by the flow-through microscope. Further, the crystal shape and a size descriptor were estimated from the projections. Crystal velocities were calculated from the measured residence times and known geometry of the HCT and are depicted in Figure 1.2. They were measured in an HCT crystallizer made of glass (length of 35 m, upward flow). Mean crystal velocities were calculated for several size classes. It was observed that large crystals of about 200 µm size are faster than smaller crystals of a size of about 100 µm. This observation holds for particles of a density which deviates from the fluid density at laminar flow rates in HCTs [66, 67]. In the PBS, the residence time can be empirically described in dependence of the crystal size by a polynomial function or by interpolation from measurements. To apply the model in a size range that exceeds the measured sizes, it can be assumed that very small crystals follow the fluid flow, as shown in Figure 1.2.

The crystal residence time depends on the process parameters. Crystallization experiments in HCTs show that crystals of different size have different velocities in HCTs. Large crystals are faster than small crystals. Size-dependent residence times can be used to separate crystals of certain sizes in batch or periodic operation.

1.2.2.3 Crystal Growth

Crystals can be grown in HCTs by cooling crystallization. The longer the tubes and the lower the fluid velocities, the more time crystals have to grow and the larger the attainable final crystal sizes. This is illustrated for the

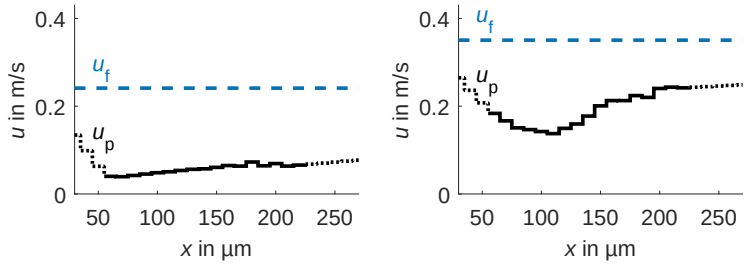

Fig. 1.2 Crystal-size dependent crystal velocities at two different laminar average fluid velocities (blue, dashed) for the univariate potash alum. Measured in experiments (black, solid) and extrapolated (black, dotted).

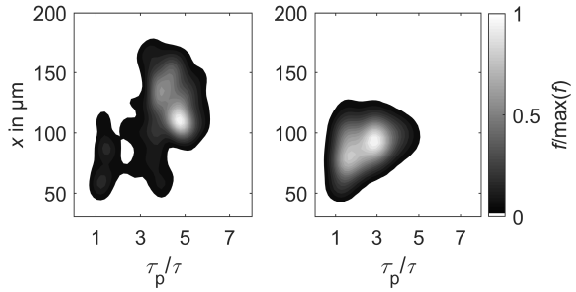

Fig. 1.3 Product crystal number density distributions after crystal growth experiments for varying average fluid flow rate: left: u =0.24 m/s; right: u =0.35 m/s. Potash alum seed fraction of a size x of (95 ± 11) µm at a feed saturation temperature of 40 °C and an initial outlet supersaturation of σ =4 %.

case of varying fluid velocity in Figure 1.3. Crystal growth can be realized continuously in HCTs to change the CSD.

Numerically, the solution of the full model of the form (1.1)–(1.6) is expensive due to the mutual coupling of the equations. Hence, the model is reduced and assumptions are made for a dynamic simulation with reasonable computation times:

a) It is assumed that the energy balance (1.2) can be neglected when a temperature profile is given.
b) The momentum balance (1.1) is neglected.
c) Only one spatial coordinate is considered, which is the z-coordinate along the tube axis.
d) A low suspension density and moderate cooling are applied experimentally to suppress nucleation, breakage, and agglomeration.
e) Crystal growth is size-independent.

The reduced population balance equation (PBE) is

$$\partial_t f + u \cdot \nabla f + G(c,T)\nabla_{\text{int}} \cdot f = 0 \quad \text{in } (0, t_{\text{end}}) \times \Omega \times \Omega_{\text{int}}. \tag{1.7}$$

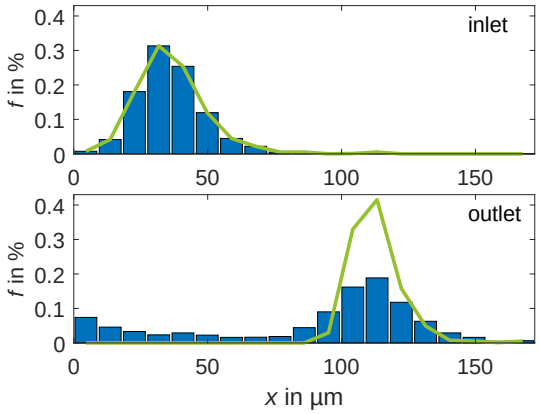

Fig. 1.4 Crystal growth experiments (blue bars) and simulation (green solid curve) of a potash alum seed fraction at a feed saturation temperature of 40 °C and an initial supersaturation of σ =17 % at the outlet for an average fluid flow rate u =0.24 m/s. Crystal number density distributions: top: seed crystals; bottom: product.

For the continuous phase, there are two balance equations, since potash alum crystallizes as dodecahydrate under consumption of water from the solution. The diffusion term in (1.3) is replaced by a dispersion term of the same structure, but of a different value for the coefficient D_c. The crystal growth rate depends on the supersaturation of the continuous phase and thereby on the local temperature $T(t, z)$. The local temperature can be set by external cooling and it can vary dynamically.

The reduced PBS consisting of (1.3) and (1.7) was discretized in space z and in the internal size coordinate x via finite volume method. The derived differential algebraic equation system was solved with the MATLAB-ODE23 solver, which is based on a Runge-Kutta approach. Product CSDs resulting after crystal growth are depicted in Figure 1.5. As expected, the final crystal size increases with tube length during cooling crystallization. In batch simulations, the size-dependent residence time leads to narrow crystal size distributions compared to a uniform particle residence time.

1.2.3 Brief Survey on Numerical Methods for Solving a PBS

Let the time interval be decomposed into subintervals $[t_{n-1}, t_n]$, $n = 1, \dots N$, with $0 = t_0 < t_1 < \dots < t_N = t_{\text{end}}$ and let the (numerical) solution $\boldsymbol{u}_{n-1}, T_{n-1}, c_{n-1}, f_{n-1}$ at the time instance t_{n-1} be given. Then, one has to apply some time stepping scheme to compute the (numerical) solution at

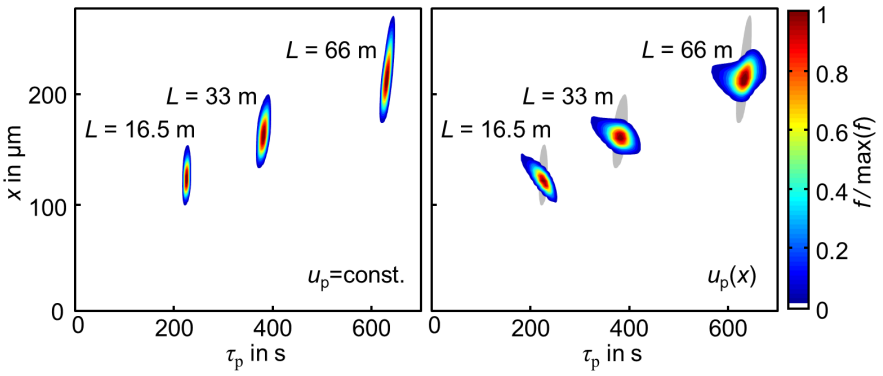

Fig. 1.5 Simulated product number density distribution based on a reduced model for the crystal growth of a normally distributed potash alum seed fraction (μ =106 µm, σ =41 µm) for different tube lengths. Left: a constant particle velocity; right: a size-dependent crystal velocity based on the measurement data.

t_n. This section provides a brief survey on methods that are proposed in the literature for computing a numerical solution at t_n.

Since a monolithic approach for solving the PBS, which computes all unknown functions together from (1.1)–(1.4), is computationally too demanding, the PBS is split into several parts and these parts are solved consecutively.

1.2.3.1 The Navier–Stokes Equations

Because the velocity appears in all equations and there is no back coupling of the other unknowns to the flow field, it is a straightforward idea to solve first the Navier–Stokes equations (1.1). These equations can be solved monolithically or decoupled by a so-called projection scheme. As temporal discretization, often first or second order time stepping schemes are used, like the Euler schemes, the Crank–Nicolson scheme, or the backward difference formula of order 2 (BDF2). The nonlinear term in the momentum balance can be treated implicitly, semi-implicitly, or explicitly. The semi-implicit approach is called implicit-explicit (IMEX) scheme. Usual spatial discretizations include finite element methods (FEM), finite volume methods (FVM), or, for simple domains, finite difference methods (FDM). A detailed description of all these approaches is far beyond the scope of this paper. Many of them are described, within the framework of FEMs, in [35, Chapter 7].

The situation becomes more complicated if the flow is turbulent. There is no mathematical definition of turbulence, but a good physical description is that a turbulent flow contains a wide range of physically important scales. In particular, there are many small scales that cannot be resolved on affordable

grids and, consequently, that cannot be simulated. Standard discretizations cannot cope with this situation since they try to simulate all important scales. Simulations with such discretizations usually blow up in finite time. Since neglecting the small scales leads to physically incorrect numerical simulations, an approach is needed to model the impact of the unresolvable scales onto the resolvable scales. This approach is called turbulence modeling. In the literature, many turbulence models are proposed, e.g., see [54, 58], and turbulence modeling is still an active field of research. There is no turbulence model that can be considered to be the best one.

At the end of this step, $\boldsymbol{u}_n$ is known and it can be used in the other equations of the PBS.

1.2.3.2 The Energy and Concentration Equations

As a next, natural step, the equations (1.2) for the energy balance and (1.3) for the concentration balance can be solved. Again, due to the numerical complexity, a monolithic solution of this system of equations does not seem to be attractive. Instead, the equations are solved individually, by using the currently available data, e.g.,

$$\begin{aligned} &1.) \quad \partial_t T_n - D_T \Delta T_n + \boldsymbol{u}_n \cdot \nabla T_n = F_{\text{ener,growth}}\left(c_{n-1}, T_{n-1}, f_{n-1}\right), \\ &2.) \quad \partial_t c_n - D_c \Delta c_n + \boldsymbol{u}_n \cdot \nabla c_n = F_{\text{conc,growth}}\left(c_{n-1}, T_n, f_{n-1}\right), \end{aligned}$$

where still the temporal derivatives have to be discretized. In this approach, one has to solve two linear equations. The individual solution of these equations can be iterated by using in the second iteration the temperature and concentration solution computed in the first iteration and so on.

In many applications, in particular in crystallization processes, the diffusion parameters in (1.2) and (1.3) are smaller by several orders of magnitude compared with the size of the velocity field. This situation is called convection-dominated and there is a similar difficulty as for turbulent flows: there are important features of the solution, so-called layers, that cannot be resolved on affordable grids. As for turbulent flows, standard numerical discretizations fail in this situation and the use of a so-called stabilized discretization is necessary, e.g., see [55]. There are many proposals for stabilized discretizations in the literature. In the context of the coupled system (1.2) and (1.3), it is essential that the numerical solution computed with the stabilized method must not possess unphysical values, so-called spurious oscillations, or it is allowed to exhibit only negligible spurious oscillations. This property is important because the computed solutions serve as data in other equations, for certain coefficients, and if the numerical solutions have spurious oscillations, then non-physical coefficients in other equations might be computed. At any rate, it was noted in [36] for a precipitation process that using a stabilized discretization that does not sufficiently suppress spurious oscillations

usually leads to a blow-up of the simulations of the coupled system in finite time.

As a matter of fact, many of the proposed stabilized schemes lead to numerical solutions with non-negligible spurious oscillations, e.g., see the numerical assessment in [39]. Some schemes that satisfy the requirement with respect to the spurious oscillations are the followings:

- finite difference methods
  - upwind; very diffusive and very inaccurate,
  - FCT (flux-corrected transport) schemes [14];
  - ENO (essentially non-oscillatory) [32], WENO (weighted ENO) [45]; much more accurate, small spurious oscillations possible,
- finite element methods
  - linear FEM-FCT [42]; often good compromise between accuracy and efficiency,
  - FEM-FCT [46, 43]; nonlinear method, often quite accurate,
- finite volume methods
  - Scharfetter–Gummel method [59]; improved upwind but still quite diffusive,
  - FCT [70].

The assessments provided above are based mostly on our experience from [37].

1.2.3.3 The Population Balance Equation

After having discretized the temporal derivative in (1.4), one obtains an equation for f_n in a four- or even higher-dimensional domain. But this difficulty is not the only one for solving the population balance equation. There is a transport operator on the left-hand side of (1.4) whose discretization requires special techniques, and on the right-hand side there are integral operators whose efficient evaluation is complicated, in particular for the first term of the agglomeration (1.5).

First of all, there are several principal ways for designing a scheme for computing a numerical approximation of f_n:

- solve an equation in the high-dimensional domain $\Omega \times \Omega_{\text{int}}$, where the left-hand side is discretized with some appropriate discretization based on FDM, FEM, or FVM, the so-called direct discretization,
- apply an operator-splitting scheme that deals first with an equation in Ω and after this with an equation in Ω_{int},
- utilize a momentum-based method to transform the population balance equation to a system of equations in a three-dimensional domain,

- apply a stochastic method for solving (1.4).

The first approaches will be discussed briefly in the following whereas the last approach is presented in detail in Section 1.2.4.

Utilizing the first approach, the direct discretization, is computationally demanding. One issue is that usual CFD codes do not support four- or higher-dimensional domains. Using an implicit approach, then the system matrix becomes comparatively dense, compared with 3d, and the question of an appropriate solver for the linear systems of equations arises. The left-hand side of (1.4) is a transport operator, which can be considered as a limit case with vanishing diffusion of the convection-dominated operators from the energy and concentration balances. The discretizations mentioned for the convection-dominated operators in Section 1.2.3.2 can be applied also for the transport operator of the population balance equation. In addition one needs a numerical method for evaluating the integral terms on the right-hand side of (1.4), see Section 1.3.2 for a discussion of this topic. Direct discretizations of 4d population balance equations can be found, e.g., in [12, 13, 60], and of a 5d population balance equation in [40].

