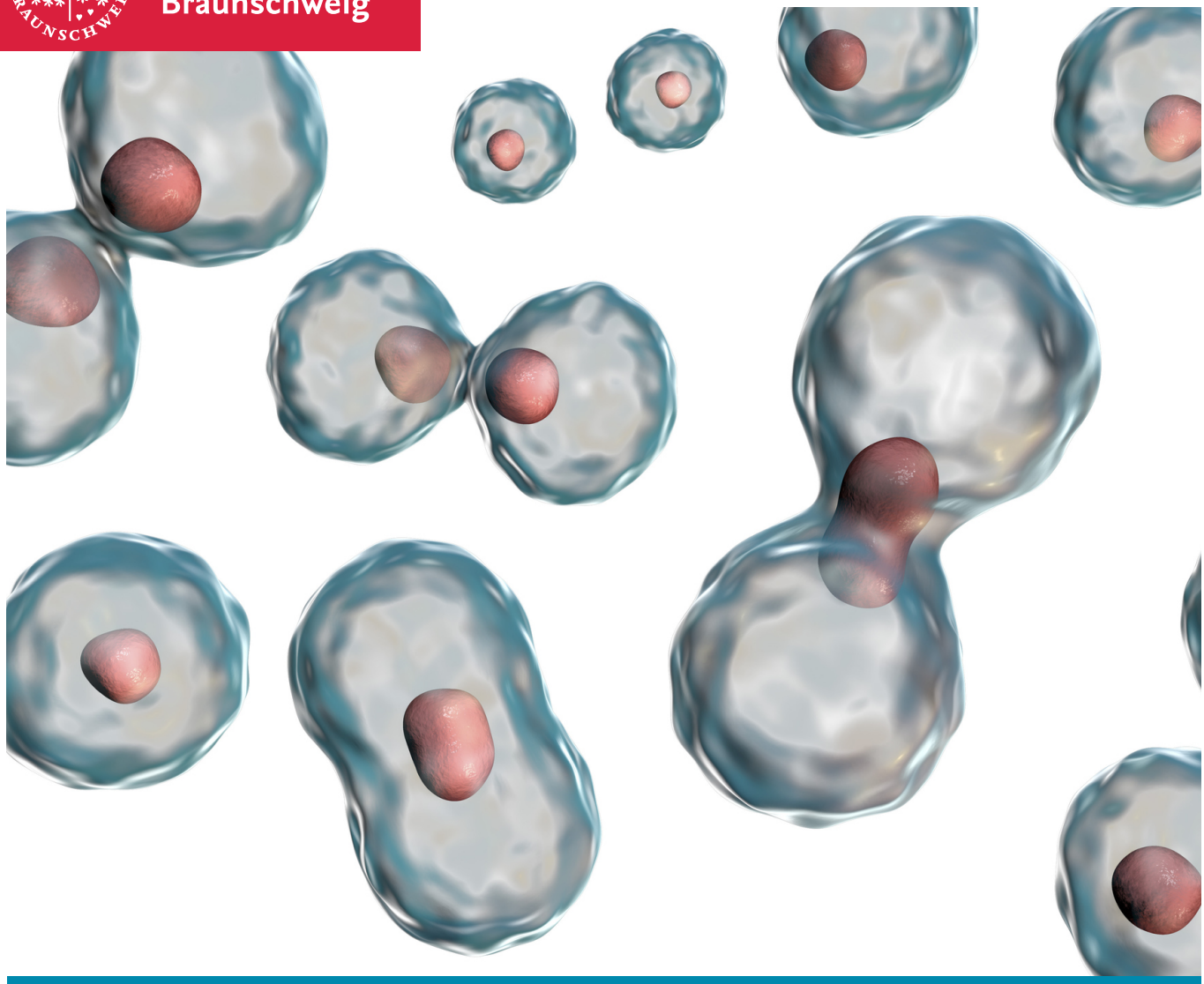
Technische Universität Braunschweig

IFM



Bachelor Thesis

On population balance modeling of cell growth and cell fission

Felix Köhler

November 29, 2018

Hiermit erkläre ich des Eides statt, dass ich die vorliegende Arbeit selbständig und ohne unerlaubte Hilfe angefertigt, andere als die angegebenen Quellen nicht benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Die Arbeit ist in gleicher oder ähnlicher Form noch nicht veröffentlicht und noch keiner Prüfungsbehörde vorgelegt worden.

Felix Köhler

Abstract

Most simulations in engineering encompass the use of discretizations to solve partial differential equations such as the Population Balance Equation that is in use in particle technology. A possible application of this equation is the prediction of cell proliferation. Due to its complexity, it requires the use of numerical methods. To improve the precision of the underlying solution scheme, an adaptive grid could be used that redistributes the mesh nodes over the process of the simulation. The choice of the monitor function that analyzes the current solution is crucial to the performance of such a scheme. In this work, we present an overview of the monitors used in the literature and derive a new one that is based on the idea of artificial diffusivity in TVD schemes found in finite volume flux approximation. The monitors are used within the framework of a novel explicit adaptive grid introduced by Sewerin and Rigopoulos (2017) for solving the population balance equation. Test cases of pure growth, pure coagulation and pure breakage as well as the simulation of cell dynamics are used to compare the monitors' performance. Therefore, a scheme for discretizing the contribution of breakage in the finite volume context is presented. Conclusions are drawn based on convergence properties and timing. In the end, one monitor was found to be superior in all tested scenarios.

Kurzzusammenfassung

Viele Simulationen in den Ingenieurswissenschaften erfordern die Lösung partieller Differentialgleichungen, wie z.B. auch die Population Balance Equation, welche sich zum Modellieren von Partikelverteilungen eignet. Sie findet unter anderem Anwendung in der Berechnung von Zellwachstum und Zellteilung. Da die zugrundeliegenden Mechanismen meist komplex sind, bedarf es der Anwendung von numerischen Methoden. Die dafür eingesetzten Werkzeuge basieren zum größten Teil auf der Diskretisierung des zugrundeliegenden Gebietes. Dieser Schritt könnte durch den Einsatz eines adaptiven Gitters optimiert werden, welches die Knotenpunkte über den Prozess der Simulation hinweg neu verteilt. Die Wahl der dabei genutzten Monitorfunktion, welche die aktuelle Lösung analysiert, hat einen großen Einfluss auf die Güte des eingesetzten Schemas. In dieser Bachelorarbeit wird ein Überblick an den in der Literatur zu findenden Monitorfunktionen gegeben. Darüber hinaus wird ein neuer Monitor hergeleitet, welcher auf der Idee der künstlichen Diffusivität basiert, die man in TVD-Schemata bei der Finite Volumen Fluss Approximation findet. Die Monitorfunktionen werden im Rahmen eines neuartigen explizit-adaptiven Gitters für die Lösung der Population Balance Equation verwendet, welches von Sewerin and Rigopoulos (2017) vorgeschlagen wurde. Im Rahmen der Lösung der Population Balance Equation werden Testfälle von reinem Wachstum, reiner Koagulation, reinem Zerbrechen sowie die Simulation von Zelldynamiken genutzt um die Fähigkeit der einzelnen Monitorfunktionen zu vergleichen. Dafür wurde ein Diskretisierungsschema für das Phänomen des Zerbrechens im Rahmen von Finite Volumen eingeführt. Ein Fazit wird dann basierend auf den Konvergenzeigenschaften und der Rechenzeit gezogen. Dabei erwies sich eine Monitorfunktion als überlegen in allen getesteten Fällen.

Preface

The following work came into existence within the 6th semester of my undergraduate studies in Mechanical Engineering. I was fascinated by higher applied mathematics and their use in simulations for engineering very early of the time at the university. Since this strengthened my interests in computer science I started to set the focus of my studies towards such topics. I chose the specialization of General Mechanical Engineering. For the future, I am looking forward to studying Computational Sciences and Engineering.

Finally, I am glad to be able to write on this topic. Over the course of this work, I learned a lot about programming and software engineering, especially in the usage of Linux and open source tools. Additionally, I could apply many mathematical concepts.

To this end, I want to thank Dr. Fabian Sewerin for his careful supervision and the time he took at our meetings and beyond.

Contents

List of Figures

List of Tables

1 Introduction

Over the last decades, the main challenges of engineering have been shifting. Systems increase in complexity and it becomes more and more the objective to not invent but further optimize structures and processes. This often requires intensive calculations to quantitatively decide between given designs. The invention of computers with their ever-improving performance and the progress in numerical mathematics hand a powerful tool to engineers. By them, they are capable of performing extensive simulations aiding in the process of research, design, maintenance and many more.

The actual numerical solution schemes play an important role in these aspects. They decide whether a simulation is worthwhile or if the costs do not outweigh the outcomes. Therefore, it is crucial to have techniques that are as efficient as possible while still yielding highly accurate results.

The topic of Population Balance Equations (PBE) presents such a case in which simulations are necessary. This equation allows for the modeling of particle evolution that is important, e.g., in calculating the proliferation of cell cultures. Predicting the state of cell cultures could help in modeling constitutive relations of biological tissue or improving in-vitro synthesization. This knowledge can then be used in medical engineering for creating prosthesis or artificial cultivation of stem cells to replacements organs for the human body that are then re-transplanted. In order to satisfy the global demand for meat products, in-vitro products might also be of interest which can help to reduce the livestock.

Aside from the potential biological applications of the PBE, this equation represents a versatile tool which is also found in other disciplines of science and engineering, e.g., the design of chemical reactors, atmospheric weather prediction or even astrophysics and many more (see Ramkrishna and Singh (2014) for a general review). Another optimization task is the development of clean combustion engines. Therefore, the PBE yields a powerful tool in predicting the formation and the state of soot or other pollutants in turbulent combustion processes.

Generally speaking, the task of developing tools to solve the population balance equations for the given cases is of interest for a wide range of applications. Most of the tools that are applicable in the biological context can then also be transferred to, e.g., reactor design.

Similar to other partial differential equations employed in engineering, solution methods more or less depend on discretizing the computational domains. Almost every time such a discretization is used it creates errors to the solution which can be reduced by refining the underlying grid. However, the task of mesh refinement is difficult and often requires manual intervention. Not to mention that it can also increase the computation time. Therefore, it might be helpful to adaptively improve the grid whenever and wherever it is necessary. That shows to be beneficial since, in the best case, the adaptive numerical solution scheme has higher accuracy while using less time to compute compared to a conventional scheme.

However, there is a downside. There are quite a few proposals of adaptive schemes but, naturally, all of them require some sort of measure to automatically analyze the current solution. We call this

relevant module a monitor. Until now, the choice of the monitor is problem dependent, explaining why adaptive grids are less in use than they could actually be. But the potential of their wider application might be immense allowing for more detailed simulations at reasonable costs.

This bachelor thesis is dedicated to the comparison of different monitor functions in the context of the population balance equation for cell proliferation. Hereby, we rely on a novel explicit adaptive grid scheme that was proposed by Sewerin and Rigopoulos (2017). Our scientific contribution is fourfold:

- Review of monitor functions used in the literature
- Derivations of a novel monitor
- Development of a finite volume scheme to cover the contribution of breakage within the PBE
- Presentation and comparison of the monitors in test cases and the real world example of cell proliferation

The necessary simulations are performed by the help of the fluid flow solver BOFFIN (BOunded Flow Fitted INtegrator) (Jones 2002) that has been modified to work with the explicit-adaptive grid in the context of the population balance equation. All routines are implemented within the Fortran programming language and compiled by the help of the open source gfortran compiler.

The thesis is structured as follows. The first chapter will define the population balance equation mathematically. Therefore all important phenomena are introduced. Next, the second chapter is dedicated to the explicit adaptive grid which is rederived after a short literature review of solution methods to the PBE. We then present the novel monitor function alongside with an overview of those found in the literature. The chapter closes with a comparison in two test cases. The last third of this work is on to the modeling of cell proliferation. The mathematical treatment is fully shown followed by the validation on a third test case. Then, the kinetics behind cell proliferation are presented. Finally, the results are shown and a conclusion is drawn based upon the possibly improved accuracy with the help of the monitors and their influence on computational cost.