Operator-splitting schemes for population balance equations in the form mentioned above were proposed in [25], see also [26]. Motivations for this proposal are efficiency, the possibility to use software that is designed for domains in usual dimensions, and the possibility to apply different discretizations for the different equations. The principal form of the equations to be solved is as follows. Let $\hat{f}_n = f_{n-1}$, solve in the first step

$$\partial_t \hat{f} + (\boldsymbol{u}_n + \boldsymbol{u}_{\mathrm{sed},n-1}) \cdot \nabla \hat{f} = 0 \quad \text{in } (t_{n-1}, t_n) \times \Omega \tag{1.8}$$

for all $y \in \Omega_{\mathrm{int}}$. Then, set $\tilde{f}_{n-1} = \hat{f}_n$, solve

$$\begin{aligned} &\partial_t \tilde{f} + \partial_m \left(G(c_n, T_n) \tilde{f} \right) \\ &= F_{\mathrm{agg}}(\boldsymbol{u}_n, c_n, T_n, \hat{f}_n) + F_{\mathrm{break}}(\boldsymbol{u}_n, c_n, T_n, \hat{f}_n) \quad \text{in } (t_{n-1}, t_n) \times \Omega_{\mathrm{int}} \end{aligned} \tag{1.9}$$

for all $\boldsymbol{x} \in \Omega$, and set $f_n = \tilde{f}_n$. There are several modifications of this basic operator-splitting scheme for population balance equations, in particular to perform the steps in a different order, e.g., see [3, 25, 27]. Equation (1.8) is usually a transport equation with dominating convection, such that one has to utilize a stabilized discretization, see Section 1.2.3.2. Also (1.9) is a transport equation, but the growth of the crystals might be sufficiently slow such that one can apply some standard discretization. The operator splitting introduces an additional splitting error which does not spoil the optimal order of convergence for low order finite element methods [25].

As already mentioned at the beginning of this section, the definition of Equation (1.4) for the crystal size distribution in a higher-dimensional domain is a major challenge for the simulation of population balance systems. A popular way to avoid this issue is the consideration of the first moments of

the crystal size distribution, as proposed the first time in [33], where the so-called Method of Moments (MOM) was derived. The kth moment of the crystal size distribution is given by

$$M_k = \int_0^\infty m^k f \, dm, \quad k = 0, 1, 2, \ldots. \tag{1.10}$$

It will be assumed, that f is zero for $m \leq m_{\min}$ and $m \geq m_{\max}$, i.e., that there are a minimal and a maximal mass for the crystals. Hence, the domain of integration in (1.10) can be restricted to this interval. On the one hand, the first moments are often important in practice because they correspond to physical quantities, like the number of crystals (0th moment) or the mass of the crystals (3rd moment). But on the other hand, the reconstruction of the crystal size distribution from its moments is a severely ill-posed problem and it is hard to design stable algorithms [34].

Multiplying (1.4) with m^k, integrating with respect to the internal coordinate, commuting this integration with differentiation in time and with respect to the external variable yields an equation for the kth moment

$$\partial_t M_k + (\boldsymbol{u} + \boldsymbol{u}_{\text{sed}}) \cdot \nabla M_k = \int_{m_{\min}}^{m_{\max}} m^k S \, d\ell, \quad k = 0, 1, 2, \ldots, \tag{1.11}$$

with

$$S = F_{\text{agg}}(\boldsymbol{u}, c, T, f) + F_{\text{break}}(\boldsymbol{u}, c, T, f) - \partial_m \left(G(c, T) f \right).$$

System (1.11) is a closed system for a finite number of moments only in special cases, e.g., if there are no agglomeration, no breakage, and special growth functions.

For the case that a closure of (1.11) cannot be found, we consider for simplicity only the growth term on the right-hand side of (1.11). Applying integration by parts and using that f vanishes at $m_{\min}$ and $m_{\max}$, this term can be reformulated as follows

$$\begin{aligned} -\int_{m_{\min}}^{m_{\max}} m^k \partial_m \left(G(c, T) f \right) \, dm &= \int_{m_{\min}}^{m_{\max}} k m^{k-1} G(c, T) f \, dm \\ &= \int_{m_{\min}}^{m_{\max}} \tilde{G}(c, T) f \, dm, \quad k \geq 1, \end{aligned}$$

with the new growth function $\tilde{G}(c, T) = k m^{k-1} G(c, T)$. Note that this integral still contains the unknown crystal size distribution f. The principal idea of the Quadrature Method of Moments (QMOM) proposed in [49] consists in approximating this integral by some quadrature formula

$$\int_{m_{\min}}^{m_{\max}} \tilde{G}(c, T) f \, dm \approx \sum_{i=1}^{N} \omega_i \tilde{G}(m_i), \tag{1.12}$$

where N is the number of quadrature points, which is prescribed by the user, ω_i are the weights of the quadrature rule and m_i are the nodes (quadrature points, abscissas). Then, at time instance t_n, one considers the system of equations for the moments

$$\partial_t M_k + (\boldsymbol{u} + \boldsymbol{u}_{\text{sed}}) \cdot \nabla M_k = \int_{m_{\min}}^{m_{\max}} \tilde{G}(c_n, T_n) f_{n-1} \, dm, \quad k = 0, \ldots, 2N-1, \tag{1.13}$$

where the left-hand side has still to be discretized appropriately and the right-hand side is approximated with (1.12).

To keep the quadrature error in (1.12) as small as possible, the weights and abscissas should be chosen such that the optimal order $(2N-1)$ of the numerical quadrature is obtained. Several algorithms are available for this purpose. In [41], it is shown that the long quotient-modified difference algorithm (LQMDA) behaves better than two other algorithms concerning stability and efficiency. For computing the optimal weights and abscissas, the knowledge of f is not necessary, but only of the first $2N$ moments of f. Thus, for the first time step $n = 1$, one can use the known initial condition of f for computing the right-hand side in (1.13) such that the first $2N$ moments at time t_1 can be computed. Then, these moments can be used for computing the right-hand side for the next time instance and so on.

Agglomeration and breakage processes can be also incorporated into the framework of QMOM, e.g., see [48]. An extension of the QMOM, which does not compute the moments, but directly the weights and abscissas, is the Direct Quadrature Method of Moments (DQMOM), as proposed in [47]. It is also possible to simulate multivariate populations with QMOM, e.g., see [18].

1.2.3.4 On Our Experience with Some of the Methods

As already mentioned above, it was noted in [36] that the use of a stabilized scheme for convection-diffusion equations, which does not suppress spurious oscillations, sufficiently often leads to a blow up of the simulations. Only cutting off such oscillations appropriately led to stable simulations. However, such cut-off techniques lead inevitably to violations of conservation properties. Moreover, in the same paper it was concluded that the use of upwind techniques led to completely smeared and practically useless results. A clear improvement of the quality of the numerical solutions was observed in [38] by using a linear FEM-FCT scheme for the convection-diffusion and transport equations in the PBS. Based on this experience, we have employed the linear FEM-FCT scheme for solving the energy equation (1.2) and concentration equation (1.3) in PBSs. Different numerical methods for the 4d population balance equation were studied in [13]. The problem of interest was a turbulent air-droplet flow in a segment of a wind tunnel, where $\Omega \times \Omega_{\text{int}}$ was

a tensor product domain in 4d. In this situation, FDM approaches can be applied easily. Two kinds of linear FEM-FCT schemes and an FDM ENO scheme were compared. It turned out that the FDM ENO scheme was by far the most efficient approach, such that it was recommended for population balance equations on tensor product domains. This scheme was also applied successfully for the simulation of a bivariate population balance in [40]. In [3], a direct discretization using the FDM ENO approach for the population balance equation (1.4) and an operator-splitting scheme were compared for an axisymmetric problem. While the operator-splitting scheme converged faster to a steady-state, the evolution of the transition was predicted more accurately by the direct discretization.

In summary, up to the publication of [3], we could, on the one hand, identify accurate and efficient approaches for simulating PBSs that are given on tensor product domains. Here, efficiency refers only to the differential operators in the population balance equation (1.4). Efficient methods for the integral operators are a different topic, which will be discussed in Section 1.3.2. But on the other hand, it is very complicated to extend our favorite approach, the direct discretization, to problems defined in more general domains, which occur usually in applications. In this respect, we could make decisive progress in the preceding years by employing and further developing a stochastic method, which will be discussed in detail in Section 1.2.4.

1.2.4 A Stochastic Method for Simulating the Crystal Size Distribution

This section describes a stochastic particle simulation (SPS) method for computing a numerical approximation of the crystal size distribution f whose behavior is modeled by the population balance equation (1.4). This method can be applied successfully for the simulation of problems given in complex spatial domains.

The basis of the SPS method that is utilized in our simulations is the method proposed in [53, 52]. This method had to be extended by all features that are caused from the movement of the crystals in the spatial domain: convective transport in three dimensions, sedimentation, crystal-wall collisions, and the coupling with the deterministic methods for solving the other equations (1.1)–(1.3) of the PBS. The algorithms from [52, 53] include convective transport in one dimension, growth, and coagulation (collision growth). With respect to the first and third feature, the method is based on two classical algorithms. The first one is Bird's direct simulation Monte-Carlo algorithm for the Boltzmann equation [8] that proposes an approach to handle the convective transport part with a splitting method. The second algorithm is the Gillespie algorithm [28, 29] that models the coagulation via stochastic jump

processes. One of the original contributions from [53] is a stochastic algorithm for simulating crystal growth via a surface reaction model.

Altogether, the splitting scheme applied in the SPS method consists of two parts: the convective transport of crystals, discussed in Section 1.2.4.1, and Markov jump processes for simulating growth, agglomeration, and insertion of crystals, described in Section 1.2.4.2. Section 1.2.4.3 presents the complete algorithm that simulates the PBSs (1.1)–(1.4).

1.2.4.1 Convective Transport of Stochastic Computational Crystals

The spatial domain Ω is triangulated by a triangulation consisting of mesh cells $K_j, j \in \{1, ..., N\}$. Each mesh cells contains a crystal ensemble $\mathcal{E}_j$. In the stochastic method, computational crystals (particles) are considered that represent an ensemble of physical crystals (particles). For simplifying the notion, the computational crystals will be called just 'crystals' in the following.

Consider a spatial mesh cell K and a crystal ensemble $(K, \mathcal{E})$, where each crystal e_i in $\mathcal{E}$ possesses a spatial and an internal coordinate $e_i = (\boldsymbol{x}_i, m_i)$, with $\boldsymbol{x}_i \in K$ and $m_i \in \Omega_{\text{int}}$. The complete ensemble $\mathcal{E}$ with $N_{\mathcal{E}}$ crystals is given by $\mathcal{E} = (e_1, ..., e_{N_{\mathcal{E}}})$.

Let Δt be a constant splitting time. First of all, the flow field $\boldsymbol{u}$ from the Navier–Stokes equations (1.1) is responsible for the transport of crystals. Second, crystals are also moved by sedimentation with the sedimentation velocity $\boldsymbol{u}_{\text{sed}}$. In the convection step, each crystal e_i is transported along the trajectories of $\boldsymbol{u} + \boldsymbol{u}_{\text{sed}}$

$$\boldsymbol{x}_i \longrightarrow \boldsymbol{x}_i + \Delta t \left(\boldsymbol{u}(\boldsymbol{x}_i) + \boldsymbol{u}_{\text{sed}}(\boldsymbol{x}_i) \right). \tag{1.14}$$

There are two topics that will be discussed in this section. From the modeling point of view, a model for the sedimentation velocity $\boldsymbol{u}_{\text{sed}}$ is needed. From the algorithmic point of view, one has to detect whether the crystal left its mesh cell after the transport step or even would hit the boundary of the domain if the transport step is performed and appropriate numerical procedures have to be performed in these situations.

For the considered application, a crystallization process in a fluidized bed crystallizer, the sedimentation of crystals has to be taken into account. Sedimentation depends on various aspects, like the form of the crystals and the actual local velocity field. In our application, the crystals can attain quite different forms. Since we could not find an appropriate sedimentation model in the literature, we decided to use as basis a sedimentation model for spherical particles, see [7, pp. 58] for its derivation. However, numerical studies in [6] showed that we had to modify this model for our purposes. Concretely, a scaling factor was introduced. Finally, the sedimentation velocity in our numerical simulations has the form

$$\boldsymbol{u}_{\text{sed}} = (0, 0, u_z)^T \quad \text{with} \quad u_z = \sigma \frac{\left(\frac{6}{\rho\pi}\right)^{\frac{2}{3}} (\rho_{\text{cryst}} - \rho) g}{18\mu} m^{\frac{2}{3}}. \tag{1.15}$$

In this model, ρ [kg/m³] is the density of the fluid, ρ_{cryst} [kg/m³] the density of the crystals, μ [kg/m·s] the dynamical viscosity of the fluid, $g = 9.81$ m/s² the gravity, and σ the numerically determined scaling factor. It can be observed that the sedimentation model (1.15) depends on the mass of the crystals. In [6], a brief numerical study led to the choice $\sigma = 0.1$ in (1.15). Section 1.2.5 will present results that are obtained also with a different scaling factor.

After having performed the transport step (1.14), it must be checked whether each moved crystal still belongs to the same mesh cell. If not, then it must be removed from its current ensemble. If the final point of the relocation is within Ω, it is inserted in the ensemble of the new cell. However, it might happen that this point is outside Ω such that the crystal hits the boundary of the flow domain. The treatment of this situation required a notable extension of the algorithm for the crystal transport.

First of all, for the considered application, we distinguished the boundary part through which the crystal would leave the domain. Crystals that would leave through the inflow boundary, which is located at the bottom of the fluidized bed crystallizer, are measured and removed from the simulations. This situation happens because of the sedimentation of crystals. Crystals that would leave through other boundaries are reflected and repositioned in the domain. Two reflection algorithms were implemented, which both model elastic wall collisions where no kinetic energy is absorbed in the collision. A perfect reflection is utilized if the starting point of the crystal's movement is sufficiently away from the boundary of the domain, i.e., its distance is larger than a prescribed tolerance. Otherwise, a random reflection is applied. This random reflection is also used in the case of double reflections at two boundary parts. For details describing the reflection algorithms, it is referred to [4, 6].

1.2.4.2 Modeling of Growth, Coagulation, and Crystal Insertion by Markov Jump Processes

The crystals are allowed to interact with each other only within their current ensemble. In particular, crystals do not have to meet in the same point in space in order to agglomerate, it is enough for them to be contained in a common mesh cell.

Growth, agglomeration, and insertion of crystals are modeled with Markov jump processes. These processes are described in this section, following [52, 53], in terms of the so-called 'stochastic weighted algorithm'. For further technical details, it is referred to [52, 53].