2 Population Balance Equation

The description of the evolution of particles is important for many aspects of engineering. It can be found in droplet creation in atmospheric clouds, soot formation in combustion processes, crystallization within chemical reactors or growth and proliferation of biological entities. These topics share the common idea to not only predict the spatial location $\mathbf{x}$ of a certain particle at a given time t but also its respective properties. Some of them might be the characteristic length of a crystal, the volume of the particle, the mass or maturity of cells in biological organisms.

One possibility to address this was first done by Hulburt and Katz (1964). They introduced the so-called internal coordinates $\mathbf{v}$ to describe different properties. To predict the evolution of a certain particle one now has to find a probability density function $n(t, \mathbf{x}, \mathbf{v})$ which states how likely it is that the particle is at the spatial location $\mathbf{x}$ at a given time t while having properties according to $\mathbf{v}$. For a macroscopic population consisting of multiple individual particles, this unknown function becomes a number density which has the unit of *Number of Particles per Volume*, i.e., $[1/m^3]$.

Population Balance equations now come into perspective at counting and balancing for the evolution of particles. Many different phenomena can be occurring simultaneously. Fig. 2.1 summarizes them.

In this chapter, the population balance equation is derived based on the idea of local conservation law, i.e., the continuity equation. Additionally, the second section captures those phenomena that are not conservative and hence are accounted for by a source term.

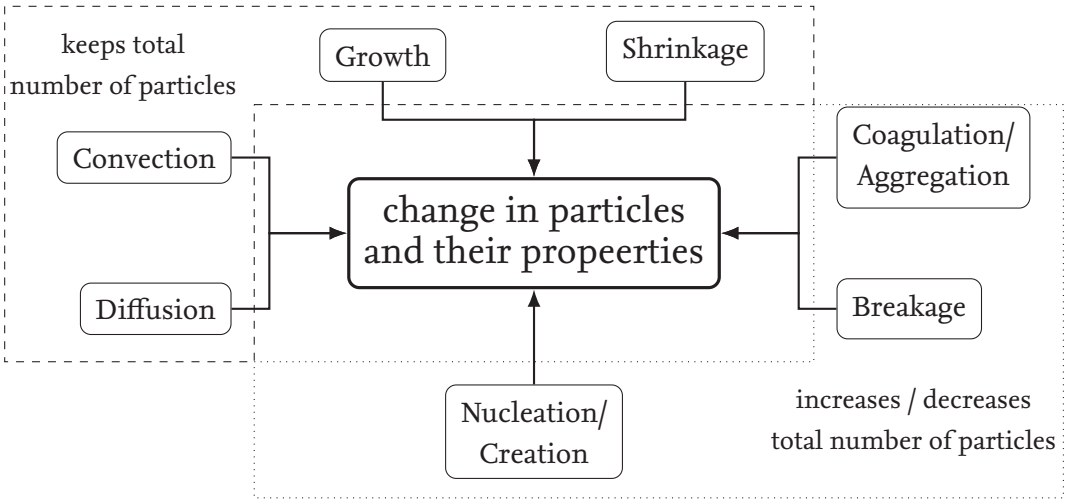


Figure 2.1: Potential phenomena occurring that the population balance equation can take care of. Convection and Diffusion refer to a movement in spatial location $\mathbf{x}$ whereas the other phenomena account for a change in internal properties $\mathbf{v}$.

2.1 Conservation Principles on Particle Property Spaces

With the phenomena in mind, it is now the task to model all the different aspects. To this end, we introduce two domains that are subsets of the external physical space and the internal particle property space. The physical space is, generally speaking, three-dimensional whereas the dimension of the internal space d depends on the number of properties considered.

$$\Omega_x \subset \mathcal{R}^3 \quad \text{and} \quad \Omega_v \subset \mathcal{R}^d \tag{2.1}$$

The total spatial dimension is therefor $3+d$, rendering general problems difficult to visualize.

The accounting for a change in number density can now be done by a local conservation law with an added source term $\dot{s}$. We decide to view this equation in Eulerian perspective for later numerical derivations.

$$\frac{\partial n}{\partial t} + \nabla_x \cdot \mathbf{j}_x + \nabla_v \cdot \mathbf{j}_v = \dot{s} \quad \mathbf{x} \in \Omega_x,\ \mathbf{v} \in \Omega_v,\ t > 0 \tag{2.2}$$

Here, we introduce the flux $\mathbf{j}$ that is a $(3+d)$-dimensional vector field incorporating the conservative change in particles and their properties meaning that it will be unable to increase or decrease the total amount of particles. Hence it can only be used for accounting of the phenomena in the top left of Fig. 2.1. A general flux consists of a convective/advective part and a diffusive part. Whereas both occur in the real physical space, only advection will be considered in the particle property space. In this case, it is just an analogy of particles moving along the internal axis that, e.g., causes an increase in their characteristic length, hence it refers to growth.

In most applications it is feasible to reduce the dimensions of the particle property space to $d = 1$. Therefore the flux component in particle property space will only consist of one entry, consequentially the divergence reduces to a simple partial derivative. If we further restrict the convection in physical space to simple advection alongside a fluid flow with the velocity field $\mathbf{u}(\mathbf{x}, t)$ we find the general population balance equation

$$\begin{aligned}\frac{\partial n(v,\mathbf{x},t)}{\partial t} + \sum_j^3 \frac{\partial (u_j(\mathbf{x},t) n(v,\mathbf{x},t))}{\partial x_j} + \frac{\partial (G(v,\mathbf{y}(\mathbf{x},t)) n(v,\mathbf{x},t))}{\partial v} = \\ \sum_j^3 \frac{\partial}{\partial x_j}\left(D_{ij}(v,\mathbf{x},t) \frac{\partial n(v,\mathbf{x},t)}{\partial x_i} \right) + \dot{s}(v,\mathbf{y}(\mathbf{x},t), n(v,t))\end{aligned} \tag{2.3}$$

The first term on the left-hand side is the accumulation term that accounts for a change in time of the particle density function keeping spatial location and internal coordinates fixed. The next part is the physical divergence describing the physical advection of particles by the velocity field $\mathbf{u}(t, \mathbf{x})$ that is created by the fluid flow the particles are immersed in. In the context of solving the PBE, the velocity field is assumed to be known. Separate computational fluid dynamics may be necessary. The next term is dedicated to the advection in particle property space. If the single internal coordinate is a characteristic length of the particle $v = L$ then $G(\mathbf{x}, v)$ becomes the growth rate in $[m/s]$. Using different internal coordinates will obviously change the units of the growth term. The first term on the right-hand side accounts for the diffusion in physical space with respective diffusivity $\mathbf{D}(t, \mathbf{x}, v)$. The constitutive parameters generally differ depending on the directions of diffusion hence the coefficient $\mathbf{D}$ becomes a matrix. Finally, $\dot{s}(t, \mathbf{x}, v)$ is the source or reaction term. It accounts for

all the events that are not conservative, i.e., increase or decrease the total particle count. They are summarized in the bottom right of Fig. 2.1 and are introduced mathematically in the next section.

For real modeling, we have to additionally consider values that affect the kinetic coefficients in particle property space, i.e., the growth rate G and the kernel functions within the source term $\dot{s}$. They might include a concentration of nutrients that are crucial for growth, e.g., proteins for cell growth or supersaturation of the bulk solution responsible for bringing fresh atoms to crystal growth. One could also save the temperature or pressure in here, depending on the case studied. These values are stored in the fluid flow composition vector $\mathbf{y}$. Naturally, the components can change over time t and external coordinate $\mathbf{x}$, hence its dependencies. In general, every quantity inside $\mathbf{y}$ is governed by its own convection-diffusion-reaction equation depending on the concrete modeling.

In conclusion, to yield a full initial boundary value problem (IBVP) one has to add an initial distribution of particles and boundary conditions for the real external physical domain Ω_x as well as for the internal particle property domain Ω_v. Since the internal domain is one-dimensional it simply becomes an interval that is generally spanned from zero to an upper threshold value that the particles will not exceed V.

$$\mathbf{v} := v \in \Omega_v = [0, V] \tag{2.4}$$

The internal value on the left side can then be set to a homogeneous Dirichlet boundary condition due to the fact that no particle of zero length/mass exists. For the external coordinates, the choice of boundary conditions generally depends on the ones used to calculate the flow field.

However, for many applications, we can omit the dependency on external coordinates $\mathbf{x}$ if the case is assumed to be spatially homogeneous in physical space. To a good approximation, this can be found in reactors that are in use for biological and chemical process engineering which are assumed to be perfectly stirred. But there are also other modeling cases that strongly depend on physical coordinate $\mathbf{x}$ such as turbulent combustion like in Sewerin and Rigopoulos (2018).

If we now reduce the dependencies $n(v, \mathbf{x}, t) \to n(v, t)$, the physical advection and diffusion of the general PBE of (2.3) vanish. The equation becomes

$$\frac{\partial n(v,t)}{\partial t} + \frac{\partial (G(v, \mathbf{y}(t)) n(v,t))}{\partial v} = \dot{s}(v, \mathbf{y}(t), n(v,t)) \quad v \in [0, V],\ t > 0. \tag{2.5}$$

This work is focused on this equation that we will later refer to as PBE. As discussed above the population balance equation is equipped with a (general) initial condition

$$n(v, 0) := n_0(v) \quad v \in [0, V] \tag{2.6}$$

and, for example, a homogeneous Dirichlet boundary condition

$$n(0, t) = 0 \quad t > 0. \tag{2.7}$$

A common measure to compare the results of the population balance equation is the use of moments

$$\Psi_k(t) = \int_0^\infty v^k n(v,t)\, \mathrm{d}v. \tag{2.8}$$

The zeroth is simply a summing up of all particles along the internal coordinate v. Hence it is the total number of particles. The first moment weights the integrand with the current position

on the internal coordinate. If we consider $v = [m^3]$ (the internal coordinate equals the particle volume), then $\Psi_1(t)$ describes the total volume all the particles occupy. One can also find physical interpretations for higher order moments. In general, the moment index does not have to be an integer.

2.2 Reaction Terms of the PBE

Since growth is naturally incorporated in the advection term inside the PBE, it now follows to describe the phenomena appearing within the inhomogeneous source or reaction term. The right-hand side (as informally considered in the bottom right of Fig. 2.1) will be responsible for a change in total particle count, i.e. the zeroth moment of the number density distribution. Generally speaking a change in total particle number is caused by addition or subtraction

$$\dot{s}(v, \mathbf{y}, n(v,t)) = \mathcal{B}(v, \mathbf{y}, n(v,t)) + \mathcal{D}(v, \mathbf{y}, n(v,t)). \tag{2.9}$$

These two aspects, namely Birth and Death of particles, are modeled differently for each of the three phenomena depending on the concrete physical background. In the following, the most widely used forms in the literature will be presented. Here we do not keep the restriction on the particle property domain and let it go from zero to infinity.