Starting at some time $t \in [0, t_{\text{end}})$, the system stays in the state $\mathcal{E}(t)$ for an exponentially distributed waiting time τ, $P(\tau \geq s) = \exp(-\lambda(\mathcal{E})s)$. Here, $\lambda(\mathcal{E})$ is the waiting time parameter that is the sum of the individual rates of all jumps that are possible in $\mathcal{E}(t)$. This parameter is the sum of the growth jump rate $\lambda_{\text{grow}}(\mathcal{E})$ and the agglomeration jump rate $\lambda_{\text{aggl}}(\mathcal{E})$:

$$\lambda(\mathcal{E}) = \lambda_{\text{grow}}(\mathcal{E}) + \lambda_{\text{aggl}}(\mathcal{E}).$$

First, the simulation of crystal growth will be described. The growth term as it stands in the population balance equation (1.4) is a transport term along the internal coordinate. The rationale behind a stochastic simulation of this term by Markov jump processes is the interpretation of crystal growth as crystal surface growth via a chemical reaction. One can derive a relation between the growth rate $G(c, T)$ and the corresponding reaction rate, e.g., see [5]. A crystal growth jump has an impact on just one crystal e_i. Given a growth height Δm_i, the state of e_i is changed by

$$e_i = (\boldsymbol{x}_i, m_i) \longrightarrow (\boldsymbol{x}_i, m_i + \Delta m_i) =: \tilde{e}_i.$$

The crystal e_j for which the next growth jump occurs is chosen with the probability

$$\frac{G(c, T, m_j)}{\Delta m_i} \left(\lambda_{\text{grow}}(\mathcal{E})\right)^{-1}. \tag{1.16}$$

In our implementation of the SPS method, c and T are assumed to be constant in K in expression (1.16). The total rate for the growth jumps in $\mathcal{E}$ is given by

$$\lambda_{\text{grow}}(\mathcal{E}) = \sum_{i=1}^{N_{\mathcal{E}}} \frac{G(c, T, m_i)}{\Delta m_i}.$$

In agglomeration jumps, two crystals e_i and e_j, with $i < j$, are involved. Such a jump has the form

$$e_i, e_j \longrightarrow (\xi(\boldsymbol{x}_i, \boldsymbol{x}_j), m_i + m_j) =: \tilde{e}_i.$$

After having performed this jump, the crystal e_j is removed from the ensemble and the crystal $\tilde{e}_i$ has to be placed in an appropriate way in the ensemble, i.e., one has to assign an appropriate position to $\tilde{e}_i$. For designing a stable method, it is proposed in [52] to choose the new position $\mathbf{y}$ of a crystal that emerged from coagulation of the crystals $\left(m_i, \boldsymbol{x}_i\right)$ and $\left(m_j, \boldsymbol{x}_j\right)$ stochastically, distributed according to the probabilities

$$P(\mathbf{y} = \boldsymbol{x}_i) = \frac{m_i}{m_i + m_j}, \quad P(\mathbf{y} = \boldsymbol{x}_j) = \frac{m_j}{m_i + m_j},$$

i.e., to use the center of mass in the probabilities. Similarly as for the growth, the total rate of agglomeration jumps is the sum of all individual agglomer-

ation jump rates of pairs of crystals

$$\lambda_{\text{aggl}}(\mathcal{E}) = \frac{1}{2N_{\mathcal{E}}} \sum_{i,j=1}^{N_{\mathcal{E}}} \kappa_{\text{agg}}(m_i, m_j).$$

The involvement of two crystals in an agglomeration jump is random with the probabilities

$$P(e_i \text{ and } e_j \text{ chosen for agglomeration}) = \frac{\kappa_{\text{agg}}(m_i, m_j)}{2N_{\mathcal{E}}}.$$

Ensembles of crystals might be changed also by insertion of crystals in the flow domain. This process affects usually only a few mesh cells. Crystal insertion is modeled by so-called inception jumps, i.e., each ensemble in mesh cells, where crystals are injected, is equipped with an additional jump rate $\lambda_{\text{in}}(\mathcal{E})$ and a corresponding jump, which adds a new crystal to the ensemble.

1.2.4.3 Coupled Simulation with a Splitting Scheme

Our basic approach for developing a code for solving the PBS (1.1)–(1.4) numerically consisted in coupling two separate codes: one designed for simulating the Computational Fluid Dynamic (CFD) equations (1.1)–(1.3) with deterministic methods, and the other one designed for simulating crystal interactions with stochastic methods. For this purpose, we used the in-house codes ParMooN [24, 68] for the CFD part and Brush [53] for the SPS part.

The complete simulation procedure is sketched in Figure 1.6. In each time instance, first the CFD equations are solved and then the population balance equation (1.4) with the SPS method. In order to couple the two codes, an interface was developed and implemented that is responsible for the data transfer between the codes. Other major extensions of Brush that were necessary include the simulation of the transport of the crystals in three dimensions, the implementation of the sedimentation model, the implementation of crystal-wall interactions, and the implementation of routines for assigning the crystals to mesh cells. For more details, it is referred to [4].

1.2.5 Numerical Simulations of a Fluidized Bed Crystallizer

The deterministic-stochastic approach described in Section 1.2.4 was utilized for the simulation of the behavior of the crystallization process for another benchmark problem. The second benchmark was a crystallization process in a fluidized bed crystallizer.

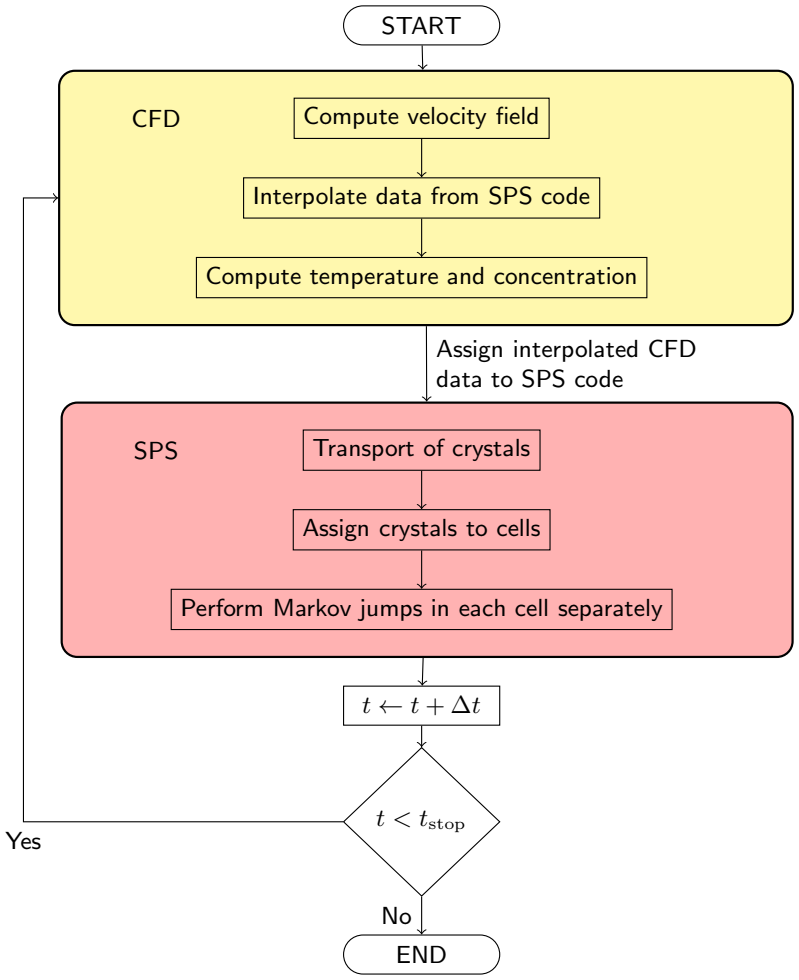

Fig. 1.6 Schematic sketch of the coupled simulation via a splitting scheme.

In a fluidized bed crystallizer, crystal growth and agglomeration can be combined, where the main control variables are temperature profiles and flow rates. Crystals can be separated by size and withdrawn at a varying crystallizer height. The size separation is again controlled by the flow rates.

The experimental implementation of such a crystallizer is depicted in Figure 1.7. Solution is removed from the top of the fluidized bed crystallizer through a filter. It is pumped back into the device from the bottom to fluidize the crystals. The crystallizer is cooled by a double jacket to increase the supersaturation over time. Crystals can be sampled from a variable height in

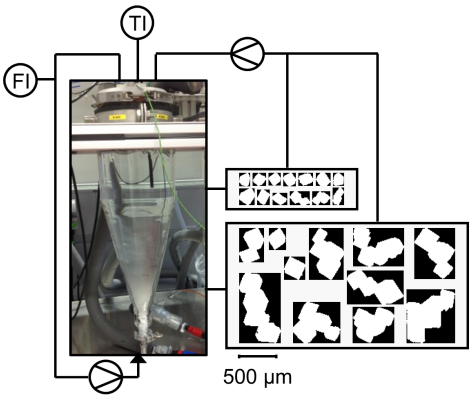

Fig. 1.7 Schematic of the benchmark experiment in the fluidized bed crystallizer with exemplary crystals in different withdrawal heights.

the fluidized bed crystallizer during an experiment. As in the first benchmark problem, the crystal shape can be analyzed by a flow-through microscope.

A PBS of the form (1.1)–(1.4) was used for modeling this process. Figure 1.8 presents the computational domain and its decomposition in tetrahedra. The computational domain neglects the small inlet extension at the bottom, compared with the fluidized bed crystallizer used in the experiment. This modification is caused from an algorithmic issue, since the routine that locates the mesh cell where a crystal is situated after a transport step requires a convex domain. A routine implemented in the research code TetGen [63] was used for this purpose. The grid shown in Figure 1.8 consists of 10 752 tetrahedra.

Preliminary numerical studies showed that the used grids were too coarse for simulating all scales of the flow field. This situation is the typical one that is encountered in the simulation of turbulent flows and it is well known that one has to utilize a turbulence model. There are many proposal for such models, e.g., see [54, 58]. In our simulations, we applied the Smagorinsky Large Eddy Simulation (LES) model, which adds to the momentum equation of the Navier–Stokes equations (1.1) the nonlinear viscous term

$$\nu_{\mathrm{Smago}} \|\nabla \boldsymbol{u}\|_F \nabla \boldsymbol{u} = C_{\mathrm{Smago}} \delta^2 \|\nabla \boldsymbol{u}\|_F \nabla \boldsymbol{u}, \tag{1.17}$$

where δ is the local filter width, which was chosen to be piecewise constant, namely twice the length of the shortest edge of a tetrahedron, C_{Smago} is a user-chosen parameter, and $\|\cdot\|_F$ is the Frobenius norm of a tensor. Numerical studies showed that the value $C_{\mathrm{Smago}} = 5 \cdot 10^{-4}$ was sufficient, which is a comparably small value and which indicates that the flow is only slightly turbulent. In the experiments, a typical average inflow velocity was $U \approx 0.08$ m/s. Together with the choice of a characteristic length $L = 0.1$ m as a typical inner diameter and the density and dynamic viscosity of purified water, the

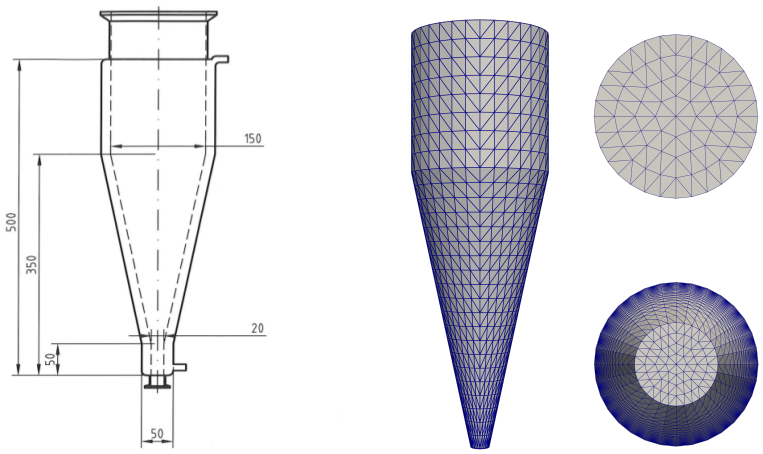

Fig. 1.8 Geometry (in mm) and mesh used in the simulations, left: front view; right: top and bottom views.

Table 1.1 Coefficients for the PBSs modeling the fluidized bed crystallizer process.

name	notation	unit	value/function
density of purified water	ρ	$\mathrm{kg/m^3}$	1050
dynamic viscosity of purified water	μ	$\mathrm{kg/m \cdot s}$	0.0014
diffusion coefficient in (1.2)	D_T	$\mathrm{m^2/s}$	$\frac{\lambda_{\mathrm{susp}}}{\rho_{\mathrm{susp}} C_{\mathrm{susp}}}$
thermal conductivity	λ_{susp}	$\mathrm{W/m \cdot K}$	0.6
suspension density	ρ_{susp}	$\mathrm{kg/m^3}$	1050
suspension specific heat capacity	C_{susp}	$\mathrm{J/kg \cdot K}$	3841
scaling parameter in (1.2)	g_T	$\mathrm{K \cdot m^3/kg}$	$\frac{\Delta h_{\mathrm{cryst}}}{\rho_{\mathrm{susp}} C_{\mathrm{susp}}}$
crystallization enthalpy	$\Delta h_{\mathrm{cryst}}$	$\mathrm{J/kg}$	89100
diffusion coefficient (c)	D_c	$\mathrm{m^2/s}$	$5.4 \cdot 10^{-10}$
scaling parameter (1.2)	g_c	$\mathrm{mol/kg}$	$-\frac{1}{M_{\mathrm{hydrate}}}$
molar mass of hydrate	M_{hydrate}	$\mathrm{kg/mol}$	0.4744
density of crystals	ρ_{cryst}	$\mathrm{kg/m^3}$	1760
Boltzmann constant	k_B	$\mathrm{J/K}$	$1.3806504 \cdot 10^{-23}$
universal gas constant	R	$\mathrm{J/K \cdot mol}$	8.314

Reynolds number is $\mathrm{Re} = \mu U L/\rho \approx 6000$, see Table 1.1 for the values of the physical coefficients. These coefficients were kept constant during the simulation since there were only small variations of the temperature (± 1 K) and the amount of crystals was negligible.

The Navier–Stokes equations (1.1) were discretized in time with the Crank–Nicolson scheme, which is of second order, and the equidistant time step $\Delta t = 0.05$ s. They were linearized with a standard Picard iteration and the arising linear saddle point problems were discretized in space with the

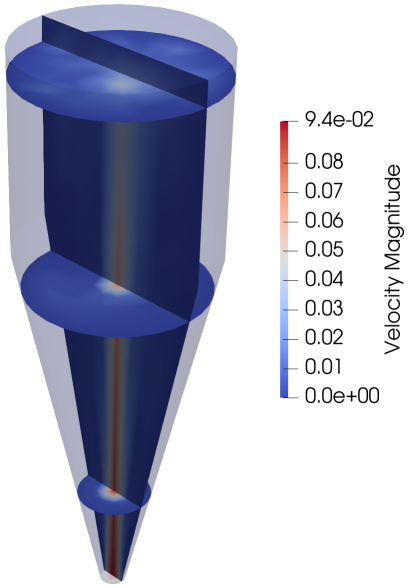

Fig. 1.9 Snapshot of the flow field, inflow rate 56 $^{\mathrm{kg}}/_{\mathrm{h}}$.

popular inf-sup stable pair P_2/P_1, a so-called Taylor–Hood pair of finite element spaces. That means, the velocity was approximated with continuous and piecewise quadratic functions and the pressure with continuous piecewise linear functions. Hence, the resolution of the velocity field is in fact twice as fine as suggested by the grid from Figure 1.8. A snapshot of the flow field is displayed in Figure 1.9. For the temporal discretization of the temperature equation (1.2) and the concentration equation (1.3) also the Crank–Nicolson scheme was used, with the same time step as for the Navier–Stokes equations. The spatial discretization was performed with the linear FEM-FCT scheme from [37, 42] with P_1 finite elements, see Section 1.2.3.2. Finally, the population balance equation (1.4) was simulated with the SPS method described in Section 1.2.4. The breakage of crystals was neglected in the numerical simulations. The coupled PBS was simulated with the splitting scheme presented in Section 1.2.4.3. There were 49419 degrees of freedom for the velocity and 2349 for the pressure, temperature, and concentration.

As final simulation time, $t_{\mathrm{end}} = 1800$ s $= 30$ min was set, such that 36 000 time steps had to be performed. For the flow, the mass flow rate at the inlet was 56 $^{\mathrm{kg}}/_{\mathrm{h}}$ in the whole time interval. The flow field was allowed to develop in the first 30 s. Then, the crystals were inserted in the flow during the time interval $[30\ \mathrm{s}, 40\ \mathrm{s}]$. In contrast to the experiment, where all crystals are inserted into the crystallizer basically at the same time, the crystals enter in the simulations during a short time interval. There is an algorithmic reason, since the SPS method works better if there is a rather uniform distribution

of crystals. The seed mass of the crystals was 10^{-4} kg. It was divided equally into two parts, one with crystals of diameter 75 µm and one of crystals with diameter 125 µm. Both parts were represented via a log-normal distribution with 25 µm standard deviation.

Storage for 256 computational crystals was assigned to each mesh cell of the flow domain Ω. As already mentioned in Section 1.2.4.1, each computational crystal represents a number of physical crystals. In preliminary simulations, 5.0×10^8 $\#\,\text{physical crystals}/\text{m}^3$ was found to be an upper bound for the concentration of physical crystals away from the bottom of the device. In this region, a linear conversion to computational crystals was used such that this upper bound corresponds to 256 computational crystals. Close to the bottom, the concentration of physical crystals was often higher, due to sedimentation. In this region, still a linear conversion was applied, but the conversion factors were increased by 10 below 0.1 m and by 100 below 0.05 m. The choice of the conversion factor is a purely numerical issue. It influences the computational cost and the numerical precision, but otherwise it has no effect on the results for the physical quantities. This setup led to roughly 150 000 computational crystals in Ω after having completed the insertion at 40 s. This number is typically reduced by around 50 % at the end of a simulation because of agglomeration and in addition since, as explained in Section 1.2.4.1, crystals that would leave through the inlet due to sedimentation were removed from the simulations.

The coefficients for the temperature equation (1.2) are given in Table 1.1. The Dirichlet boundary data for the temperature were linearly interpolated in the time interval $[0, t_{\text{end}}]$, where the initial temperature was $T(0\,\text{s}) = 288.95$ K, i.e. 15.8 °C, and the final temperature was $T(3600\,\text{s}) = 288.35$ K, which is 15.2 °C. Also the coefficients for the concentration equation (1.3) are provided in Table 1.1. As initial condition $c(0\,\text{s}) = 207\ {}^{\text{mol}}/_{\text{m}^3}$ was chosen, which corresponds to the saturation concentration at 17 °C.

For the sedimentation, the model (1.15) was utilized. A brief numerical study in [6] showed that one has to choose the scaling factor in this model rather small. Otherwise, too many crystals would leave through the inlet of the domain due to sedimentation, compare Section 1.2.4.1. In [6], $\sigma = 0.1$ was used. In this section, also results obtained with $\sigma = 0.05$ are presented to continue the study with respect to the scaling factor.

For the growth term in the population balance equation (1.4), a model from [64] is utilized

$$G_d = \begin{cases} \frac{\sqrt{2}}{\pi^{\frac{1}{3}}} k_{\text{G}_1} \exp\left(-\frac{k_{\text{G}_2}}{RT}\right) \left(S_{\text{hyd,H}_2\text{O}+} - 1\right)^{k_{\text{G}_3}}\ [{}^{\text{m}}/_{\text{s}}], & \text{if } S_{\text{hyd,H}_2\text{O}+} > 1, \\ 0 & \text{else,} \end{cases}$$

where the model parameters are given by $k_{\text{G}_1} = 5 \cdot 10^7\ {}^{\text{m}}/_{\text{s}}$, $k_{\text{G}_2} = 75 \cdot 10^3\ {}^{\text{J}}/_{\text{mol}}$, $k_{\text{G}_3} = 1.4$. The factor ${}^{\sqrt{2}}/_{\pi^{\frac{1}{3}}}$ comes from converting an octahedral to a spherical crystal shape. The quantity

$$S_{\mathrm{hyd,H_2O+}} = \frac{w_{\mathrm{hyd,H_2O+}}}{w^{\mathrm{eq}}_{\mathrm{hyd,H_2O+}}(T)} \ [\mathrm{kg/kg}]$$

is the relative supersaturation of the solution. Here, $w_{\mathrm{hyd,H_2O+}}$ [kg/kg] is the current mass loading and $w^{\mathrm{eq}}_{\mathrm{hyd,H_2O+}}(T)$ [kg/kg] the mass loading in equilibrium given by

$$w^{\mathrm{eq}}_{\mathrm{hyd,H_2O+}}(T) = a_1 + a_2 T + a_3 T^2 + a_4 T^3 + a_5 T^4 \quad \left[\frac{\text{kg hydrate}}{\text{kg added water}}\right],$$

with coefficients $a_1 = 0.0506$, $a_2 = 0.0023$, $a_3 = 7.76 \cdot 10^{-5}$, $a_4 = -2.43 \cdot 10^{-6}$, and $a_5 = 4.86 \cdot 10^{-8}$. This solubility model is known to be valid in a temperature range from 10 °C to 60 °C. To apply this growth model in our simulations, a number of conversions had to be made, see [4, 6] for details.

For the agglomeration kernel in (1.5), the Brownian kernel

$$\kappa_{\mathrm{agg}}(T, m_1, m_2) = \kappa \frac{2Tk_B}{3\mu} \left(\frac{1}{d(m_1)} + \frac{1}{d(m_2)}\right) (d(m_1) + d(m_2)) \quad \left[\mathrm{m^3/s}\right] \tag{1.18}$$

was utilized, where κ is a scaling parameter and $d(m) = \sqrt[3]{6m/\rho_{\mathrm{cryst}}\pi}$ [m] is the sphere equivalent diameter. The same kernel was applied in the simulations presented in [6], where different values of $\kappa \leq 5000$ were tested. In fact, most results from [6] were computed with $\kappa = 5000$. However, in other applications, where we used the Brownian kernel, we found higher values of the scaling factor, e.g., $\kappa = 7000$ in [3] and even $\kappa \in [200\,000, 300\,000]$ for a strongly agglomeration-dominated problem studied in [31]. For this reason, we continued the numerical studies with respect to the scaling factor of the Brownian kernel to higher values of κ and the results will be presented in this section.

The internal coordinate in the PBS is crystal mass. However, for the evaluation of the numerical simulations, the sphere equivalent diameter in µm will be used, since this facilitates the interpretation of the computational results and the comparison with the experimental data.

In the simulations, the nuclei were of 5 µm diameter. Figure 1.10 presents the temporal development of the average crystal diameter in the whole fluidized bed crystallizer for the two considered parameters $\sigma \in \{0.05, 0.1\}$ in the sedimentation model (1.15) and for different values of the parameter κ in the Brownian agglomeration kernel (1.18). For both values of σ there is the same tendency: the larger κ, the larger is the average diameter. For smaller values of κ, the temporal growth of the average diameter is approximately linear in the considered time interval. There is also a linear growth in the first part of the time interval for larger values. But then, a flattening of the curves can be observed. At the final time, one obtains in average larger crystals with $\sigma = 0.1$. With this higher value of the sedimentation parameter, there is a higher concentration of crystals close to the inlet, which increases the probability for agglomeration events in this region. These crystals are com-

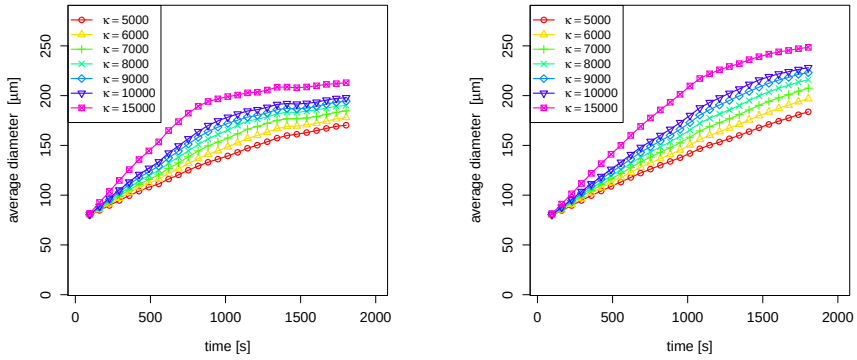

Fig. 1.10 Dependency of the average crystal diameter on the parameter κ of the Brownian agglomeration kernel: left $\sigma = 0.05$, right $\sigma = 0.1$. Averaging was performed for all crystals with diameter larger or equal to 5 µm.

parably large since the sedimentation velocity depends also on the mass of the crystals, such that the agglomeration events lead to even larger crystals.

In the experiment, the smallest measurable crystals were of diameter 50 µm. In order to compare numerical results and experimental data, the same value was used as lower threshold for computing the average diameter of the simulation results. Experimental data are displayed in Figure 1.11. One can see that in the considered time interval, the averaged diameter increased approximately linearly by around 80 µm. At the final time, the average diameter is between 234 µm and 261 µm. There is no separation of different sizes of crystals in different heights of the fluidized bed crystallizer. In the results for the simulations, Figures 1.12 and 1.13, the value of the coordinate z comprises all computational crystals in the interval $[z - 0.025, z + 0.025]$ of 5 cm width.

Figure 1.12 presents the results obtained with the parameter $\sigma = 0.1$ in the sedimentation model (1.15), which is the same parameter as used in [6]. One can observe that for all parameters κ of the Brownian kernel (1.18) there is more or less a linear increase of the crystal diameter only at the beginning of the process. In the last part of the time interval, the average diameter is nearly constant. There is a slight increase of the average diameter with an increase of κ. For $\kappa = 5000$, the average diameter at the final time is in the interval $[187, 205]$ µm and for $\kappa = 10000$, it is in the interval $[196, 215]$ µm. There is a clear separation of the average diameter with respect to the regions of the fluidized bed crystallizer. The largest crystals are close to the inlet and the smallest crystals in the upper region. Comparing the curves of Figures 1.10 and 1.12, one can observe that there are only comparatively small differences of the average diameter at the final time. Hence, there are not many small crystals left in the fluidized bed crystallizer.

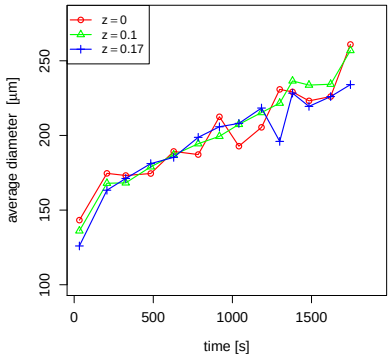

Fig. 1.11 Development of the average diameter in different heights [m] of the fluidized bed crystallizer, experimental results. Crystals with diameter larger or equal to 50 µm were measured. The average diameter is between 234 µm and 261 µm at the final time.

The results for the newly considered segmentation parameter $\sigma = 0.05$ are shown in Figure 1.13. For this parameter, there is in a long part of the time interval an almost linear increase of the average diameter. Like for $\sigma = 0.1$, the average diameter increases if the parameter κ of the Brownian kernel increases and there is layering of the crystals with the largest crystals close to the inlet and the smallest crystals in the upper part of the device. The average crystal parameter at the final time is between 211 µm and 236 µm for $\kappa = 5000$ and for $\kappa = 10000$, it is the interval $[241, 272]$ µm. From comparing Figures 1.10 and 1.13, one can see that the average diameter at the final time is considerably larger if the small crystals with diameter smaller than 50 µm are neglected. Hence, it seems there are still many small crystals in the fluidized bed crystallizer.

Altogether, the results show the enormous impact of the choice of the sedimentation parameter σ in model (1.15) on the obtained computational results. The results for $\sigma = 0.05$ are considerably closer to the experimental data, both with respect to the nearly linear increase of the average diameter and with respect to the average diameter at the final time, than the results for $\sigma = 0.1$. There is a particularly good agreement with respect to the second issue for $\kappa = 8000$, where the average diameter is in the interval $[232, 260]$ µm.

1.3 Bi-variate Processes

The evolution of the crystal population is defined in (1.4) for a d_{int}-dimensional internal property coordinate. The internal coordinates are estimates for the crystal size and shape. In Section 1.2.1, the description of a univariate sub-

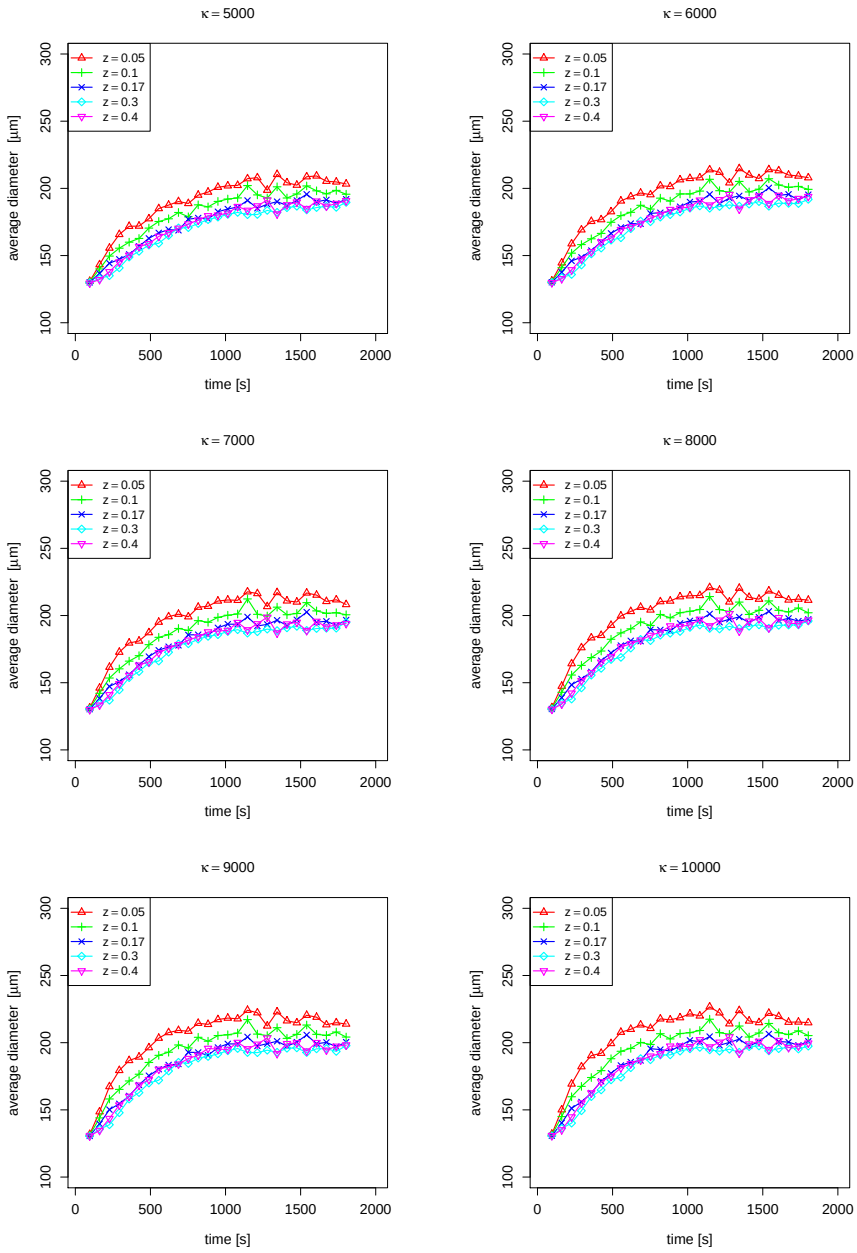

Fig. 1.12 Development of the average diameter in different heights of the fluidized bed crystallizer, $\sigma = 0.1$ and $\kappa \in \{5000, 6000, 7000, 8000, 9000, 10000\}$, top left to bottom right. Averaging was performed for all crystals with diameter larger or equal to 50 µm.