- **Breakage**: A particle splits into fractions of itself. Naturally, this will conserve the mass and hence does not result in a change of the first moment. Based on the number of particles created by breakage, one divides between the general and binary case. Parting into more than two particles may appear in chemical processes while a famous example of binary breakage would be cell fission. For this simpler case, the contribution to the source term reads (Ramkrishna 2000)

$$\dot{s}_{breakage}(v, \mathbf{y}, n(v,t)) = \underbrace{2 \int_v^\infty n(v',t)\gamma(v', \mathbf{y})p(v,v') \, \mathrm{d}v'}_{\mathcal{B}_{breakage}} - \underbrace{n(v,t)\gamma(v, \mathbf{y})}_{\mathcal{D}_{breakage}}. \tag{2.10}$$

  The first term accounts for the birth of two daughter particles. In order to model breakage, a division rate $\gamma(v', \mathbf{y})$ is needed. Based on the current state of the composition $\mathbf{y}(t)$, this component will express the likelihood of a mother particle of state v' to split. For instance, in the context of cell populations, older cells in more nutrient-rich environments are more likely to commit fission. Consequentially, $p(v,v')$ describes the probability of the mother particle to divide into a particle of coordinate v and hence into another particle of state $(v'-v)$ due to binary breakage. In the case of an equal distribution of mass $p(v,v')$ will become the Dirac delta distribution $\delta(v-(v'/2))$ centered at $v'/2$. Combined with the number density of mother particles at state v' and integrated over all possible states of the mother particle, this yields the net source of particles.

  Additionally, breakage reduces the particle count at higher coordinates because the mother particles disappear. This idea is covered by the second term accounting for the net sink of particles.

- **Coagulation**: Particles combine to create a new one. Similar to breakage this will conserve the mass and the first moment. Different varieties of coagulation exist depending on the

(a) Advection (b) Breakage (c) Coagulation

Figure 2.2: Visualization of the phenomena in population balance modeling. Nucleation is not to be seen.

shape of the newly formed particle. The following equation is generally valid for coalescence in which the basic shape of the resulting particle is similar to those of the combined, hence the definition of a characteristic length will be kept the same. Restricting ourselves to binary coalescence, the contribution to the source term would be (Ramkrishna 2000)

$$\dot{s}_{coalescence}(v, \mathbf{y}, n(v,t)) = \underbrace{\frac{1}{2}\int_0^v n(v',t)a(v',v-v')n(v-v',t)\,\mathrm{d}v'}_{\mathcal{B}_{coalescence}} - \underbrace{\int_0^\infty n(v',t)a(v',v)n(v,t)\,\mathrm{d}v'}_{\mathcal{D}_{coalescence}}. \tag{2.11}$$

Similar to breakage, there is a kernel function employed, namely the coalescence rate $a(v, v')$. It will examine the probability of two particles of state v and v' to merge. Hence, the second term describes the contribution of death for a particle of a given state v when coagulation with particles at all other possible states $v' \in (0, \infty)$. On the other hand, the Birth term will account for the newly created particle of state v that comes into existence when v' merges with $(v - v')$. By integrating over $[0, v]$ this will incorporate all possible combinations with their respective probability.

- **Nucleation**: In the context of crystallization, particles may precipitate out of a supersaturated solution. Initially, this leads to particles of internal coordinate close to zero (their size is small or their maturity is zero). In general, there is a range of internal coordinate over which new particles are created. If we assume that particles will always nucleate at a certain coordinate v_0, birth due to nucleation can be formulated as a point-wise realization of probability law by the help of a Dirac Delta Distribution

$$\dot{s}(v, \mathbf{y}, n(v,t)) = \mathcal{B}_{nucleation} = B_0(\mathbf{y})\delta(v - v_0). \tag{2.12}$$

The mechanisms of nucleation can be further categorized. The one seen above is called primary nucleation because of the nucleation rate B_0 highly depending on supersaturation (a potential component of $\mathbf{y}(t)$). Here particles immerse from themselves. Given external assistance, there are obviously more ways (e.g., impellers, pollutants) to the creation of new particles. Those phenomena fall under the category of secondary nucleation and are often modeled differently. A broad overview is found in Omar and Rohani (2017). Naturally, nucleation leads to an increase in the zeroth and the first moment which is caused by the addition of particles and the introduction of new mass.

If the domain is restricted to $\Omega_v = [v_o, V]$, then nucleation can also be modeled by an inflow boundary condition.

3 Adaptive Mesh Discretization and Monitor Functions

Since the first appearance of the Population Balance Equation in the work of Hulburt and Katz (1964) many methods have been developed to solve this equation. Due to their integro-differential nature analytical solutions only exist in simplified cases with basic kinetics. If the PBE becomes homogeneous, i.e., the reaction terms are neglected, then solutions can be found via the method of characteristics because the equation reduces to the case of simple advection.

However, as the problem size and complexity increases, numerical methods need to be used. In the following, the most widely used ones are shortly introduced. They present the frame of reference our method has to compete with. We will not consider Monte-Carlo-based methods since they are fundamentally different from the scope of this work. For a more comprehensive review of the different methods available, the reader is referred to Ramkrishna (2000), or more recently Rigopoulos (2010).

- **Method of Moments**: When solving for the PBE, one is looking for a solution that depends on time t and internal coordinate v that represents a distribution of particles. However, there are times when the only matter of interest is total particle count, total particle mass or any other moment. Then it is unnecessary to compute a complete solution to the partial differential equation. A transformation of the PBE would yield a set of Ordinary Differential Equations (ODE) that account for the evolution of the different moments over time. This method was first introduced by Hulburt and Katz (1964). Based on this idea, many advanced techniques have been developed. A recent overview is found in Omar and Rohani (2017). On the upside, the methods of moments are cheaper to compute since only a few ODEs have to be solved. However, it is impossible to accurately resemble the underlying number density distribution without any further information. Additionally, the resulting set of ODEs is often unclosed.

- **Discretized Population Balance/Pivot Technique/Cell Averaging Technique**: In a series of three articles Kumar and Ramkrishna (1996a,b, 1997) proposed a method that discretizes the computational domain and selects a representative point for all of the created bins, called a pivot point. The solution concept presented in these articles has also been extended by the idea of a moving grid. According to Kumar et al. (2009), this technique shows high accuracy at correctly predicting the moments but the reconstructed distribution is worse compared to other methods.

- **Direct Discretization (FE, FV)**: As in many other disciplines that incorporate partial differential equations, direct discretization on fixed grids is a common way to solve the population balance equation. Either by approximating derivatives on grid nodes (Finite Difference - FD), approximating the solution with local shape functions (Finite Element - FE) or the averaging

of the solution on discretized cells (Finite Volume - FV), these are backed with various research background.

- **Finite Element**: The article of Rigopoulos and Jones (2003) presents a good overview of the applicability of Finite-Element schemes.
- **Finite Volume**: Aside from the classical implementation as seen in Qamar et al. (2006), it is possible to transform the PBE into a form of mass conservation (Filbet and Laurençot 2004). This brings the advantage of being able to naturally capture the source terms of breakage and coagulation (Qamar et al. 2009). A comparison with the cell averaging technique has been conducted by Kumar et al. (2009). The authors concluded that this method is favorable when an accurately resembled distribution is needed and an only moderate capturing of the moments is acceptable.

Any method that discretizes a computational domain inevitably introduces an error. If a given approximation converges, the accuracy of the solution improves at finer grids. However, more grid nodes increase the time it takes to compute the numerical solution. A compromise would be to refine the grid only locally in regions where it is necessary. These regions might comprise high solution gradients or sharp corners, i.e., near discontinuities that the solution procedure has to capture.

This chapter is dedicated to those kinds of methods. Especially, we derive an explicit adaptive grid scheme for the Finite Volume discretization and test its performance with the help of different monitor functions.

3.1 Mesh Redistribution Techniques

Static or fixed uniform grid discretization schemes bring the disadvantage of having a mesh that is, in the worst case, insufficiently fine in regions where the solution has steep gradients and unnecessarily dense in smooth regions. Using non-uniform grids, we are able to increase the accuracy of the solution by keeping the total amount of grid nodes constant. This is a common procedure in structural mechanics or fluid mechanics since there are certain regions of interest, e.g., the boundary layer around a wing. Often this incorporates a (manual) process of refining the grid in advance of the simulation. However, not all simulation cases allow for this. For the solution of the PBE, there are generally no special regions of interest in the particle property space.

A next step would be the use of moving grid schemes. These techniques are based on the method of characteristics and exploit the natural transport of information in hyperbolic conservation equation (like the PBE). Here, we can find the relation between space and time, the so-called characteristic lines. Then the fine grid regions are simply moved alongside those lines. Kumar and Ramkrishna (1996b) used this moving idea to further improve their pivot technique. In general, these methods advance the idea of direct discretization. However, they show problems when it comes to sudden changes in the solution. This makes them hard to apply to inhomogeneous cases (with the reaction terms of the PBE active), since that will interfere with the movement of the characteristics through space-time.

If the nodes are now redistributed based on the current solution, this is called an adaptive grid since the mesh will be adapted. Adaptivity requires the introduction of a measure to detect the regions where a finer grid is necessary. The choice of this so-called monitor is crucial for correctly

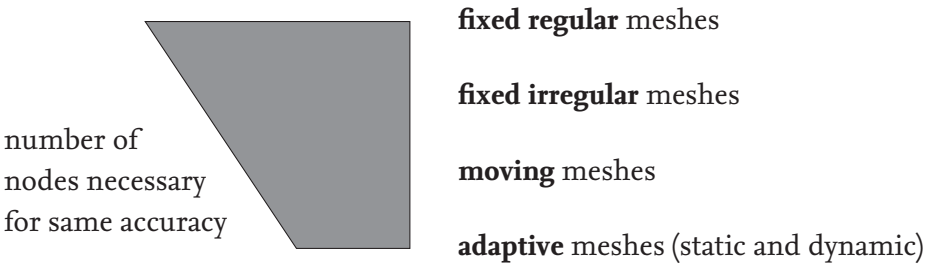


Figure 3.1: Informal visualization of the logical transition to adaptive grid schemes.

analyzing the solution. In section 3.3 we give a summary of the monitors available.

Over the broad range of adaptive grid methods, a further differentiation is made. Static schemes such as the two-step one of Tang and Tang (2003) evolve the solution to the partial differential equation separately from the change that is made to the grid. After a given time the solution is then interpolated onto the new grid.

By contrast, dynamic adaptive grids do not need those possibly very intensive interpolation procedures. Here, a change in the solution will immediately change the grid equation. The moving mesh partial differential equation adapted to the context of the PBE by Lim et al. (2001) belongs to this category.

Informally speaking, the process up to adaptive grid schemes presents a transition to what reduces the number of nodes necessary to achieve the same accuracy. Fig. 3.1 shows a visualization of this idea. However, to further compare the usefulness the inevitable increase in computation time has to be considered. The application and implementation of an adaptive grid is beneficial when the accuracy increases or the simulation time decreases.

3.2 An Explicit Adaptive Grid

The previously presented adaptive grid methods rely on solving a separate equation alongside with the actual PDE to describe the evolution of the nodes. In this section, we use the approach of an explicit adaptive grid that directly describes the adaptive grid. Therefore a second virtual domain is created that is always uniform and on which the solution is discretized. [1] The real domain is then subject to change under the influence of the monitor function. For any further calculations that rely on the real grid, this information can simply be substituted by the grid node distance of the current adaptive grid. As a connection between the two domains, a coordinate transformation is introduced, that maps from the virtual domain to the real domain.

The new domain is addressed by the coordinate τ. Hence the task of the following subsections is to transform the population balance equation from its v-dependent form into an equation consisting of derivatives with respect to τ.

Sewerin and Rigopoulos (2017) introduced this concept in the context of the population balance equation. The following derivations are loosely based on this article. In the continuing simulation, already implemented code in the solver BOFFIN (bounded flow fitted integrator) was used.

[1] Since we omitted the dependency on any spatial location in (2.3), speaking of a domain always refers to the domain in the particle property space. In the one-dimensional case this simply becomes an interval $v \in \Omega_v = [0, V]$.