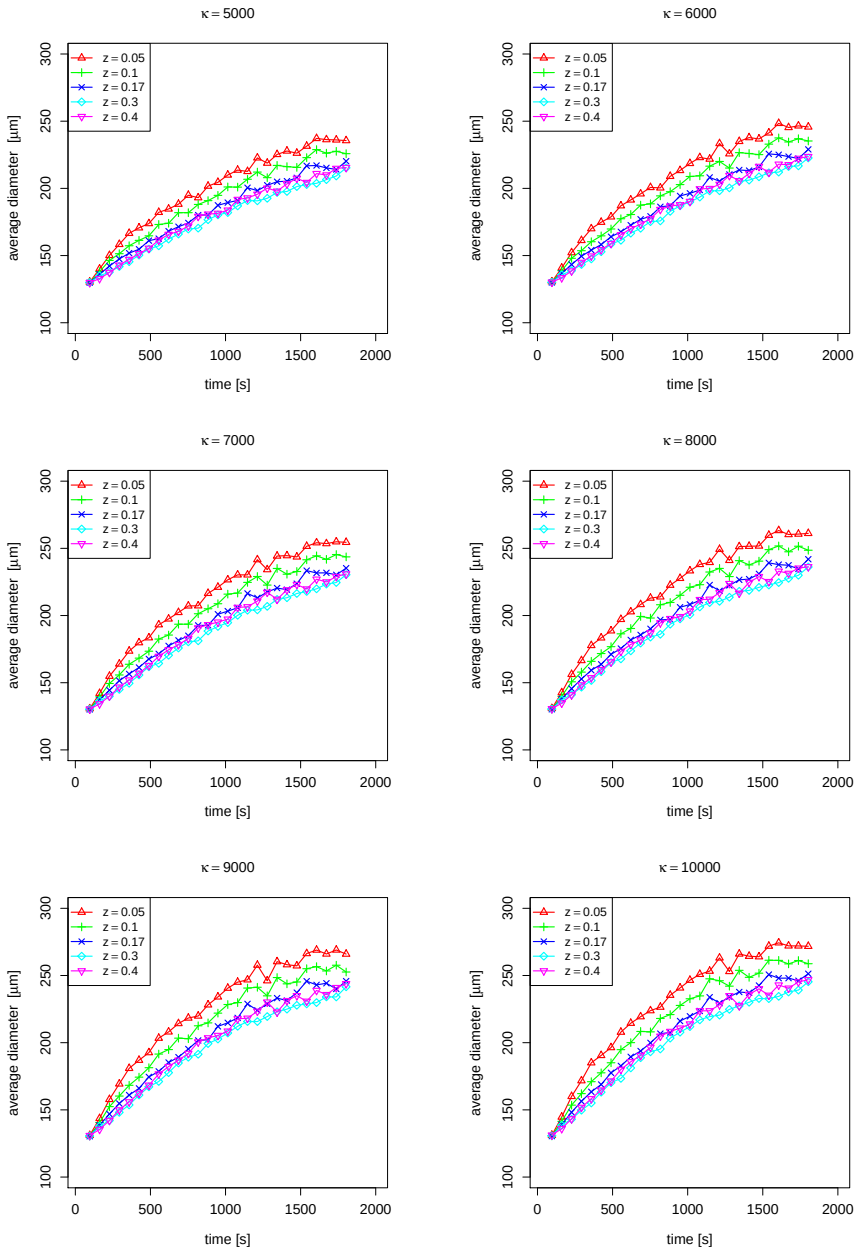

Fig. 1.13 Development of the average diameter in different heights of the fluidized bed crystallizer, $\sigma = 0.05$ and $\kappa \in \{5000, 6000, 7000, 8000, 9000, 10000\}$, top left to bottom right. Averaging was performed for all crystals with diameter larger or equal to 50 µm.

stance was introduced. An example bivariate substance is potassium dihydrogen phosphate. Also for the bivariate system, the 3d-crystal shape of single crystals can be determined with high accuracy [15]. For agglomerated particles, further descriptors can be selected to describe the size and shape of a particle that is composed of several primary particles. Primary and agglomerated potash alum crystals are depicted in Figure 1.7. The descriptors for agglomerates may again be based on a shape estimation, e.g., the projections may be fitted to geometrical polytopes [61]. There is a large number of further shape descriptors, such as the Feret diameter [23], the length of the boundary curve of a projected particle, the projection area, the area of the convex hull of the projection, the diameter, perimeter, and volume of a circle of the same projected area, the widths of the major and minor axes of an ellipse, the convexity [23], the eccentricity [71], the sphericity, and the fractal dimension [65]. Here, the volume of a sphere of equivalent diameter is chosen since the agglomerates in the considered benchmark process are compact. The volume is used to calculate the mass of a crystal. The mass is assumed to be an additive property.

1.3.1 Agglomeration Kernel Identification From Experiments

In Section 1.2.5, an agglomeration dominated crystallizer was presented. Agglomeration depends on the local distribution of crystals in the fluidized bed crystallizer (FBC), which is determined by the fluid dynamics in the FBC [6]. The particle movement was therefore simulated as described in Section 1.2.5. In the agglomeration term (1.5) of the PBE, the agglomeration kernel κ_{agg} determines the rate of agglomeration. Required agglomeration kernels can be identified from measurement data and numerical simulations by solving inverse problems. An example of such measurement data is shown schematically in Figure 1.14.

The agglomeration kernel is usually an unknown functional relation of the volumes of agglomerating particles. Thus, the identification of the kernel is an ill-conditioned problem. To solve the problem, two approaches can be applied. A set of unknown parameters can be identified using measurement data. For this approach, the structure of the kernel has to be known, which can be estimated from modeling the agglomeration process [51] or from known approaches [17]. A second approach is the solution of inverse problems [19, 69]. This approach is based on the measurement data and the dynamic agglomeration model whereas a priori knowledge on the kernel is not required. Solving the inverse problem, the kernel can be approximated with Laurent polynomials [22]. Like this, the kernel can be described with a small set of parameters and it is separable. An efficient calculation of a separable source term is possible via the fast Fourier transform [16, 30].

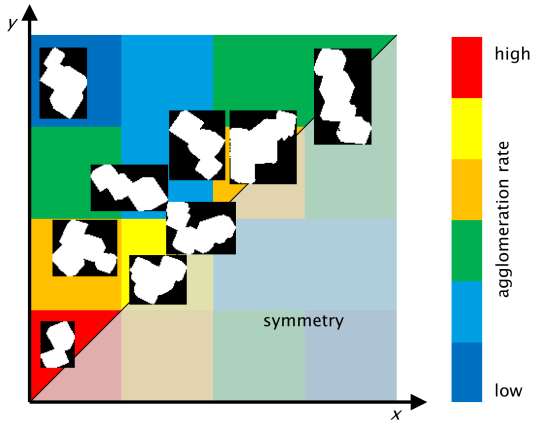

Fig. 1.14 Exemplary agglomeration kernel for stronger agglomeration of small crystals for the internal size coordinates x and y of two agglomerating particles.

1.3.2 Efficient Evaluation of Agglomeration Terms

This section is concerned with the efficient evaluation of the agglomeration terms (1.5) in the univariate case ($d_{\text{int}} = 1$) as well as the extension to multivariate distributions ($d_{\text{int}} \geq 2$). The foundation for an efficient method for the univariate case has been laid in [30] and numerically realized, tested and extended in [16, 21, 56, 57, 62]. In a multivariate case, the particle properties are denoted by $\mathbf{m} = (m_1, \ldots, m_{d_{\text{int}}}) \in \mathbb{R}_{\geq 0}^{d_{\text{int}}}$ with a maximum value of $m_{\max}$, i. e., $m_j \in [0, m_{\max}]$. In this section, the internal properties are not associated with physical units (e.g. length or mass) but treated as dimensionless quantities. For simplicity of notation, the kernel is assumed to be only dependent on the particle properties $\mathbf{m}$ and $\mathbf{m}'$ but neither on time nor location. Under these assumptions, the agglomeration term is given by

$$\begin{aligned} F_{\text{agg}}(f, \mathbf{m}) &= F_{\text{agg}}^{+}(f, \mathbf{m}) - F_{\text{agg}}^{-}(f, \mathbf{m}) \\ &= \frac{1}{2} \int_0^{m_1} \cdots \int_0^{m_{d_{\text{int}}}} \kappa_{\text{agg}}(\mathbf{m} - \mathbf{m}', \mathbf{m}') f(\mathbf{m} - \mathbf{m}') f(\mathbf{m}') \, d\mathbf{m}' \\ &\quad - \int_0^{m_{\max}} \cdots \int_0^{m_{\max}} \kappa_{\text{agg}}(\mathbf{m}, \mathbf{m}') f(\mathbf{m}) f(\mathbf{m}') \, d\mathbf{m}', \end{aligned} \tag{1.19}$$

where $F_{\text{agg}}^{+}(f, \mathbf{m})$ denotes the source term and $F_{\text{agg}}^{-}(f, \mathbf{m})$ denotes the sink term of the agglomeration process. This definition of the source term does not account for any particles forming with a property larger than the maximum $m_{\max}$. The sink term, however, allows a particle to disappear, when it agglomerates with another particle to one with property larger than $m_{\max}$.

Hence, technically particles may be lost over time if $m_{\max}$ is too small. The choice of $m_{\max}$ should reflect this consideration. The two key ingredients toward the proposed efficient evaluation of these integrals are a discretization of the property space Ω_{int} on a uniform grid and a separable approximation of the agglomeration kernel,

$$\kappa_{\text{agg}}(\mathbf{m}, \mathbf{m}') \approx \sum_{\nu=1}^{M} \alpha^\nu(\mathbf{m}) \cdot \beta^\nu(\mathbf{m}') \tag{1.20}$$

for a moderate *separation-rank* $M \in \mathbb{N}$ of the kernel κ_{agg}. This allows to simplify the convolution-type integral of $F^+_{\text{agg}}(f, \mathbf{m})$ to a sum of M multi-dimensional convolution integrals,

$$\begin{aligned} F^+_{\text{agg}}(f, \mathbf{m}) &= \frac{1}{2} \int_0^{m_1} \cdots \int_0^{m_{d_{\text{int}}}} \sum_{\nu=1}^{M} \alpha^\nu(\mathbf{m} - \mathbf{m}') \beta^\nu(\mathbf{m}') f(\mathbf{m} - \mathbf{m}') f(\mathbf{m}') \, d\mathbf{m}' \\ &= \frac{1}{2} \sum_{\nu=1}^{M} \int_0^{m_1} \cdots \int_0^{m_{d_{\text{int}}}} \phi^\nu(\mathbf{m} - \mathbf{m}') \psi^\nu(\mathbf{m}') \, d\mathbf{m}' \end{aligned} \tag{1.21}$$

with $\phi^\nu(\mathbf{m}) := \alpha^\nu(\mathbf{m}) f(\mathbf{m})$ and $\psi^\nu(\mathbf{m}) := \beta^\nu(\mathbf{m}) f(\mathbf{m})$.

Analogously, the sink term results in

$$\begin{aligned} F^-_{\text{agg}}(f, \mathbf{m}) &= \int_0^{m_{\max}} \cdots \int_0^{m_{\max}} \sum_{\nu=1}^{M} \alpha^\nu(\mathbf{m}) f(\mathbf{m}) \beta^\nu(\mathbf{m}') f(\mathbf{m}') \, d\mathbf{m}' \\ &= \sum_{\nu=1}^{M} \phi^\nu(\mathbf{m}) \cdot \int_0^{m_{\max}} \cdots \int_0^{m_{\max}} \psi^\nu(\mathbf{m}') \, d\mathbf{m}'. \end{aligned} \tag{1.22}$$

In particular, $\mathbf{m}$-dependent terms have been factored out of the integral which reduces the complexity to evaluate $F^-_{\text{agg}}(f, \mathbf{m})$.

1.3.2.1 Discretization of the Property Space

In order to evaluate the integrals in (1.21) and (1.22) numerically, a suitable discretization of the property space Ω_{int} to discretize $f(\mathbf{m})$ is introduced. One first defines a uniform tensor grid $\mathcal{G}$ by choosing the number of degrees of freedom per property, n, and divides the interval $(0, m_{\max})$ into n sub-intervals of width $h := \frac{m_{\max}}{n}$ which is used to define grid points $\mathbf{g_j} = (j_1 h, \cdots, j_{d_{\text{int}}} h)$ and cells

$$\mathcal{C}_\mathbf{j} := (j_1 h, j_1 h + h) \times \cdots \times (j_{d_{\text{int}}}, j_{d_{\text{int}}} h + h), \text{ for } \mathbf{j} \in \{0, \ldots, n-1\}^{d_{\text{int}}}.$$

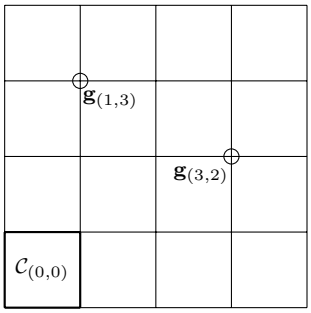

Fig. 1.15 A uniform tensor grid with $d_{\text{int}} = 2$ and $n = 4$.

An example of this grid with $d_{\text{int}} = 2$ and $n = 4$ is given in Figure 1.15. Each of the $N := n^{d_{\text{int}}}$ cells has volume of $V := h^{d_{\text{int}}}$.