3.2.1 The transformed PBE

In this subsection, the population balance equation will be transformed from the real v-space into the virtual τ-space. Therefore we start off with the general PBE of the previous chapter

$$\frac{\partial n}{\partial t} + \frac{\partial (G(v, \mathbf{y}, t) n)}{\partial v} = \dot{s}(v, \mathbf{y}, n(\cdot)). \tag{3.1}$$

Again, we restrict the domain to $\Omega_v = [0, V]$ by choosing an upper limit V that no particle will pass. However, the domain can be any interval of arbitrary position and length. Parallel to this domain, another domain Ω_τ is created that shares the same boundaries as Ω_v. To map between those instances, a C^1-continuous coordinate transformation is introduced

$$\bar{v} : \tau \to v = \bar{v}(\tau, t) \quad \text{with} \quad \bar{v} : [0, V] \times \Re_0^+ \to [0, V]. \tag{3.2}$$

In the following, we assume the transformation is known. Section 2.2.3 will be covering the details on prescribing such a transformation.

Three constraints are set to assure the mapping is bijective. First, the Jacobian, a measure of distortion, is set to be greater than zero. The second and third constraints force the boundary points of the domain to be transformed onto each other. As a consequence the entire length of the domain will be kept constant.

$$\frac{\partial \bar{v}(\tau, t)}{\partial \tau} := w(\tau, t) > 0 \quad \forall t \geq 0, \tag{3.3}$$

$$\bar{v}(0, t) = 0 \quad \forall t \geq 0, \tag{3.4}$$

$$\bar{v}(V, t) = \int_0^V w(\tau, t) \, \mathrm{d}\tau = V \quad \forall t \geq 0 \tag{3.5}$$

Since the mapping is bijective there is an inverse transformation $\bar{\tau} := \bar{v}^{-1}$ mapping from the real internal coordinate v to the virtual coordinate τ. Based on this, the point-wise identity holds.

$$\bar{\tau}(\bar{v}(\tau, t), t) = \tau \quad \forall \tau \in [0, V], t \geq 0 \tag{3.6}$$

For the following derivations some relations are helpful that are mainly based on the identities

$$\left.\frac{\partial \bar{\tau}}{\partial v}\right|_{\bar{v}} = \frac{1}{w}, \tag{3.7}$$

$$\left.\frac{\partial \bar{\tau}}{\partial t}\right|_{\bar{v}} = \underbrace{\frac{\mathrm{d}\bar{v}}{\mathrm{d}t}}_{=0} - \underbrace{\left.\frac{\partial \bar{\tau}}{\partial v}\right|_{\bar{v}}}_{(3.7)} \frac{\partial \bar{v}}{\partial t} = -\frac{1}{w}\frac{\partial \bar{v}}{\partial t}, \tag{3.8}$$

$$\frac{\partial (1/w)}{\partial \tau} = -\frac{1}{w^2}\frac{\partial w}{\partial \tau} = -\frac{1}{w^2}\frac{\partial^2 \bar{v}}{\partial \tau^2}, \tag{3.9}$$

$$\frac{\mathrm{d}G|_{\bar{v}}}{\mathrm{d}\tau} = \left.\frac{\partial G}{\partial v}\right|_{\bar{v}} \underbrace{\frac{\partial \bar{v}}{\partial \tau}}_{(3.3)} = w \left.\frac{\partial G}{\partial v}\right|_{\bar{v}}. \tag{3.10}$$

For convenience, one now introduces a transformed number density that is defined on the virtual coordinate τ. The transformed PBE will then be a partial differential equation for this function

$$n(v, t) := f(\bar{\tau}(v, t), t). \tag{3.11}$$

At this point all the information about the transformation has been collected. In a next step, the terms of (3.1) will be evaluated at $v = \bar{v}(\tau, t)$ which means the new equation is formulated depending on τ, the coordinate of the transformed space. In the source term, one can simply change v for $\bar{v}$ and n for f. For the accumulation and advection term we get

$$\left.\frac{\partial n}{\partial t}\right|_{\bar{v}} = \frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\partial f}{\partial \tau}\underbrace{\left.\frac{\partial \bar{\tau}}{\partial t}\right|_{\bar{v}}}_{(3.8)} + \frac{\partial f}{\partial t} = -\frac{1}{w}\frac{\partial f}{\partial \tau}\frac{\partial \bar{v}}{\partial t} + \frac{\partial f}{\partial t}, \tag{3.12}$$

$$\left.\frac{\partial (Gn)}{\partial v}\right|_{\bar{v}} = f\left.\frac{\partial G}{\partial v}\right|_{\bar{v}} + G|_{\bar{v}}\left.\frac{\mathrm{d}f}{\mathrm{d}v}\right|_{\bar{v}} = f\left.\frac{\partial G}{\partial v}\right|_{\bar{v}} + G|_{\bar{v}}\left(\frac{\partial f}{\partial \tau}\underbrace{\left.\frac{\partial \bar{\tau}}{\partial v}\right|_{\bar{v}}}_{(3.7)}\right) = f\left.\frac{\partial G}{\partial v}\right|_{\bar{v}} + \frac{G|_{\bar{v}}}{w}\frac{\partial f}{\partial \tau}. \tag{3.13}$$

Reinserting these transformed differentials into the original PBE (3.1) and simplifying yields the most basic form of the transformed PBE

$$\frac{\partial f}{\partial t} + \underbrace{\frac{\partial f}{\partial \tau}\frac{1}{w}\left(G|_{\bar{v}} - \frac{\partial \bar{v}}{\partial t}\right)}_{(a)} - f\underbrace{\left.\frac{\partial G}{\partial v}\right|_{\bar{v}}}_{(b)} = \dot{s}(\bar{v}, \mathbf{y}, f). \tag{3.14}$$

For any further analysis, it is helpful to get rid of the derivative of the growth rate with respect to the old coordinate. Therefore a general function is evaluated as a derivative with respect to τ, with the help of the product rule and the chain rule this is

$$\frac{\partial}{\partial \tau}\left\{f\frac{1}{w}\left(G|_{\bar{\tau}} - \frac{\partial \bar{v}}{\partial t}\right)\right\} = \underbrace{\frac{\partial f}{\partial \tau}\frac{1}{w}\left(G|_{\bar{\tau}} - \frac{\partial \bar{v}}{\partial t}\right)}_{(a)} + f\underbrace{\frac{\partial (1/w)}{\partial \tau}}_{(3.9)}\left(G|_{\tau} - \frac{\partial \bar{v}}{\partial t}\right) + f\frac{1}{w}\left(\underbrace{\frac{\mathrm{d}G|_{\bar{v}}}{\mathrm{d}\tau}}_{(3.10)} - \frac{\partial^2 \bar{v}}{\partial t \partial \tau}\right). \tag{3.15}$$

Rearranging for (a) and reinserting into (3.14) will yield the final transformed population balance equation

$$\frac{\partial f}{\partial t} + \frac{\mathrm{d}}{\mathrm{d}\tau}\left\{\frac{f}{w}\left(G|_{\bar{v}} - \frac{\partial \bar{v}}{\partial t}\right)\right\} = -\frac{f}{w}\left\{\frac{1}{w}\frac{\partial^2 \bar{v}}{\partial \tau^2}\left(G|_{\bar{v}} - \frac{\partial \bar{v}}{\partial t}\right) + \frac{\partial^2 \bar{v}}{\partial \tau \partial t}\right\} + \dot{s}(\bar{v}, \mathbf{y}, f). \tag{3.16}$$

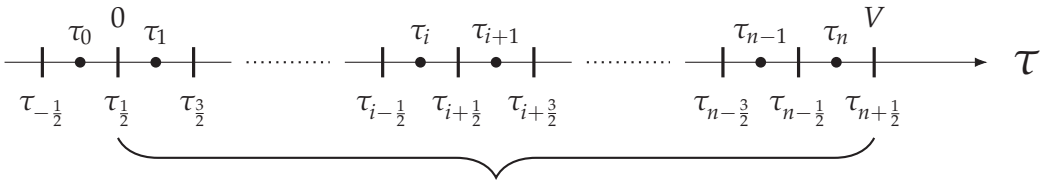
In this equation (b) vanishes with the help of (3.10). This equation contains higher order derivatives on the right-hand side. We therefore assume that the prescribed coordinate transformation varies only linearly in τ and t hence its derivatives on the right hand side will vanish leaving

$$\frac{\partial f(\tau, t)}{\partial t} + \frac{\mathrm{d}}{\mathrm{d}\tau}\left\{\frac{f(\tau, t)}{w(\tau, t)}\left(G(\bar{v}(\tau, t), \mathbf{y}) - \frac{\partial \bar{v}(\tau, t)}{\partial t}\right)\right\} = \dot{s}(\bar{v}(\tau, t), \mathbf{y}, f). \tag{3.17}$$

In this equation, the growth rate is modified by the node speed. That is the cost for using a transformed space. However, when solving for this solution it is unnecessary to map any number density value f to its real counterpart n. The coordinate transformation takes care of any grid adjustments.

3.2.2 Semidiscrete form in transformed space

The transformed population balance equation describes the evolution of the number density f over time t and transformed internal coordinate τ given the coordinate transformation $\bar{v}(\tau, t)$. As mentioned earlier, the coordinate transformation will be based on the current solution. Continuously



$$[0, V] = \Omega = \bigcup_{i=1}^{n} \Omega_i \quad \text{with} \quad \Omega_i = [\tau_{i-\frac{1}{2}}, \tau_{i+\frac{1}{2}}]$$

Figure 3.2: Discretization of the domain $\Omega_v = [0, V]$ into n sub-domains with $n+1$ grid points. For numerical convenience a ghost cell τ_0 is added left to the lower boundary.

speaking, the transformed equation and the equation governing the prescription of the transformation are creating a combined system. Every time the solution changes the coordinate transformation changes too. In a numerical manner, this means that the two discrete equations have to be solved in conjunction. At first, we have to solve the transformed equation. Therefore a finite volume scheme will be applied to the virtual transformed τ-space.

The first step is to discretize the computational domain Ω_τ into cells of finite volume. Since we are in the transformed space a uniform grid is used. $\tau_{i+\frac{1}{2}}$ will address a cell node, i.e., the right cell face. τ_i will be a representative for the i-th cell and also the location of the cell center.

$$\tau_{i+\frac{1}{2}} = i \cdot \Delta\tau, \quad \Delta\tau = \frac{V}{n}, \quad i = 0, 1, \ldots, n \tag{3.18}$$

There are $n+1$ grid points $\tau_{i+\frac{1}{2}}$ and $n+1$ cells τ_i due to the usage of a 'ghost cell' at the left boundary, i.e., the domain Ω is divided into n subdomains Ω_i. Fig. 3.2 schematically depicts this.

The solution of the transformed number density $f(\tau, t)$ is averaged over these cells. This will also be applied to the initial condition.

$$F_i(t) = \frac{1}{\Delta\tau} \int_{\tau_{i-\frac{1}{2}}}^{\tau_{i+\frac{1}{2}}} f(\tau, t)\, \mathrm{d}\tau \qquad i = 1, 2, ..., n \tag{3.19}$$

If (3.17) is now integrated over one uniform cell $[\tau_{i-\frac{1}{2}}, \tau_{i+\frac{1}{2}}]$ and normalized by the cell width $\Delta\tau$ the semi-discrete formulation is found to

$$\frac{\mathrm{d}F_i}{\mathrm{d}t} + \frac{1}{\Delta v_i} \left[f \left(G(\bar{v}, \mathbf{y}) - \frac{\partial \bar{v}}{\partial t} \right) \right]_{\tau_{i-\frac{1}{2}}}^{\tau_{i+\frac{1}{2}}} = \frac{1}{\Delta\tau} \int_{\tau_{i-\frac{1}{2}}}^{\tau_{i+\frac{1}{2}}} \dot{s}(\bar{v}, \mathbf{y}, f)\, \mathrm{d}\tau. \tag{3.20}$$

In the approximation of the advective term the normalization by $\Delta\tau$ changes to the cell width in physical space Δv_i. Since the coordinate transformation is piecewise linear the Jacobian becomes exact by a finite difference expression $w = \Delta v_i / \Delta\tau$. This can be factored out, changing the denominator. The expression inside the square brackets is the flux function, responsible for the conservative change of the cell averaged value

$$\mathcal{F}(\tau, t) = f(\tau, t) \underbrace{\left(G(\bar{v}(\tau, t), \mathbf{y}) - \frac{\partial \bar{v}(\tau)}{\partial t} \right)}_{\tilde{G}(v(\tilde{\tau}, t), \mathbf{y})}. \tag{3.21}$$

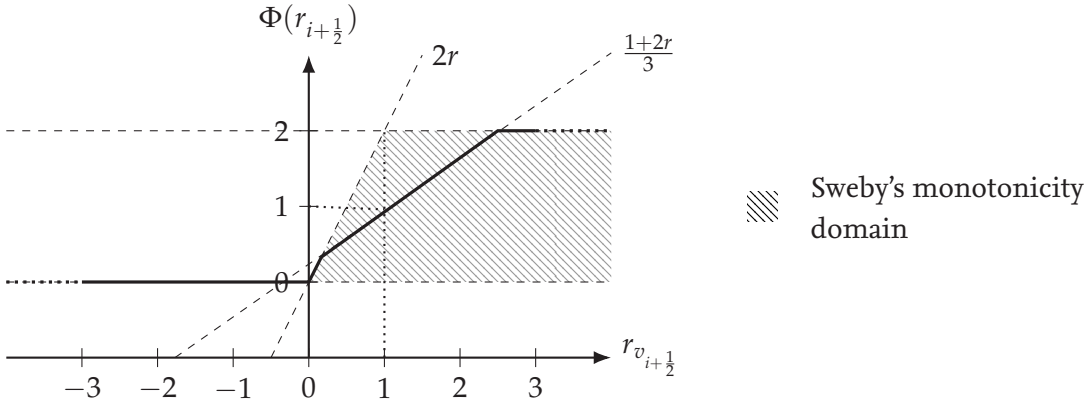


Figure 3.3: The function that is created by the limiter $\Phi(r)$ always lies in or at boundary of Sweby's monotonicity domain (Sweby 1984). Negative values of the solution gradient r result in zero. The light dotted line displays the case $r = 1$.

We can now identify the modified growth rate $\tilde{G}(\bar{\tau}, \mathbf{y})$, as the flux shall also be dependent on the movement speed of the nodes due to the invoking of an adaptive grid. By the idea of fluxes representing the values of the cells being transported in the wind direction, the node movement incorporates an additional wind speed. When a grid node is moving in the positive direction, it creates additional wind from right. This can be seen by the minus sign in front of the term.

For a numerical evaluation of this flux at the cell faces $\tau_{i+\frac{1}{2}}$ a high-resolution scheme can be used, for instance, one that was suggested by Koren (1993) and introduced in the context of the PBE by Qamar et al. (2006). This one is second order accurate and total variation diminishing (TVD). It differentiates between wind from the left ($\tilde{G}|_{v_{i+\frac{1}{2}}} \geq 0$) and from the right ($\tilde{G}|_{v_{i+\frac{1}{2}}} < 0$).

$$\mathcal{F}(\tau_{i+\frac{1}{2}}) := \mathcal{F}_{i+\frac{1}{2}} = \tilde{G}|_{v_{i+\frac{1}{2}}} \begin{cases} F_i + \frac{1}{2}\Phi(r^{left}_{i+\frac{1}{2}})\,(F_i - F_{i-1}) & \tilde{G}|_{v_{i+\frac{1}{2}}} \geq 0 \\ F_i + \frac{1}{2}\Phi(r^{right}_{i+\frac{1}{2}})\,(F_i - F_{i+1}) & \tilde{G}|_{v_{i+\frac{1}{2}}} < 0 \end{cases} \tag{3.22}$$

Here, $\Phi(r)$ is a flux limiter to oppress the creation of wiggles. It is called on the ratio of consecutive solution gradients that also differs between wind from the left and wind from the right.

$$r_{i+\frac{1}{2}} := \begin{cases} r^{left}_{i+\frac{1}{2}} = \frac{F_{i+1} - F_i + \epsilon}{F_i - F_{i-1} + \epsilon} & \tilde{G}|_{v_{i+\frac{1}{2}}} \geq 0 \\ r^{right}_{i+\frac{1}{2}} = \frac{F_i - F_{i+1} + \epsilon}{F_{i+1} - F_{i+2} + \epsilon} & \tilde{G}|_{v_{i+\frac{1}{2}}} < 0 \end{cases} \tag{3.23}$$

A small value ϵ is chosen to avoid a division by zero in uniform flow regions. The flux limiter draws a line that lies inside or at the boundary of an area of monotonicity (Sweby 1984) which is visualized in Fig. 3.3.

$$\Phi(r) = max\left(0, min\left(2r, min\left(\frac{1+2r}{3}, 2\right)\right)\right) \tag{3.24}$$

Together with an approximation for the source integral of (3.20) we the right-hand side $\dot{\mathcal{H}}_i$ becomes

$$\dot{\mathcal{H}}_i = -\frac{1}{\Delta v_i}\left[\mathcal{F}_{i+\frac{1}{2}} - \mathcal{F}_{i-\frac{1}{2}}\right] + \frac{1}{\Delta\tau}\int_{\tau_{i-\frac{1}{2}}}^{\tau_{i+\frac{1}{2}}} \dot{s}(\bar{v}, \mathbf{y}, f)\,\mathrm{d}\tau. \tag{3.25}$$

The minus in front of the fluxes arises because they have been moved to the right-hand side of the equation. In total we now have a system of ordinary differential equations (ODE) for the cell averaged values.

$$\frac{\mathrm{d}F_i}{\mathrm{d}t} = \dot{\mathcal{H}}(\bar{v}(\tau,t), \mathbf{y}, f(\tau,t)) \qquad i = 1,2,\ldots,n \tag{3.26}$$

This can be fed into an ODE solver with adaptive time stepping, such as dopri5 or, in case it gets stiff, radau5. The solver itself is called on the time horizon Δt_k that lies between the update of the transformation. The solver will adjust its step sizes $\Delta t_{solver} \leq \Delta t_k$.

3.2.3 Calculating and updating the coordinate transformation

Once the time horizon $t_k + \Delta t_k$ is reached, the adaptive grid gets updated. Therefore, the current solution is analyzed for steep gradients, near discontinuities, and other features. That is the task of a so-called monitor function. After evaluation, the grid points will be relocated according to an equidistribution principle alongside the monitor value. Additional steps have to be taken to assure that the grid is not overly stretched. Finally, the coordinate transformation is updated in a way that it is marching in time, i.e., the time-dependent transformation $\bar{v}(\tau,t)$ over the next time horizon $t \in [t_{k+1}, t_{k+2}]$ will be calculated based on the old transformation at the endpoint of the last time step $\bar{v}(\tau, t_{k+1})$ and the update $\bar{v}_\infty(\tau)$.

Equidistribution alongside a monitor value The inverse of the Jacobian of the coordinate transformation (3.3) measures the distortion appearing when mapping from the real particle property space to the transformed one. Since the grid in τ-space is always uniform, a high distortion results in densely located nodes in v-space at this given point. By introducing a functional on the number density $m(n)(v)$ that has high value in feature-rich regions this quantity should be set proportional to the inverse Jacobian. We call this functional a monitor.

$$\frac{\partial \bar{\tau}(v,t)}{\partial v} = \frac{1}{w(\bar{\tau}(v,t),t)} = c \cdot m(n)(v) \tag{3.27}$$

In this equation $c > 0$ represents the normalization constant or simply the factor of proportionality. We can fix the time $t = t_{k+1}$ and hence omit any further time dependency in this subsection. Solving this ordinary differential equation for $\bar{\tau}(v,t)$ by taking into account the boundary conditions defined by the constraints in (3.4) and (3.5) yields the equidistribution principle

$$\bar{\tau}_0(v) = \frac{V}{\int_0^V m(n)(v')\,\mathrm{d}v'} \int_0^v m(n)(v')\,\mathrm{d}v'. \tag{3.28}$$

The coefficient in front of the integral defines the normalization constant, assuring that the newly placed nodes will not leave the domain. The equidistribution distorts the virtual τ-domain. Originally, one wants to renew the grid points in the real v-domain. Therefore, at first, the nodes are mapped back by the old transformation and the grid in τ-space is reset to uniform.

$$\bar{v}_0(\tau) := \bar{v}(\tau_0) \qquad \text{and} \qquad \bar{\tau}(v) := \tau_{i+\frac{1}{2}} \quad i = 0,1,2,\ldots,n \tag{3.29}$$

Based on this, the Jacobian of the new transformation can be calculated.

Additional Adjustments to the node density Solely relying on the equidistribution of the nodes alongside the monitor value might bring up two risks:

1. Nodes will only lie in feature rich-regions. Consequentially, the grid may become sluggish and the node movement might not be capable of capturing sudden changes. A smooth grid is preferable. Due to Dorfi and Drury (1987) a common measure is to restrict the local boundedness
$$\frac{1}{\alpha} \leq \frac{v_{i+\frac{1}{2}} - v_{i-\frac{1}{2}}}{v_{i-\frac{1}{2}} - v_{i-\frac{3}{2}}} \leq \alpha \qquad i = 2, 3, \ldots, n, \tag{3.30}$$
where $\alpha > 0$ defines the maximum admissible grid stretching in v-space.

2. A change of the solution in a different region than where features lie, e.g. caused by nucleation, might be under-resolved. To address this, a minimum node density has to be prescribed on v-space that the coordinate transformation has to respect.
$$\rho(v) \geq \rho_{min}(v) \qquad \forall v \in [0, V] \tag{3.31}$$

Since it is unlikely that the found coordinate transformation $\bar{v}_0(\tau)$ is able to fulfill both conditions, it has to be further adjusted to a transformation that we call $\bar{v}_\infty(\tau)$. To do this, Sewerin and Rigopoulos (2017) proposed an Initial Value Problem in terms of the inverse Jacobian that implements this adjustment with the help of the iterator variable $s \in \{1, 2, \ldots, maxStep\}$

$$\begin{cases} \dfrac{\mathrm{d}\bar{\tau}_{s+\frac{1}{2}}(v)}{\mathrm{d}v} := \dfrac{\mathrm{d}\bar{\tau}_s(v)}{\mathrm{d}v} + \max\left(\dfrac{\max(\rho_{min}(v), P(\rho_)(v_j))}{\rho_\tau} - \dfrac{\mathrm{d}\bar{\tau}_s(v)}{\mathrm{d}v},\ 0\right) \\ \dfrac{\mathrm{d}\bar{\tau}_{s+1}(v)}{\mathrm{d}v} = \dfrac{\mathrm{d}\bar{\tau}_{s+\frac{1}{2}}(v)}{\mathrm{d}v}\left(\dfrac{1}{V}\displaystyle\int_0^V \dfrac{\mathrm{d}\bar{\tau}_{s+\frac{1}{2}}(v')}{\mathrm{d}v'}\,\mathrm{d}v'\right)^{-1} \end{cases} . \tag{3.32}$$

The first equation can be understood as removing points from high node density regions and reinserting them in locations that need more points either to respect the grid stretching α or the minimum node density ρ_{min}. In the second equation the grid is renormalized. It is repeatedly applied until the change of the coordinate transformation in one step is under a given tolerance
$$\frac{\mathrm{d}\bar{\tau}_{s+1}(v)}{\mathrm{d}v} - \frac{\mathrm{d}\bar{\tau}_s(v)}{\mathrm{d}v} < tolerance. \tag{3.33}$$

In (3.32) $P(\rho)(v)$ denotes a padding that is applied to the current node density $\rho(v)$. The padding can be seen as a functional used in smoothing sharp node density peaks. It is calculated with the help of three linked equations

$$\begin{cases} P(\rho)(v) := \max\limits_{v' \in [0,V]} \left(\frac{\rho_0(v')}{1+\lambda|v-v'|\rho_0(v')}\right) \\ \lambda = \frac{ln\,\alpha}{d} \\ d = \frac{1}{n}\int_0^V P(\rho)(v)\,\mathrm{d}v \end{cases} . \tag{3.34}$$

The maximum admissible grid stretching α is obeyed within the calculations of the constant λ. With the help of the initial guess $d \approx \Delta\tau$, this system is solved iteratively until the padding of the node density $P(\rho)(v)$ does not change anymore.