In the following derivations, the density distribution $f(\mathbf{m})$ (and the kernel factors $\alpha^\nu(\mathbf{m})$ and $\beta^\nu(\mathbf{m}')$) are discretized to be piecewise constant with respect to this grid $\mathcal{G}$, i. e.,

$$f(\mathbf{m}) = f(\mathbf{m}') =: f_{\mathbf{j}} \quad \text{if } \mathbf{m}, \mathbf{m}' \in \mathcal{C}_{\mathbf{j}}, \tag{1.23}$$

hence the function $f(\mathbf{m})$ is approximated by a tensor $f \in \mathbb{R}^{n \times \cdots \times n}$ with N entries $f_{\mathbf{j}}$.

For piecewise constant integrands, the agglomeration integrals (1.21) and (1.22) can be evaluated exactly at all grid points through evaluation of the nested sums

$$F^+_{\text{agg}}(\mathbf{g}_{\mathbf{j}+\mathbf{1}}) = \frac{V}{2} \sum_{\nu=1}^{M} \sum_{k_1=0}^{j_1} \cdots \sum_{k_{d_{\text{int}}}=0}^{j_{d_{\text{int}}}} \phi^\nu_{\mathbf{j}-\mathbf{k}} \cdot \psi^\nu_{\mathbf{k}} =: \frac{V}{2} \sum_{\nu=1}^{M} Q^{+,\nu}_{\text{agg}}(\mathbf{j}), \tag{1.24}$$

using $Q^{+,\nu}_{\text{agg}} \in \mathbb{R}^{n \times \cdots \times n}$ in (1.24) to denote the unscaled and unshifted result of the discrete convolution. The efficient evaluation of $Q^{+,\nu}_{\text{agg}}$ will be the focus of subsection 1.3.2.2 to reduce the complexity of the straightforward evaluation $\mathcal{O}(N^2)$ to a log-linear complexity of $\mathcal{O}(N \log N)$. The resulting F^+_{agg} is piecewise linear and needs to be projected to a piecewise constant function. This issue is addressed in subsection 1.3.2.3.

The sink-term (1.22) within a cell $C_{\mathbf{j}}$ is computed as

$$F^-_{\text{agg}}(\mathbf{m})|_{\mathcal{C}_{\mathbf{j}}} = V \cdot \sum_{\nu=1}^{M} \phi^\nu_{\mathbf{j}} \cdot \sum_{k_1=0}^{n} \cdots \sum_{k_{d_{\text{int}}}=0}^{n} \psi^\nu_{\mathbf{k}} =: V \cdot \sum_{\nu=1}^{M} \phi^\nu_{\mathbf{j}} \cdot S^{-,\nu}_{\text{agg}} \tag{1.25}$$

with a scalar $S^{-,\nu}_{\text{agg}}$ as the result of the summation. The computation of (1.25) is of complexity $\mathcal{O}(kN)$ and results in a piecewise constant function (in the

form of a tensor with N entries) corresponding to the number of disappearing particles in each cell.

1.3.2.2 Efficient Evaluation of a Discrete Convolution via Fourier Transform

This section deals with the efficient evaluation of

$$Q^+_{\text{agg}}(\mathbf{j}) = \sum_{k_1=0}^{j_1} \cdots \sum_{k_{d_{\text{int}}}=0}^{j_{d_{\text{int}}}} \phi_{\mathbf{j}-\mathbf{k}} \cdot \psi_{\mathbf{k}} \tag{1.26}$$

which is required in order to compute the source term $F^+_{\text{agg}}(f, \mathbf{m})$ in (1.24). Since the computation is analogous for all kernel factors, the index ν has been dropped.

It is well known that a discrete convolution (1.26) can be evaluated simultaneously for all $\mathbf{j}$ using the multi-dimensional convolution theorem ([50]),

$$Q^+_{\text{agg}} = \mathcal{F}^{-1}(\mathcal{F}(\phi) \odot \mathcal{F}(\psi)), \tag{1.27}$$

where $\mathcal{F}$ and $\mathcal{F}^{-1}$ denote the Fourier transform and its inverse and $\odot$ denotes the elementwise (or Hadamard) product.

The result of a convolution of a tensor of size $n \times \cdots \times n$ is a tensor of size $2n \times \cdots \times 2n$ with an index $\mathbf{j} \in \{0, \ldots, 2n-1\}^{d_{\text{int}}}$. However, one is only interested in the $n \times \cdots \times n$ subtensor since all other entries go beyond the computational domain (properties larger than $m_{\max}$). In order to calculate this full convolution result via a sequence of univariate Fourier transforms, the input tensors ϕ and ψ need to be enlarged to this size by adding zeros. One then obtains tensors $\widetilde{\phi}, \widetilde{\psi} \in \mathbb{R}^{2n \times \cdots \times 2n}$ with entries

$$\widetilde{\phi}_{\mathbf{j}}, \widetilde{\psi}_{\mathbf{j}} = \begin{cases} \phi_{\mathbf{j}}, \psi_{\mathbf{j}} & \text{if } \mathbf{j} \in \{0, \ldots, n-1\}^{d_{\text{int}}}, \\ 0 & \text{else,} \end{cases} \tag{1.28}$$

in a process called *zero-padding*.

The multivariate Fourier transform (1.27) is defined by

$$\begin{aligned} &\mathcal{F} : \mathbb{R}^{2n \times \cdots \times 2n} \to \mathbb{C}^{2n \times \cdots \times 2n}, \quad \widetilde{\phi} \mapsto \mathcal{F}(\widetilde{\phi}) \qquad \text{with} \\ &(\mathcal{F}(\widetilde{\phi}))_{\mathbf{j}} := \sum_{\mathbf{s}=\mathbf{0}}^{\mathbf{2n-1}} \widetilde{\phi}_{\mathbf{s}} \cdot \prod_{q=1}^{d_{\text{int}}} e^{\mathrm{i}\pi s_q j_q / n}. \end{aligned} \tag{1.29}$$

The function $\mathcal{F}$ is rewritten in the form

$$\begin{aligned}\mathcal{F}(\widetilde{\phi}) &= \sum_{\mathbf{s}=\mathbf{0}}^{2\mathbf{n}-1} \widetilde{\phi}_{\mathbf{s}} \cdot \prod_{q=1}^{d_{\text{int}}} e^{i\pi s_q j_q/n} \\ &= \sum_{s_{d_{\text{int}}}=0}^{2n-1} \cdots \left(\sum_{s_1=0}^{2n-1} \widetilde{\phi} \cdot e^{\mathrm{i}\pi s_1 j_1/n} \right) \cdots e^{\mathrm{i}\pi s_{d_{\text{int}}} j_{d_{\text{int}}}/n} \\ &= \mathcal{F}^{d_{\text{int}}} \circ \mathcal{F}^{d_{\text{int}}-1} \circ \cdots \circ \mathcal{F}^1(\widetilde{\phi}), \end{aligned} \tag{1.30}$$

reducing it to a composition of univariate Fourier transforms in the q^{th} dimension,

$$\begin{aligned}\mathcal{F}^q : \mathbb{C}^{2n\times\cdots\times 2n} \to \mathbb{C}^{2n\times\cdots\times 2n}, \quad \widetilde{\phi} \mapsto \mathcal{F}^q(\widetilde{\phi}) \qquad \text{with} \\ (\mathcal{F}(\widetilde{\phi}))^q_{\mathbf{j}} := \sum_{s=0}^{2n-1} \widetilde{\phi}_{j_1,\cdots,j_{m-1},s,j_{m+1},\ldots,j_{d_{\text{int}}}} \cdot e^{\mathrm{i}\pi s j_q/n}.\end{aligned} \tag{1.31}$$

The implication is that the complete Fourier transform of $\widetilde{\phi}$ can be computed via a sequence of one-dimensional Fourier transforms. Every $\mathcal{F}^q$ can be calculated via multiple applications of the FFT-algorithm [20]. This reduces the complexity of each one-dimensional Fourier transform to $\mathcal{O}(n \log n)$ and hence reduces the complexity of $\mathcal{F}$ down to $\mathcal{O}(d_{\text{int}} n^{d_{\text{int}}} \log n) = \mathcal{O}(N \log N)$. The same techniques are employed for the inverse Fourier transform to calculate the complete convolution in $\mathcal{O}(N \log N)$ instead of $\mathcal{O}(N^2)$ without using FFT.

An additional acceleration of the calculation is achieved by exploiting the zero-padding, which is necessary to obtain the full convolution result. When computing $\mathcal{F}^1(\widetilde{\phi})$, one needs to calculate $(2n)^{d_{\text{int}}-1}$ fast Fourier transforms of length $2n$, each one over $d_{\text{int}} - 1$ fixed indices j_2 through $j_{d_{\text{int}}}$ from 0 to $2n-1$. By taking the zero-pattern of $\widetilde{\phi}$ into account, many one-dimensional Fourier transforms are applied to zero-vectors which can be skipped to save computational time. These superfluous Fourier transforms are characterized by a multi-index $\mathbf{j}$ with at least one $j_q > n$ for $q > 1$. This reduces the number of one-dimensional FFTs during the computation of $\mathcal{F}^1$ from $(2n)^{d_{\text{int}}-1}$ to $n^{d_{\text{int}}-1}$, a factor of $2^{d_{\text{int}}-1}$ compared to the straightforward implementation. The same argument can be used to reduce the number of one-dimensional Fourier transforms in the subsequent calculations of $\mathcal{F}^2$ to $\mathcal{F}^{d_{\text{int}}-1}$ as part of the zero-pattern is preserved. An illustration of this zero-pattern for $d_{\text{int}} = 3$ is shown in Figure 1.16. The number of one-dimensional Fourier transforms for the computation of $\mathcal{F}^q$ is reduced to $2^q \cdot n^{d_{\text{int}}-1}$, reducing the total number of one-dimensional Fourier transforms from $d \cdot (2n)^{d_{\text{int}}-1}$ to $(2^{d_{\text{int}}} - 1) n^{d_{\text{int}}-1}$. The total complexity is thereby reduced to $\mathcal{O}(N \log n)$. Further details can be found in [1].

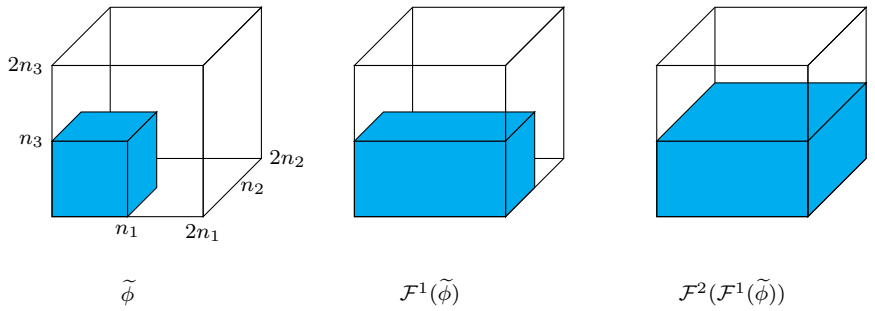

Fig. 1.16 Illustration of the non-zero-pattern of $\widetilde{\phi}$ and the intermediate results of its Fourier transform $\mathcal{F}(\widetilde{\phi})$.

1.3.2.3 Conservation of Multivariate Moments

So far, the source term (1.24) has been calculated at the grid-points $\mathbf{g_j}$ that can efficiently be calculated via the procedure outlined in subsection 1.3.2.2. The function $F^+_{\text{agg}}(f, \mathbf{m})$ is piecewise linear with respect to all internal variables since it is the result of an integration of a piecewise constant function. The values $F_\mathbf{j} = F(\mathbf{g_j})$ of the function at every grid point $\mathbf{g_j}$ are given, $F^+_{\text{agg}}(f, \mathbf{m})$ is represented with the standard basis of piecewise linear "hat" functions

$$\Lambda_\mathbf{j}(\mathbf{m}) = \prod_{q=1}^{d_{\text{int}}} T_{j_q}(m_q) \quad \text{with} \tag{1.32}$$

$$T_{j_q}(m_q) = \begin{cases} m_q/h - j_q + 1 & \text{, if } (j_q - 1) \cdot h \leq m_q \leq j_q \cdot h, \\ -m_q/h + j_q + 1 & \text{, if } j_q \cdot h \leq m_q \leq (j_q + 1) \cdot h, \\ 0 & \text{, else,} \end{cases} \tag{1.33}$$

with standard hat functions $T_{j_q}(\cdot)$. The function $\Lambda_\mathbf{j}(\mathbf{m})$ satisfies $\Lambda_\mathbf{j}(\mathbf{g_j}) = 1$ and $\Lambda_\mathbf{j}(\mathbf{g}_{\widetilde{\mathbf{j}}}) = 0$ if $\widetilde{\mathbf{j}} \neq \mathbf{j}$ which allows to write

$$F^+_{\text{agg}}(f, \mathbf{m}) = \sum_{\mathbf{j}=0}^{\mathbf{n}-1} F_\mathbf{j} \cdot \Lambda_\mathbf{j}(\mathbf{m}).$$

Since the result is piecewise linear, it does not satisfy (1.23) and requires a projection. It is possible to construct a projection that preserves all square-free moments.

A multivariate moment $M_\mathbf{e}(f)(t)$ (for a vector $\mathbf{e} = (e_1, \ldots, e_{d_{\text{int}}}) \in \mathbb{N}_0^{d_{\text{int}}}$) of a particle density distribution $f(\mathbf{m})$ at time t is defined by

$$M_{\mathbf{e}}(f)(t) := \iint\limits_{\Omega_{\text{int}}} f(\mathbf{m}) \prod_{q=1}^{d_{\text{int}}} m_q^{e_q} \, d\mathbf{m}, \tag{1.34}$$

and a moment $M_{\mathbf{e}}(f)(t)$ with all $e_q < 2$ is called *square-free*, i. e., if $\mathbf{e} \in \{0,1\}^{d_{\text{int}}}$.

It is a natural choice to distribute all particles associated with a single basis function $\Lambda_{\mathbf{j}}(\mathbf{m})$ onto its $2^{d_{\text{int}}}$ cells of support denoted by $\mathcal{C}_{\mathbf{j}+\mathbf{k}}$ with $\mathbf{k} = \{-1,0\}^{d_{\text{int}}}$ to preserve all $2^{d_{\text{int}}}$ square-free moments. This can be done for each basis function individually since local preservation implies global preservation of moments.