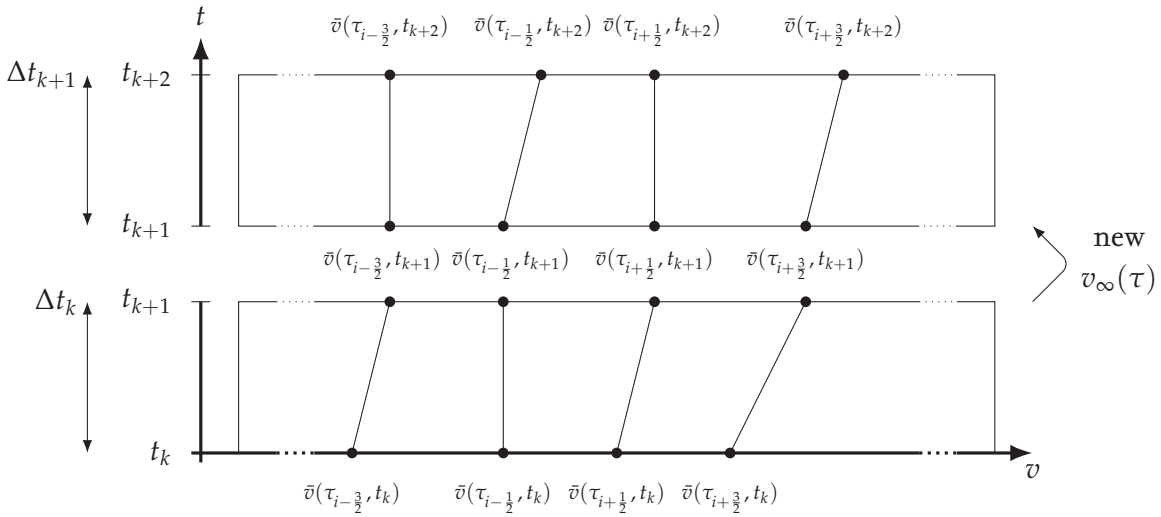


Figure 3.4: The time dependent coordinate transformation $\bar{v}(\tau, t)$ spans a $v-t$-mesh. At a given point in time t_k the previous transformation is known $\bar{v}(\tau, t_k)$. With the help of the newly calculated update $\bar{v}_\infty(\tau)$, we can extrapolate the transformation of the next time step t_{k+1}. The one in-between can be linearly interpolated.

Every iteration of the two-step scheme (3.32) takes a full calculation of (3.34) to obtain the padding $P(\rho)(v)$. In the transformed τ-space the nodes are uniformly distributed, resulting in a constant node density $\rho_\tau = n/V$. The current node density in the real particle property space that is needed can be computed by the help of the distortion of the current inverse Jacobian

$$\rho_0(v) = \frac{\mathrm{d}\bar{\tau}_0(v)}{\mathrm{d}v}\rho_\tau. \tag{3.35}$$

When the two-step scheme (3.32) is converged, the adjusted Jacobian then yields the wanted coordinate transformation $\bar{v}_\infty(\tau)$.

Coordinate Transformation for the next time step Given the coordinate transformation at discrete points in time $\bar{v}(\tau, t_k)$, we can estimate the new transformation at the next point in time by a smooth transition between the old one (that was just an estimate made from the one at t_{k-1}) and the current update $\bar{v}_\infty(\tau)$. By the use of exponentials, the next transformation is

$$\bar{v}(\tau, t_{k+1}) = \left(1 - e^{-\frac{\Delta t_k}{\kappa}}\right)\bar{v}_\infty(\tau) + e^{-\frac{\Delta t_k}{\kappa}}\bar{v}(\tau, t_k) \tag{3.36}$$

The exponential prefactors implement a first-order lag element with a constant of $\kappa > 0$. By this scheme, the transformation marches in time meaning that the transformation at every next step also depends on all old transformations. Now, for every discrete point in the time horizon there is a new transformation. However, to solve the actual population balance equation in transformed space information on the grid node locations is needed more frequently, i.e., at all the adapted time steps the ODE solver chooses. We therefore use a linear interpolation between the current transformation $\bar{v}(\tau, t_k)$ and the guess at the next time horizon $\bar{v}(\tau, t_{k+1})$ to

$$\bar{v}(\tau, t) = \bar{v}(\tau, t_k) + \left(\bar{v}(\tau, t_{k+1}) - \bar{v}(\tau, t_k)\right)\frac{t - t_k}{\Delta t_k} \qquad \text{for } t \in [t_k, t_{k+1}]. \tag{3.37}$$

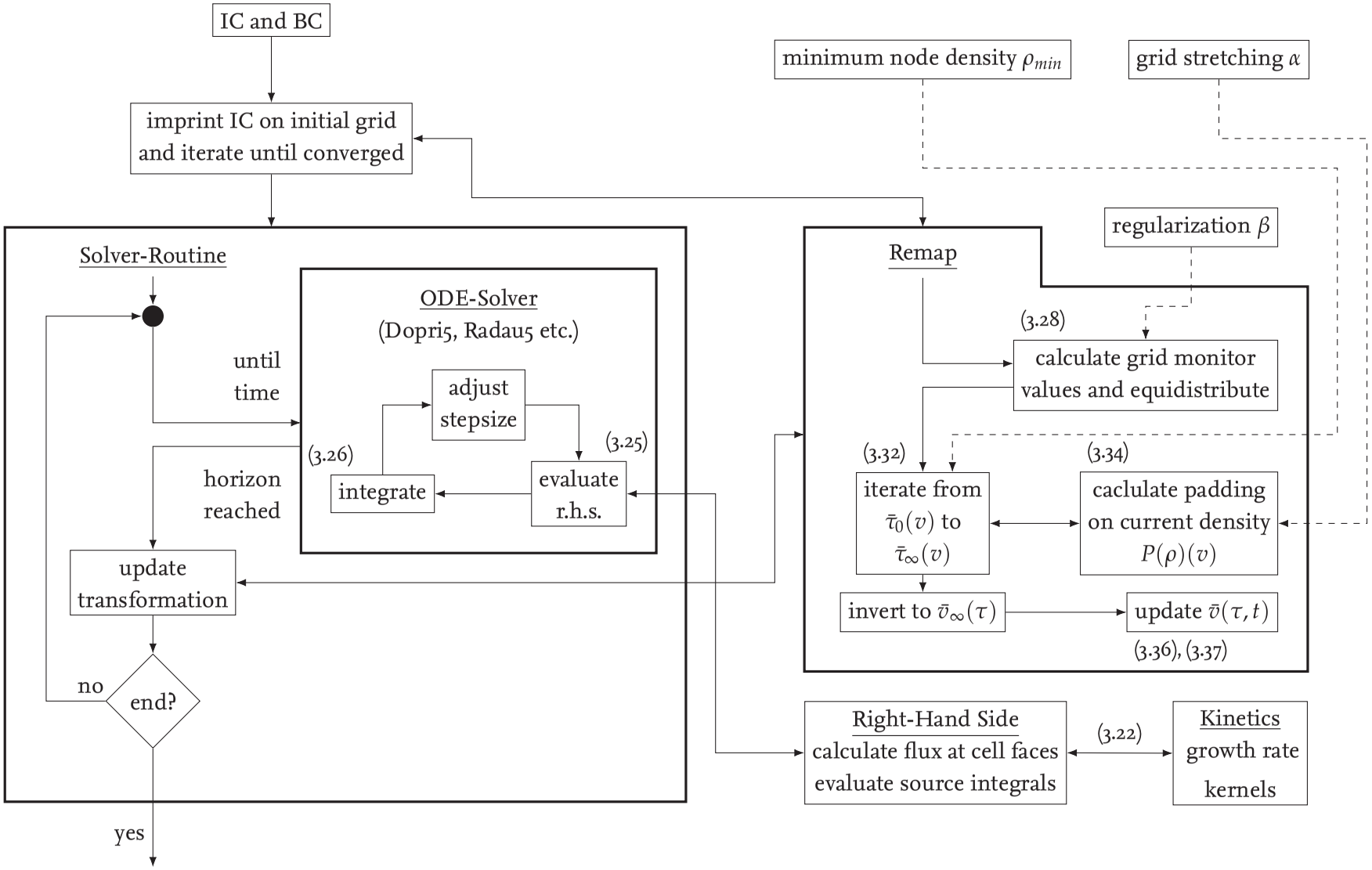


Figure 3.5: Flow scheme of the algorithm used in this work. The numbers refer to the corresponding equations.

For the first point in time, no previous transformation is available meaning it will be set to the 'update' of the first time step.

$$\bar{v}(\tau, t_0 = 0) = \bar{v}_\infty(\tau) \tag{3.38}$$

Summary of the steps For future reference, we call the combination of all the procedures introduced in this subsection the remap subroutine. It is called multiple times in advance of a simulation to find a suitable initial grid by constantly reinterpolating the initial condition onto the grid. As an aid to the reader, a flow scheme incorporating the entire simulation process is given in Fig. 3.5.

3.3 Monitor Functions used in the literature

Monitor functions deliver the information necessary to analyze the current solution, i.e., they identify regions that require higher node density. As to the equidistribution principle (3.28) the monitor values are directly linked to the coordinate transformation.

In a more general framework Dorfi and Drury (1987) adapted the most frequently used monitor of *Arclength*. It is based on the idea that in regions of near discontinuity the solution spans over high values of n in small areas of v hence a long arclength can be seen. In the case of an one-dimensional PBE, this becomes the contribution of an infinitesimal small length element of the two-dimensional curve $\mathbf{x} = (v, n(v))^T$

$$ds = \sqrt{(\mathrm{d}x_1)^2 + (\mathrm{d}x_2)^2}. \tag{3.39}$$

When transforming the differential from $\mathrm{d}s$ to $\mathrm{d}v$, the *Arclength* functional $m(n)(v)$ can be identified

$$\mathrm{d}s = \sqrt{\left(\frac{\mathrm{d}x_1}{\mathrm{d}v}\right)^2 + \left(\frac{\mathrm{d}x_2}{\mathrm{d}v}\right)^2}\,\mathrm{d}v = \underbrace{\sqrt{1 + \left(\frac{\mathrm{d}n}{\mathrm{d}v}\right)^2}}_{m^{arclength}(n)(v)}\,\mathrm{d}v. \tag{3.40}$$

Many adaptive grid schemes use this monitor. Ceniceros and Hou (2001) and Tang and Tang (2003) even adapted it for multidimensional cases where it represents the surface the function spans over an inifinitesimal small rectangle $\mathrm{d}v_1 \times \mathrm{d}v_2$.

Based on the idea of *Arclength*, a similar monitor can be derived that equidistributes the grid nodes alongside the *Curvature* $\mathrm{d}c$ of a function (Blom and Verwer 1989).

$$\mathrm{d}c = \underbrace{\sqrt[4]{1 + \left(\frac{\mathrm{d}^2 n}{\mathrm{d}v^2}\right)^2}}_{m^{curvature}(n)(v)}\,\mathrm{d}v \tag{3.41}$$

Both *Arclength* and *Curvature* directly depend on the shape of the solution and represent the most basic ideas to measure this function. Going further, one can find descendants of these monitors. Sewerin and Rigopoulos (2017) used a simpler version of the family of *Arclength* monitors by omitting the square of the first derivative. We will refer to this as *Arc-Simple*. Since a near discontinuity consists of a steep gradient resulting in a high arclength and sharp corners resulting in high curvature, a logical consequence would be to combine both monitors. This was first introduced by Dorfi and Drury (1987) and we will refer to this as *Arc-Curv*.