One calculates

$$\begin{aligned} M_{\mathbf{e}}(\Lambda_{\mathbf{j}}) &= \iint\limits_{\Omega_{\text{int}}} \Lambda_{\mathbf{j}}(\mathbf{m}) \prod_{q=1}^{d_{\text{int}}} m_q^{e_q} \, d\mathbf{m} \\ &= F_{\mathbf{j}} \prod_{q=1}^{d_{\text{int}}} \int_{j_q h - h}^{j_q h + h} m_q^{e_q} T_{j_q}(m_q) \, dm_q = F_{\mathbf{j}} \prod_{q=1}^{d_{\text{int}}} \mathcal{I}_{e_q}^{j_q}, \end{aligned}$$

with

$$\mathcal{I}_{e_q}^{j_q} := \int\limits_{(j_q-1)h}^{(j_q+1)h} m_q^{e_q} T_{j_q}(m) \, dm = \begin{cases} h & \text{, if } e_q = 0, \\ h^2 j_q & \text{, if } e_q = 1, \end{cases} \tag{1.35}$$

to simplify notation. A cell $\mathcal{C}_{\mathbf{j}+\mathbf{k}}$ with an associated piecewise constant value $w_{\mathbf{j}+\mathbf{k}}$ carries moments determined by

$$\begin{aligned} M_{\mathbf{e}}(\mathcal{C}_{\mathbf{j}}) &= \iint\limits_{\Omega_{\text{int}}} w_{\mathbf{j}} \prod_{q=1}^{d_{\text{int}}} m_q^{e_q} \, d\mathbf{m} \\ &= w_{\mathbf{j}+\mathbf{k}} \prod_{q=1}^{d_{\text{int}}} \int\limits_{(j_q+k_q)h}^{(j_q+k_q+1)h} m_q^{e_q} \, dm_q = w_{\mathbf{j}+\mathbf{k}} \prod_{q=1}^{d_{\text{int}}} \mathcal{J}_{e_q}^{j_q+k_q}, \end{aligned}$$

with

$$\mathcal{J}_{e_q}^{j_q} := \int\limits_{j_q h}^{(j_q+1)h} m^{e_q} \, dm = \begin{cases} h & \text{, if } e_q = 0, \\ h^2 \cdot (j_q + 0.5) & \text{, if } e_q = 1, \end{cases} \tag{1.36}$$

to again simplify the integral.

Moment equality can be preserved by choosing values $w_{\mathbf{j}+\mathbf{k}}$ that satisfy

$$M_{\mathbf{e}}(\Lambda_{\mathbf{j}}) = \sum_{\mathbf{k}=\{-1,0\}^{d_{\text{int}}}} M_{\mathbf{e}}(\mathcal{C}_{\mathbf{j}+\mathbf{k}})$$

$$\iff \quad F_{\mathbf{j}} \prod_{q=1}^{d_{\text{int}}} \mathcal{I}_{e_q}^{j_q} = \sum_{\mathbf{k}=\{-1,0\}^{d_{\text{int}}}} w_{\mathbf{j}+\mathbf{k}} \prod_{q=1}^{d_{\text{int}}} \mathcal{J}_{e_q}^{j_q+k_q}. \tag{1.37}$$

By using

$$\mathcal{J}_e^{j-1} + \mathcal{J}_e^j = \left\{ \begin{array}{l} 2h \ , \text{ if } e = 0, \\ 2h^2 j \ , \text{ if } e = 1 \end{array} \right\} = 2\mathcal{I}_e^j,$$

which follows directly from the definitions of (1.35) and (1.36), one pairs the $2^{d_{\text{int}}}$ summands in (1.37) and obtains

$$w_{\mathbf{j}+\mathbf{k}} = \frac{F_{\mathbf{j}}}{2^{d_{\text{int}}}},$$

implying a uniform distribution of particles associated with a single grid-point $\mathbf{g_j}$ to its surrounding $2^{d_{\text{int}}}$ cells.

This result is somewhat surprising as it does neither rely on the size of the grid (the cell-width h) nor the index $\mathbf{j}$ of the cell in question. Further details can also be found in [2].

1.3.2.4 Multivariate Moments for the Pure Agglomeration Settings

This section is devoted to the analysis of moments $M_{\mathbf{e}}(f)(t)$ of a multivariate particle distribution $f(\mathbf{m})$ over time. For this, one obtains expressions to track any multivariate moment in the absence of breakage and growth and compares those values to numerical moments obtained over the course of a simulation using the discretization presented here.

Let $d_{\text{int}} = 2$ and denote the internal particle properties with $\mathbf{m} = (m_1, m_2)$. The change of a moment $M_{\mathbf{e}}(f)$ of a two-dimensional distribution $f(\mathbf{m})$ over time is given by

$$\begin{aligned} \frac{dM_{\mathbf{e}}(f)(t)}{dt} &= \iint_{\Omega_{\text{int}}} m_1^{e_1} m_2^{e_2} F_{\text{agg}}(f, \mathbf{m}) \ d\mathbf{m} \\ &= \frac{1}{2} \iint_{\Omega_{\text{int}}} m_1^{e_1} m_2^{e_2} \int_0^{m_1} \int_0^{m_2} \kappa_{\text{agg}}(\mathbf{m} - \mathbf{m}', \mathbf{m}') f(\mathbf{m} - \mathbf{m}') f(\mathbf{m}') \ d\mathbf{m}' \ d\mathbf{m} \\ &\quad - \iint_{\Omega_{\text{int}}} m_1^{e_1} m_2^{e_2} \iint_{\Omega_{\text{int}}} \kappa_{\text{agg}}(\mathbf{m}, \mathbf{m}') f(\mathbf{m}) f(\mathbf{m}') \ d\mathbf{m}' \ d\mathbf{m}. \end{aligned}$$

Setting $\kappa_{\text{agg}}(\mathbf{m}, \mathbf{m}') = 1$ eliminates the kernel from the equation. The domain of integration of the inner integral in the source term can be expanded to $[0, m_{\max}]^2$ by setting $f(\mathbf{m}-\mathbf{m}') := 0$ if any component of $\mathbf{m}-\mathbf{m}'$ is negative. A further change in the integration variable gives

$$\begin{aligned}\frac{dM_{\mathbf{e}}(f)(t)}{dt} &= \frac{1}{2} \iint_{\Omega_{\text{int}}} \iint_{\Omega_{\text{int}}} f(\mathbf{m}) f(\mathbf{m}') \cdot (m_1 + m_1')^{e_1} \cdot (m_2 + m_2')^{e_2} \, d\mathbf{m}' \, d\mathbf{m} \\ &\quad - \iint_{\Omega_{\text{int}}} \iint_{\Omega_{\text{int}}} m_1^{e_1} m_2^{e_2} f(\mathbf{m}) f(\mathbf{m}') \, d\mathbf{m}' \, d\mathbf{m}.\end{aligned}$$

The binomials in the first line are expanded in order to separate the integrations with respect to $\mathbf{m}$ and $\mathbf{m}'$ and then the order of summations and integrations is changed. A similar separation in the second line leads to

$$\begin{aligned}&\frac{dM_{\mathbf{e}}(f)(t)}{dt} \\ &\quad = \frac{1}{2} \sum_{k_1=0}^{e_1} \sum_{k_2=0}^{e_2} \binom{e_1}{k_1} \binom{e_2}{k_2} \iint_{\Omega_{\text{int}}} \mathbf{m}^{\mathbf{k}} f(\mathbf{m}) \, d\mathbf{m} \cdot \iint_{\Omega_{\text{int}}} (\mathbf{m}')^{\mathbf{e}-\mathbf{k}} f(\mathbf{m}') \, d\mathbf{m}' \\ &\quad - \iint_{\Omega_{\text{int}}} m_1^{e_1} m_2^{e_2} f(\mathbf{m}) \, d\mathbf{m} \cdot \iint_{\Omega_{\text{int}}} f(\mathbf{m}') \, d\mathbf{m}'.\end{aligned}$$

Every integral is in the form of (1.34) and is replaced accordingly. The rate of change of one moment then reads

$$\begin{aligned}\frac{dM_{\mathbf{e}}(f)(t)}{dt} &= \frac{1}{2} \sum_{k_1=0}^{e_1} \sum_{k_2=0}^{e_2} \binom{e_1}{k_1} \binom{e_2}{k_2} \cdot M_{(k_1,k_2)}(f)(t) \cdot M_{(e_1-k_1,e_2-k_2)}(f)(t) \\ &\quad - M_{\mathbf{e}}(f)(t) \cdot M_{(0,0)}(f)(t), \qquad (1.38)\end{aligned}$$

which is an ordinary differential equation in the moments, independent of the detailed particle distribution $f(\mathbf{m})$. With this, one can calculate the evolution of all moments given the moments of an initial distribution.

By using the moments of a discrete distribution $f(\mathbf{m})$ (as opposed to a continuous distribution) as initial values for (1.38), the only error present is due to the projection presented in subsection 1.3.2.3.

For the numerical simulation with $d_{\text{int}} = 2$, the initial distribution is discretized by

$$f(\mathbf{m}, 0) = N_0 e^{-200(m_1-0.1)^2} \cdot e^{-200(m_2-0.1)^2} \qquad (1.39)$$

with $n = 512$ for both internal coordinates with $m_{\max} = 10$. This results in $N = 512^2$ degrees of freedom and a width of $h = \frac{10}{512}$. The constant N_0 will be chosen such that $M_{(0,0)}(f) = 0.1$ at $t = 0$.

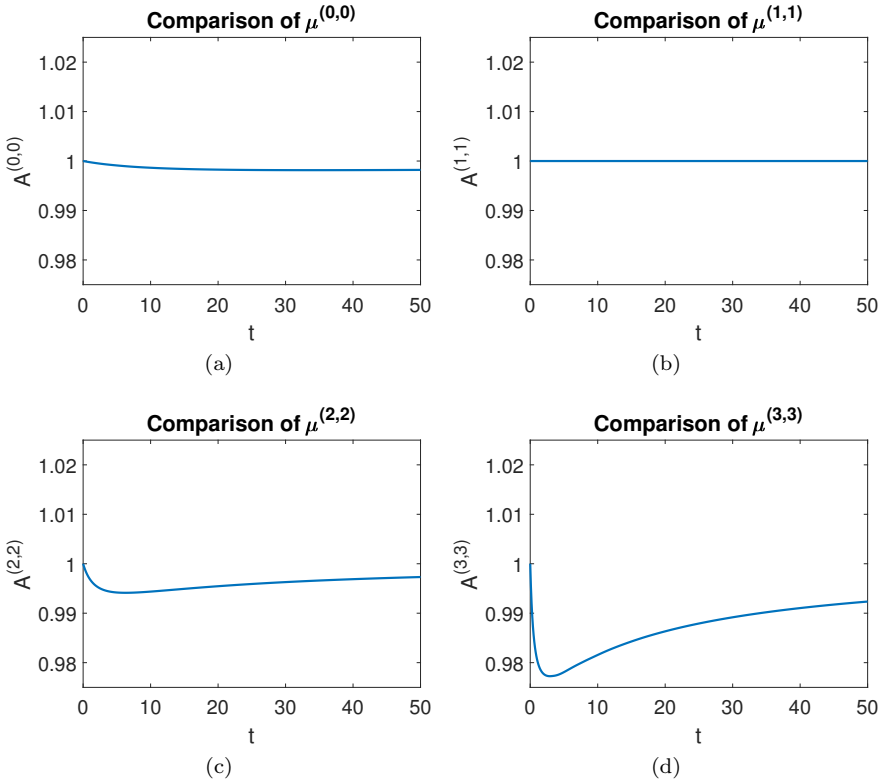

Fig. 1.17 $A^{\mathbf{e}}$ (1.40) over the course of a simulation for moments $M^{\mathbf{e}}$ with $\mathbf{e} = (0,0)$ (top left), $\mathbf{e} = (1,1)$ (top right), $\mathbf{e} = (2,2)$ (bottom left) and $\mathbf{e} = (3,3)$ (bottom right).

Numerical simulation up to $t = 50$ will result in $M_{(0,0)}(f)(50) = 0.0286$ at the end. In order to keep track of the numerical moments $\widetilde{M^{\mathbf{e}}}(f)(t)$ in the simulation, the ratio between numerical and theoretical moments,

$$A^{\mathbf{e}} = \widetilde{M^{\mathbf{e}}}(f)/M_{\mathbf{e}}(f), \tag{1.40}$$

is computed.
The ratio $A^{(0,0)}$ is shown in the top left of Figure 1.17 which displays a small loss of 0.18% (minimal ratio is $A^{(0,0)}(50) = 0.9982$) over time that is most likely caused by coarse time steps. There is no loss in the first cross moment as one finds $M_{(1,1)}(f) = 1$ for all t (shown in the top right of Figure 1.17). A smaller ratio occurs for higher order moments that are not square-free. It is not expected that $A^{\mathbf{e}}(t) = 1$ for all t since the moments are not preserved analytically. The ratio $A^{(2,2)}$ is shown in the bottom left of Figure 1.17 and indicates an under-prediction over the entire time period, similar to $A^{(3,3)}$ in the bottom right. The error does not exceed 0.6% and 2.3%, respectively.

Even though there is an error in the prediction of higher order moments, they are predicted with a very good accuracy given the grid with $N = 512^2$ degrees of freedom.

1.4 Summary and Outlook

The model experiment in the HCT crystallizer led to interesting and new results concerning the particle fluid interaction. Crystal size distributions change from the inlet to the outlet, already without growth, as crystals of varying size have a different residence time in the HCT. This newly found effect can be used to change the width of the CSD; or be used under growth conditions to keep the typically observed broadening of the CSD very low. Some first insights on the shape dependence of this effect are encouraging to investigate this in more detail in our future work.

For the numerical simulation of PBSs, a deterministic-stochastic algorithm was developed that enables the simulation of PBSs in rather complex spatial domains. Here, it was utilized for the simulation of a crystallization process in a fluidized bed crystallizer. To this end, two codes, one with deterministic finite element methods for the CFD equations and one with stochastic methods for the crystals, were coupled. Furthermore, the stochastic code was extended by all features that are due to the transport of the crystals in the three-dimensional domain.

Future work concerning the numerical simulations comprises algorithmic and modeling issues. From the algorithmic point of view, the code with the SPS methods should be MPI parallelized (the finite element CFD code is already) such that simulations of PBSs can be performed on clusters of computers to enhance the efficiency. From the point of view of the model, different agglomeration kernels and sedimentation models should be implemented and the breakage of crystals should be included. In addition, the SPS method should be extended to multi-variate crystals. This method provides a natural framework for the simulation of crystals that are modeled in a more complex way than with one internal coordinate.

The computational bottleneck of the expensive computation of agglomeration terms has been overcome by the exploitation of fast Fourier transformation (FFT) for the evaluation of convolution sums and tensor trains (TT) for the representation and arithmetic of multivariate density distributions. In a sense, numerical simulations have become feasible for a much higher resolution than experimental results are available. Future work might include the identification of models that include multivariate agglomeration, e. g. the determination of physically relevant agglomeration kernels. Another point of interest would be the extension of tensor trains to the entire population balance model, for example their adaptation to the breakage term. Future work could also focus on a parallelization of the introduced algorithms.

The agglomeration phenomenon in the fluidized bed crystallizer clearly shows the potential of such a device for a selective product removal, which is an important aspect for the post-processing of solids under industrial settings. As agglomeration can be modeled appropriately only with very specific parameters and kernels sets, the developed opportunity of big data acquisition within such a model experiment can be of great interest to the community of crystallization process engineers. In combination with the presented fast simulation methods for agglomeration and the corresponding fluid flow calculations, one may even start to optimize flow geometries in order to control the agglomeration in such crystallizers in the future.

Acknowledgements The work at this report was supported by the grants JO329/10-3, BO4141/1-3, and SU189/6-3 within the DFG priority programme 1679: Dynamic simulation of interconnected solid processes.

References

1. Robin Ahrens and Sabine Le Borne. FFT-based evaluation of multivariate aggregation integrals in population balance equations on uniform tensor grids. *Journal of Computational and Applied Mathematics*, 338:280–297, August 2018.
2. Robin Ahrens and Sabine Le Borne. Tensor trains and moment conservation for multivariate aggregation in population balance modeling. 2019. submitted.
3. Felix Anker, Sashikumaar Ganesan, Volker John, and Ellen Schmeyer. A comparative study of a direct discretization and an operator-splitting solver for population balance systems. *Computers & Chemical Engineering*, 75:95–104, 2015.
4. Clemens Bartsch. *A Coupled Stochastic-Deterministic Method for the Numerical Solution of Population Balance Systems*. PhD thesis, Freie Universität Berlin, Department of Mathematics and Computer Science, 2018.
5. Clemens Bartsch, Volker John, and Robert I.A. Patterson. Simulations of an ASA flow crystallizer with a coupled stochastic-deterministic approach. *Computers & Chemical Engineering*, 124:350–363, 2019.
6. Clemens Bartsch, Viktoria Wiedmeyer, Zahra Lakdawala, Robert I.A. Patterson, Andreas Voigt, Kai Sundmacher, and Volker John. Stochastic-deterministic population balance modeling and simulation of a fluidized bed crystallizer experiment. *Chemical Engineering Science*, 208:115102, 2019.
7. Howard C. Berg. *Random walks in biology*. Princeton University Press, Princeton, NJ, 1983.
8. G. A. Bird. Direct simulation and the Boltzmann equation. *Physics of Fluids*, 13(11):2676–2681, 1970.
9. Christian Borchert and Kai Sundmacher. Efficient formulation of crystal shape evolution equations. *Chemical Engineering Science*, 84:85–99, 2012.
10. Christian Borchert and Kai Sundmacher. Morphology evolution of crystal populations: Modeling and observation analysis. *Chemical Engineering Science*, 70:87–98, 2012. 4th International Conference on Population Balance Modeling.
11. Christian Borchert, Erik Temmel, Holger Eisenschmidt, Heike Lorenz, Andreas Seidel-Morgenstern, and Kai Sundmacher. Image-based in situ identification of face specific crystal growth rates from crystal populations. *Crystal Growth & Design*, 14(3):952–971, 2014.

12. Róbert Bordás, Volker John, Ellen Schmeyer, and Dominique Thévenin. Measurement and simulation of a droplet population in a turbulent flow field. *Computers & Fluids*, 66:52–62, 2012.
13. Róbert Bordás, Volker John, Ellen Schmeyer, and Dominique Thévenin. Numerical methods for the simulation of a coalescence-driven droplet size distribution. *Theoretical and Computational Fluid Dynamics*, 27(3-4):253–271, 2013.
14. Jay P. Boris and David L. Book. Flux-corrected transport. I: SHASTA, a fluid transport algorithm that works. *J. Comput. Phys.*, 11:38–69, 1973.
15. Sabine Le Borne, Holger Eisenschmidt, and Kai Sundmacher. Image-based analytical crystal shape computation exemplified for potassium dihydrogen phosphate (KDP). *Chemical Engineering Science*, 139:61–74, 2016.
16. Sabine Le Borne, Lusine Shahmuradyan, and Kai Sundmacher. Fast evaluation of univariate aggregation integrals on equidistant grids. *Computers & Chemical Engineering*, 74:115–127, 2015.
17. Allan S. Bramley, Michael J. Hounslow, and Rosemary L. Ryall. Aggregation during precipitation from solution: A method for extracting rates from experimental data. *Journal of Colloid and Interface Science*, 183(1):155–165, 1996.
18. A. Buffo, M. Vanni, D.L. Marchisio, and R.O. Fox. Multivariate quadrature-based moments methods for turbulent polydisperse gas–liquid systems. *International Journal of Multiphase Flow*, 50(0):41–57, 2013.
19. Jayanta Chakraborty, Jitendra Kumar, Mehakpreet Singh, Alan Mahoney, and Doraiswami Ramkrishna. Inverse problems in population balances. Determination of aggregation kernel by weighted residuals. *Industrial & Engineering Chemistry Research*, 54(42):10530–10538, 2015.
20. James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, 19(90):297–297, May 1965.
21. M. Dosta, E.-U. Hartge, R. Ahrens, S. Heinrich, and S. Le Borne. Investigation of an FFT-based solver applied to dynamic flowsheet simulation of agglomeration processes. *Adv. Powder Technol.*, 30:555–564, 2019.
22. Holger Eisenschmidt, Moudar Soumaya, Naim Bajcinca, Sabine Le Borne, and Kai Sundmacher. Estimation of aggregation kernels based on Laurent polynomial approximation. *Computers & Chemical Engineering*, 103:210–217, 2017.
23. Tiago Ferreira and Wayne Rasband. *ImageJ User GuideIJ 1.46r*, 2012.
24. S. Ganesan, V. John, G. Matthies, R. Meesala, S. Abdus, and U. Wilbrandt. An object oriented parallel finite element scheme for computing PDEs: Design and implementation. In *IEEE 23rd International Conference on High Performance Computing Workshops (HiPCW) Hyderabad*, pages 106–115. IEEE, 2016.
25. Sashikumaar Ganesan. An operator-splitting Galerkin/SUPG finite element method for population balance equations: stability and convergence. *ESAIM Math. Model. Numer. Anal.*, 46(6):1447–1465, 2012.
26. Sashikumaar Ganesan and Lutz Tobiska. An operator-splitting finite element method for the efficient parallel solution of multidimensional population balance systems. *Chemical Engineering Science*, 69(1):59–68, 2012.
27. Sashikumaar Ganesan and Lutz Tobiska. Operator-splitting finite element algorithms for computations of high-dimensional parabolic problems. *Appl. Math. Comput.*, 219(11):6182–6196, 2013.
28. Daniel T. Gillespie. The stochastic coalescence model for cloud droplet growth. *Journal of the Atmospheric Sciences*, 29(8):1496–1510, 1972.
29. Daniel T. Gillespie. An exact method for numerically simulating the stochastic coalescence process in a cloud. *Journal of the Atmospheric Sciences*, 32(10):1977–1989, 1975.
30. Wolfgang Hackbusch. On the efficient evaluation of coalescence integrals in population balance models. *Computing*, 78(2):145–159, 2006.

31. Wolfgang Hackbusch, Volker John, Aram Khachatryan, and Carina Suciu. A numerical method for the simulation of an aggregation-driven population balance system. *Internat. J. Numer. Methods Fluids*, 69(10):1646–1660, 2012.
32. Ami Harten, Björn Engquist, Stanley Osher, and Sukumar R. Chakravarthy. Uniformly high-order accurate essentially nonoscillatory schemes. III. *J. Comput. Phys.*, 71(2):231–303, 1987.
33. H.M. Hulburt and S. Katz. Some problems in particle technology: A statistical mechanical formulation. *Chemical Engineering Science*, 19(8):555–574, 1964.
34. V. John, I. Angelov, A.A. Öncül, and D. Thévenin. Techniques for the reconstruction of a distribution from a finite number of its moments. *Chem. Eng. Sci.*, 62(11):2890–2904, 2007.
35. Volker John. *Finite element methods for incompressible flow problems*, volume 51 of *Springer Series in Computational Mathematics*. Springer, Cham, 2016.
36. Volker John, Teodora Mitkova, Michael Roland, Kai Sundmacher, Lutz Tobiska, and Andreas Voigt. Simulations of population balance systems with one internal coordinate using finite element methods. *Chemical Engineering Science*, 64(4):733–741, 2009.
37. Volker John and Julia Novo. On (essentially) non-oscillatory discretizations of evolutionary convection-diffusion equations. *J. Comput. Phys.*, 231(4):1570–1586, 2012.
38. Volker John and Michael Roland. On the impact of the scheme for solving the higher dimensional equation in coupled population balance systems. *Internat. J. Numer. Methods Engrg.*, 82(11):1450–1474, 2010.
39. Volker John and Ellen Schmeyer. Finite element methods for time-dependent convection-diffusion-reaction equations with small diffusion. *Comput. Methods Appl. Mech. Engrg.*, 198(3-4):475–494, 2008.
40. Volker John and Carina Suciu. Direct discretizations of bi-variate population balance systems with finite difference schemes of different order. *Chemical Engineering Science*, 106:39–52, 2014.
41. Volker John and Ferdinand Thein. On the efficiency and robustness of the core routine of the quadrature method of moments (QMOM). *Chemical Engineering Science*, 75:327–333, 2012.
42. Dmitri Kuzmin. Explicit and implicit FEM-FCT algorithms with flux linearization. *J. Comput. Phys.*, 228(7):2517–2534, 2009.
43. Dmitri Kuzmin and Matthias Möller. Algebraic flux correction. I. Scalar conservation laws. In *Flux-corrected transport*, Sci. Comput., pages 155–206. Springer, Berlin, 2005.
44. Alison Lewis, Marcelo Seckler, Herman Kramer, and Gerda van Rosmalen. *Industrial Crystallization: Fundamentals and Applications*. Cambridge University Press, 2015.
45. Xu-Dong Liu, Stanley Osher, and Tony Chan. Weighted essentially non-oscillatory schemes. *J. Comput. Phys.*, 115(1):200–212, 1994.
46. Rainald Löhner, Ken Morgan, Jaime Peraire, and Mehdi Vahdati. Finite element flux-corrected transport (FEM-FCT) for the Euler and Navier-Stokes equations. *Int. J. Numer. Methods Fluids*, 7:1093–1109, 1987.
47. Daniele L. Marchisio and Rodney O. Fox. Solution of population balance equations using the direct quadrature method of moments. *Journal of Aerosol Science*, 36(1):43–73, 2005.
48. Daniele L. Marchisio, R. Dennis Vigil, and Rodney O. Fox. Quadrature method of moments for aggregation-breakage processes. *Journal of Colloid and Interface Science*, 258(2):322–334, 2003.
49. Robert McGraw. Description of aerosol dynamics by the quadrature method of moments. *Aerosol Science and Technology*, 27(2):255–265, 1997.
50. Henri J. Nussbaumer. *Fast Fourier Transform and Convolution Algorithms*. Springer Berlin Heidelberg, 1982.
51. David R. Ochsenbein, Thomas Vetter, Manfred Morari, and Marco Mazzotti. Agglomeration of needle-like crystals in suspension. ii. modeling. *Crystal Growth & Design*, 15(9):4296–4310, 2015.

52. Robert I. A. Patterson and Wolfgang Wagner. A stochastic weighted particle method for coagulation-advection problems. *SIAM J. Sci. Comput.*, 34(3):B290–B311, 2012.
53. Robert I. A. Patterson, Wolfgang Wagner, and Markus Kraft. Stochastic weighted particle methods for population balance equations. *J. Comput. Phys.*, 230(19):7456–7472, 2011.
54. Stephen B. Pope. *Turbulent flows*. Cambridge University Press, Cambridge, 2000.
55. Hans-Görg Roos, Martin Stynes, and Lutz Tobiska. *Robust numerical methods for singularly perturbed differential equations*, volume 24 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, second edition, 2008. Convection-diffusion-reaction and flow problems.
56. L. Shahmuradyan S. Le Borne. Algorithms for the Haar wavelet based fast evaluation of aggregation integrals in population balance equations. *Appl. Numer. Math.*, 108:1–20, 2016.
57. L. Shahmuradyan S. Le Borne. Fast algorithms for hp-discretized univariate population balance aggregation integrals. *Comput. Chem. Eng.*, 97:1–12, 2017.
58. Pierre Sagaut. *Large eddy simulation for incompressible flows*. Scientific Computation. Springer-Verlag, Berlin, third edition, 2006. An introduction, Translated from the 1998 French original, With forewords by Marcel Lesieur and Massimo Germano, With a foreword by Charles Meneveau.
59. D.L. Scharfetter and H.K. Gummel. Large signal analysis of a silicon Read diode. *IEEE Trans. Elec. Dev.*, 16:64–77, 1969.
60. Ellen Schmeyer, Róbert Bordás, Dominique Thévenin, and Volker John. Numerical simulations and measurements of a droplet size distribution in a turbulent vortex street. *Meteorologische Zeitschrift*, 23(4):387–396, 09 2014.
61. Stefan Schorsch, Jean-Hubert Hours, Thomas Vetter, Marco Mazzotti, and Colin N. Jones. An optimization-based approach to extract faceted crystal shapes from stereoscopic images. *Computers & Chemical Engineering*, 75:171–183, 2015.
62. Lusine Shahmuradyan. *Efficient and accurate evaluation of aggregation integrals in population balance equations*. PhD thesis, Hamburg University of Technology, Institute of Mathematics, 2016.
63. Hang Si. TetGen, a Delaunay-based quality tetrahedral mesh generator. *ACM Trans. Math. Software*, 41(2):Art. 11, 36, 2015.
64. Erik Temmel, Holger Eisenschmidt, Heike Lorenz, Kai Sundmacher, and Andreas Seidel-Morgenstern. A short-cut method for the quantification of crystallization kinetics. 1. Method development. *Cryst. Growth Des.*, 16(12):6743–6755, 2016.
65. Lisa-Marie Terdenge, Stefan Heisel, Gerhard Schembecker, and Kerstin Wohlgemuth. Agglomeration degree distribution as quality criterion to evaluate crystalline products. *Chemical Engineering Science*, 133:157–169, 2015. 19th International Symposium on Industrial Crystallization.
66. Viktoria Wiedmeyer, Felix Anker, Clemens Bartsch, Andreas Voigt, Volker John, and Kai Sundmacher. Continuous crystallization in a helically coiled flow tube: Analysis of flow field, residence time behavior, and crystal growth. *Industrial and Engineering Chemistry Research*, 56(13):3699–3712, 2017.
67. Viktoria Wiedmeyer, Andreas Voigt, and Kai Sundmacher. Crystal population growth in a continuous helically coiled flow tube crystallizer. *Chemical Engineering & Technology*, 40(9):1584–1590, 2017.
68. Ulrich Wilbrandt, Clemens Bartsch, Naveed Ahmed, Najib Alia, Felix Anker, Laura Blank, Alfonso Caiazzo, Sashikumaar Ganesan, Swetlana Giere, Gunar Matthies, Raviteja Meesala, Abdus Shamim, Jagannath Venkatesan, and Volker John. ParMooN—A modernized program package based on mapped finite elements. *Comput. Math. Appl.*, 74(1):74–88, 2017.
69. Harold A. Wright and Doraiswami Ramkrishna. Solutions of inverse problems in population balances — I. Aggregation kinetics. *Computers & Chemical Engineering*, 16(12):1019–1038, 1992.

70. Steven T. Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. *J. Comput. Phys.*, 31(3):335–362, 1979.
71. Eberhard Zeidler. *Springer-Handbuch der Mathematik I.* Springer Spektrum, Wiesbaden, 2013.