Table 3.1: Overview of different monitor functions appearing in the literature. Before applying, the number density function is normalized $n(v)/\Psi_0$ by the zeroth moment, i.e, by the total number of particles.

class	name	continuous form $m(n)(v) =$	discrete implementation $M_{i+1/2} =$	main focus
shape dependent	(a) **Arclength**	$\sqrt{1+\beta\left(\frac{\mathrm{d}n}{\mathrm{d}v}\right)^2}$	$\sqrt{1+\beta\left(\frac{N_{i+1}-N_i}{v_{i+1}-v_i}\right)^2}$	steep gradients
	(b) **Curvature**	$\sqrt[4]{1+\beta\left(\frac{\mathrm{d}^2n}{\mathrm{d}v^2}\right)^2}$	$\mathcal{A}\left(\sqrt[4]{1+\beta N_{vv_i}^2}\,,\ \sqrt[4]{1+\beta N_{vv_{i+1}}^2}\right)$	sharp corners
	(c) **Arc-Simple**	$\sqrt{1+\beta\left\lvert\frac{\mathrm{d}n}{\mathrm{d}v}\right\rvert}$	$\sqrt{1+\beta\left\lvert\frac{N_{i+1}-N_i}{v_{i+1}-v_i}\right\rvert}$	faster (a)
	(d) **Arc-Curv**	$\sqrt{\beta_1\left(\frac{\mathrm{d}n}{\mathrm{d}v}\right)^2+\sqrt{1+\beta_2\left(\frac{\mathrm{d}^2n}{\mathrm{d}v^2}\right)^2}}$	$\sqrt{\beta\left(\frac{N_{i+1}-N_i}{v_{i+1}-v_i}\right)^2+80\mathcal{A}\left(\sqrt{1+\beta N_{vv_i}^2}\,,\ \sqrt{1+\beta N_{vv_{i+1}}^2}\right)}$	combine (a) & (b)
(e) **Artificial Diffusivity**		$\left\lvert\Gamma^{art}(n)(v)\right\rvert^\eta$	$\epsilon+\lvert\Gamma_i\rvert^\eta$	oscillations

(a) (Blom and Verwer 1989; Dorfi and Drury 1987; Stockie et al. 2001)

(b) (Blom and Verwer 1989)

(c) 'A2' in (Sewerin and Rigopoulos 2017)

(d) (Dorfi and Drury 1987)

$$\text{Second Derivative inside cell } N_{vv_i} = 2\cdot\frac{\frac{N_{i+1}-N_i}{v_{i+1}-v_i}-\frac{N_i-N_{i-1}}{v_i-v_{i-1}}}{(v_{i+1}-v_{i-1})}.$$

$$\text{Cell face value on nonuniform grids } \mathcal{A}_{i+\frac{1}{2}}(a_i,a_{i+1}) = \frac{v_{i+1}-v_{i+\frac{1}{2}}}{v_{i+1}-v_i}a_i+\frac{v_{i+\frac{1}{2}}-v_i}{v_{i+1}-v_i}a_{i+1}.$$

Other combinations such as the one of Qamar et al. (2007) that adds the absolute value of the function under the square root of an arclength monitor

$$m(n)(v) = \sqrt{1 + |n| + \left(\frac{\mathrm{d}n}{\mathrm{d}v}\right)^2}, \tag{3.42}$$

are not suitable for our cases. Here, higher values of the function will be additionally scored which is not the intent of a monitor function.

At this point, we have defined the continuous counterparts of four monitors that are mainly based on the shape of the current solution. To oppress any unwanted influences to the monitor, the current solution is normalized by the total number of particles, i.e., the zeroth moment

$$n(v) \leftarrow \frac{n(v)}{\Psi_0} \quad \text{with} \quad \Psi_0 = \int_0^V n(v)\,\mathrm{d}v. \tag{3.43}$$

Now the monitor will only analyze the shape. Additionally, we introduce a regularization constant $\beta > 0$ that magnifies the value of the derivatives against the additive one. The value has an influence on how well the monitor function can analyze the solution. According to Sewerin and Rigopoulos (2017) there is a threshold of $\beta = 10^4$ for the case of step advection above which the accuracy is the best. Therefore, we adapt this value.

If the derivatives are now approximated by finite difference, we obtain the discrete forms as in Tab. 3.1. Here v_i denotes the position of the cell center that is midway between the cell faces $v_i = (v_{i+\frac{1}{2}} - v_{i-\frac{1}{2}})/2$. For the approximation of the second derivative, we use the formula of Blom and Verwer (1989). Since this value of the second derivative is associated with the cell center v_i, a cell face averaging operator for non-uniform grids $\mathcal{A}$ is introduced.

The combined *Arc-Curv* monitor originally consists of two regularization constants since the magnitude of curvature is different than *Arclength*. We instead used one value β and multiplied by a constant. For the case of step advection, we found 80 to be a good value to deliver a smooth blend of both contributions.

3.4 A Monitor Function based on Artificial Diffusivity

> 'Monotone behavior of a numerical solution cannot be assured for linear finite-difference methods with more than first-order accuracy.' (Godunov 1959)

Godunov's theorem states that there are no finite volume flux approximations that are free of introducing wiggles with higher accuracy than first order. Wiggles occur when the discretization scheme has to approximate steep gradients. It will therefore intuitively create oscillations around it to achieve a better approximation of the steep slope of the cell averaged solution (Miller and Miller 1981). These additional oscillations, however, disrupt the monotonicity of the solution, hence Godunov's theorem. This is visualized schematically in Fig. 3.6. Only low accuracy schemes, i.e., ones that are first-order accurate will not be affected since their underlying stencil is too small to capture those gradients.

Additionally, some finite volume schemes will experience numerical diffusivity that smears out its sharp corners. In general, this behavior is unwanted since it reduces the information of these features. So-called total variation diminishing (TVD) schemes or high-resolution schemes now

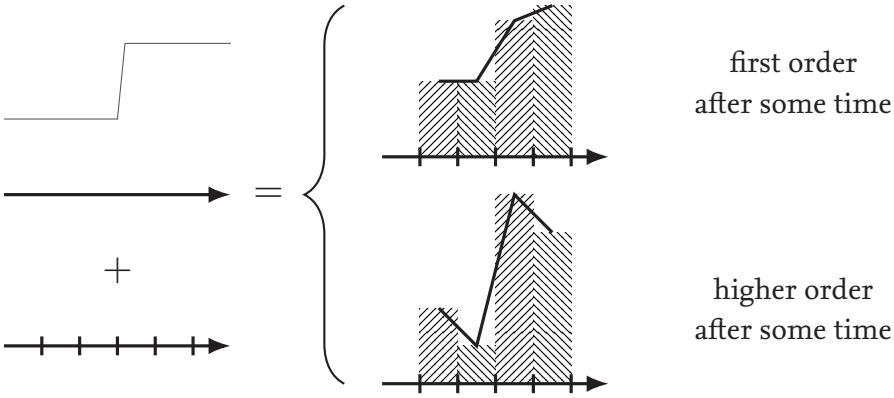


Figure 3.6: Higher order schemes naturally introduce oscillations into the solution to better approximate a steep gradient (Miller and Miller 1981). TVD-schemes (not to be seen here) try to oppress the creation of such wiggles.

attempt to incorporate artificial diffusivity to oppress the creation of wiggles. Therefore, they try to limit the flux where necessary to obtain a monotone solution, i.e., the flux is smeared a bit. In the previous section, we introduced the scheme of Koren (1993) for approximating the cell face flux. This scheme incorporated a flux limiter $\Phi(r)$ that was defined based on the help of a monotonicity domain.

Given the background that the scheme inevitably introduces oscillations that it tries to oppress at the same time, it naturally includes a measure of discontinuity. In other words, the way the flux is limited tells about the features of the solution. That is what one could exploit and use in the sense of a monitor function. Therefore, we compare the fluxes of the scheme of Koren (1993) with the fluxes a second-order central difference scheme yields and with the fluxes an approximation of a general diffusion equation uses. In the end, a different measure of the current solution is found compared to the monitors of Tab. 3.1 since they all directly rely on the shape of the solution.

3.4.1 Numerical Diffusivity as a measure of discontinuity

In this subsection, an expression for the artificial diffusivity is derived based on a comparison of flux approximations. Therefore, we consider a general diffusion equation on the uniform transformed space with diffusivity $\Gamma(v)$

$$\frac{\partial f(\tau,t)}{\partial t} = \frac{\partial}{\partial \tau}\left(\Gamma(\tau)\frac{\partial f(\tau,t)}{\partial \tau}\right). \tag{3.44}$$

We apply the standard finite volume discretization of Fig. 3.2 by integrating over the cell $\Omega_i = [\tau_{i-\frac{1}{2}}, \tau_{i+\frac{1}{2}}]$ and normalizing by the cell width $\Delta\tau$. Doing so and incorporating a finite difference approximation of the inner derivative we get

$$\frac{\mathrm{d}F_i}{\mathrm{d}t} = \frac{1}{\Delta\tau}\Bigg[\underbrace{\Gamma|_{\tau i+\frac{1}{2}}\frac{F_{i+1}-F_i}{\tau_{i+1}-\tau_i}}_{\mathcal{F}^{diff}_{i+\frac{1}{2}}} - \underbrace{\Gamma|_{\tau_{i-\frac{1}{2}}}\frac{F_i-F_{i-1}}{\tau_i-\tau_{i-1}}}_{\mathcal{F}^{diff}_{i-\frac{1}{2}}}\Bigg]. \tag{3.45}$$

Here $\mathcal{F}^{diff}_{i\pm\frac{1}{2}}$ denotes the diffusive flux at the left cell face and the right cell face, respectively.

Now, we recall the advective flux approximation on the right cell face in the case of wind from the left of (3.22). Instead of the modified growth rate $\tilde{G} = (G - \partial v/\partial t)/w$, we neglect the node speed

$\partial v/\partial t$ since we are looking for a transformation that is fixed in time, hence the advective coefficient becomes the original growth rate that is corrected by the Jacobian

$$\mathcal{F}^{adv}_{i+\frac{1}{2}} = -\frac{G_{i+\frac{1}{2}}}{w_{i+\frac{1}{2}}}\left(F_i + \frac{1}{2}\Phi(r^{up}_{i+\frac{1}{2}})(F_i - F_{i-1})\right). \tag{3.46}$$

The minus sign arises as we assume the advective term would have been moved to the right side of the equation as similarly shown in (3.25). By adding an artificial zero to the expression in the parenthesis (by adding and subtracting a weighted cell face number density value), one can identify a central difference flux approximation $\mathcal{F}^{adv,central}_{i+\frac{1}{2}}$. Additionally, we can multiply the second term by an artificial one.

$$\begin{aligned}\mathcal{F}^{adv}_{i+\frac{1}{2}} = \underbrace{-\frac{G_{i+\frac{1}{2}}}{w_{i+\frac{1}{2}}}\Bigg(N_i + \overbrace{\frac{\tau_{i+\frac{1}{2}} - \tau_i}{\tau_{i+1} - \tau_i}(F_{i+1} - F_i)}^{\text{added artificial 0}} - \frac{\tau_{i+\frac{1}{2}} - \tau_i}{\tau_{i+1} - \tau_i}(F_{i+1} - F_i)}_{\mathcal{F}^{adv,central}_{i+\frac{1}{2}}} + \\ \frac{1}{2}\Phi(r^{up}_{i+\frac{1}{2}})(F_i - F_{i-1})\underbrace{\frac{\tau_{i+1} - \tau_i}{\tau_{i+1} - \tau_i}\frac{F_{i+1} - F_i}{F_{i+1} - F_i}}_{\text{multiplied by 1}}\Bigg)\end{aligned} \tag{3.47}$$

Up one resorting the order of multiplication in the two lasting terms, we can factor out a finite difference expression of the number density. Additionally, the inverse of the solution gradient can be found as $(F_i - F_{i-1})/(F_{i+1} - F_i) = (r^{left}_{i+\frac{1}{2}})^{-1}$. In a comparison with the diffusive flux of the right cell face $\mathcal{F}^{diff}_{i+\frac{1}{2}}$, obtained in (3.45), we can define the rest of the expression as the artificial diffusivity introduced by the TVD-scheme since a numerical flux by a central difference approximation does not incorporate any additional diffusivity.

$$\mathcal{F}^{adv}_{i+\frac{1}{2}} = \mathcal{F}^{adv,central}_{i+\frac{1}{2}} + \frac{F_{i+1} - F_i}{\tau_{i+1} - \tau_i}\cdot\underbrace{\frac{G_{i+\frac{1}{2}}}{w_{i+\frac{1}{2}}}\left((\tau_{i+\frac{1}{2}} - \tau_i) - \frac{1}{2r^{up}_{i+\frac{1}{2}}}\Phi(r^{up}_{i+\frac{1}{2}})(\tau_{i+1} - \tau_i)\right)}_{\text{Artificial diffusivity at } \tau_{i+\frac{1}{2}} := \Gamma^{art}_{i+\frac{1}{2}}} \tag{3.48}$$

Similar to the derivation made here, the expression in the case of wind from the right can be found whereas the solution gradients $r_{i+\frac{1}{2}}$ depending on the wind direction are defined in (3.23). The flux limiter $\Phi(r)$ is given by (3.24).

$$\Gamma^{art}_{i+\frac{1}{2}} = \frac{G_{i+\frac{1}{2}}}{w_{i+\frac{1}{2}}}\begin{cases}(\tau_{i+\frac{1}{2}} - \tau_i) - \frac{1}{2r^{left}_{i+\frac{1}{2}}}\Phi(r^{left}_{i+\frac{1}{2}})(\tau_{i+1} - \tau_i) & G_{i+\frac{1}{2}} \geq 0 \\ \frac{1}{2r^{right}_{i+\frac{1}{2}}}\Phi(r^{right}_{i+\frac{1}{2}})(\tau_{i+1} - \tau_i) - (\tau_{i+1} - \tau_{i+\frac{1}{2}}) & G_{i+\frac{1}{2}} < 0\end{cases} \tag{3.49}$$

Since the grid in τ-space is uniform the node distance will always be $\tau_{i+1} - \tau_i = \Delta\tau = const.$ Consequentially, the distance between the cell midpoint and the cell face is $\tau_{i+\frac{1}{2}} - \tau_i = \Delta\tau/2 = const.$ By factoring out these constant values, we see that they only magnify the value of the artificial

diffusivity at a given grid node but do not deliver any additional information on the shape of the solutions. Hence we can neglect them yielding

$$\Gamma^{art}_{i+\frac{1}{2}} = \frac{G_{i+\frac{1}{2}}}{w_{i+\frac{1}{2}}} \begin{cases} 1 - \frac{\Phi(r^{left}_{i+\frac{1}{2}})}{r^{left}_{i+\frac{1}{2}}} & G_{i+\frac{1}{2}} \geq 0 \\ \frac{\Phi(r^{right}_{i+\frac{1}{2}})}{r^{right}_{i+\frac{1}{2}}} - 1 & G_{i+\frac{1}{2}} < 0 \end{cases}. \tag{3.50}$$

The expression left for the *Artificial Diffusivity* $\Gamma^{art}_{i+\frac{1}{2}}$ analyzes the solution in multiple ways. First, the monitor value will be high for fast-growing particles, hence more nodes will be placed in regions where high flow occurs. Secondly, the expression is inversely proportional to the Jacobian. Since it measures the distortion when mapping between the τ-space and the v-space the monitor value will be high in regions where the nodes are already densely packed meaning it will intensify the grid. Lastly, it also depends on the solution itself. The flux limiter decides whether the flux should rather be approximated by central difference or upwind. It is one in the regions of uniform solution where the ratio of consecutive solution gradients $r_{i+\frac{1}{2}}$ is also one. By subtracting it from one, we find the deviation from uniform flow regions hence the regions of change that require higher node density.

3.4.2 Implementation in a numerical solution scheme

To conclude the definition of (3.50), we have to find an expression for the Jacobian value at the cell face $w_{i+\frac{1}{2}}$. This can be approximately done by the help of the average of the Jacobian on both sides of the cell face

$$w_{i+\frac{1}{2}} = \frac{1}{2}(w_i + w_{i+1}) = \frac{1}{2}\left(\frac{\Delta v_i}{\Delta\tau} + \frac{\Delta v_{i+1}}{\Delta\tau}\right) \sim \Delta v_i + \Delta v_{i+1} = v_{i+\frac{3}{2}} - v_{i-\frac{1}{2}}. \tag{3.51}$$

Now the set of equations to calculate the value of *Artificial Diffusivity* is complete.

This monitor offers a new way to describe the solution that is also based on the way the underlying discretization scheme works, i.e., how the cell face flux is approximated. Therefore, it behaves differently from the monitors presented in Tab. 3.1. According to the equidistribution principle, the grid nodes in τ-space are distorted by the equidistribution alongside the monitor value. However, this process incorporates a division by the monitor values and since *Artificial Diffusivity* will be zero in regions where the solution will not change, one has to add a small value ϵ to avoid a division by zero.

$$M_{i+\frac{1}{2}} = \epsilon + \left|\Gamma^{art}_{i+\frac{1}{2}}\right|^{\eta} \tag{3.52}$$

An issue of this monitor function is that its values will become very high compared to the other monitors. This is mainly caused by the Jacobian in the denominator. In regions where the mesh is already dense, one divides by a small value hence the grid will additionally increase in node density at this point. The presented implementation of padding and minimum node density may be incapable of smoothing the grid. This was observed by some early implementations in which the application of this monitor did not respect the prescribed grid stretching α and the iteration of the two-step scheme (3.32) did not converge.

A solution would be to additionally smooth the monitor values before they start distorting the τ-grid by the equidistribution principle (3.28). As seen above, one could smooth the value by a general

Table 3.2: Values for the root limiter η found to be best suited. They will depend on the chosen kind of floating point number in the code implementation (here for double precision).

Number of Cells	**Grid stretching** α	16	32	64	128	256	512
Limiter η (normal domains)	2.0	4.0	0.95	0.55	0.40		
Limiter η (exponential domains)	1.5	2.0	0.6	0.5	0.3	0.15	0.1

root function with the limiter η. We also tried to use a low-pass filter as seen in Tang and Tang (2003)

$$M_{i+\frac{1}{2}} \leftarrow \frac{1}{4}\left[\frac{M_{i-\frac{1}{2}} \cdot (v_{i+\frac{3}{2}} - v_{i+\frac{1}{2}}) + M_{i+\frac{3}{2}} \cdot (v_{i+\frac{1}{2}} - v_{i-\frac{1}{2}})}{v_{i+\frac{3}{2}} - v_{i-\frac{1}{2}}} + 2 \cdot M_{i+\frac{1}{2}}\right]. \tag{3.53}$$

However, the results were not satisfying. Therefore, we stick to the original approach of (3.52). In general, the value η is best set to be as high as possible for the given number of cells employed. Setting it too low ignores the adaptive grid itself since the monitor value will then become ϵ almost everywhere. Tab. 3.2 summarizes the values we found to be best. It differentiates between domains spanning over one or two orders of magnitude and exponential domains that encompass multiple orders.

3.5 Advection of a unit step profile

In this section, we test the various monitor functions on an example of a homogeneous PBE with a constant growth rate, i.e., a simple advection equation

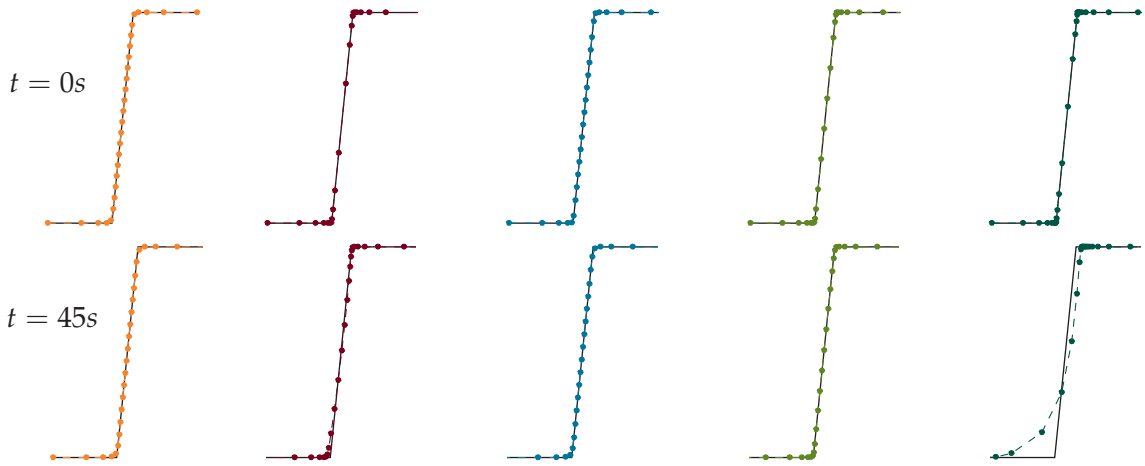
$$\frac{\partial n(v,t)}{\partial t} + g\frac{\partial n(v,t)}{\partial v} = 0 \quad v \in \Omega = [0, V], t > 0. \tag{3.54}$$

Here, we set the right boundary to $V = 4$ and the growth rate/advection speed to $g = 0.05\,\mathrm{s}^{-1}$. According to the methods of characteristics, the solution to this partial differential equation will be the initial condition propagating to the right at a constant velocity $n(v,t) = n_0(v - gt)$. To measure the capability of resolving steep gradients and sharp peaks, we introduce a unit step function with a high slope and two sharp corners.

$$n(v,0) := n_0(v) = \begin{cases} 0 & v < 0.15 \\ \frac{v-0.15}{0.05} & v \in (0.15, 0.2) \\ 1 & v > 0.2 \end{cases} \tag{3.55}$$

The simulation went for 60 s with a major time horizon of $\Delta t_k = 0.01\,\mathrm{s} = const$, i.e., the grid was recalculated every 1/100 s.

We differentiate between the time at the beginning of the simulation and time $t = 45\,\mathrm{s}$. Fig. 3.7 presents the shape of the numerical solution magnified from the entire domain $\Omega_v = [0, V]$ to $v \in [0, 0.35]$ in the top row and $v \in [2.25, 2.65]$ in the bottom row. The height of the solution was captured well by all 5 investigated monitor function. The marks in the given plots represent, on the one hand, the grid node locations and, on the other hand, the cell face values. These have been linearly interpolated by the operator $\mathcal{A}$ introduced at the bottom of Tab. 3.1. To compare the results



(a) Arclength (b) Curvature (c) A2 (d) ArcCurv (e) Diffusion

Figure 3.7: The different shapes present how the various monitors capture the solution at the initial condition and at time $t = 45\,\mathrm{s}$. The plots are magnified from the computational domain and are calculated with 32 finite volume cells at a grid stretching of $\alpha = 2.0$.

quantitatively, we use the L_1-error measure

$$error(t) = \frac{1}{V} \int_0^V \left| n^{analytical}(v,t) - n^{numerical}(v,t) \right| \mathrm{d}v, \tag{3.56}$$

and refer to it as *average deviation from analytical solution* or simply *error*.

If we now compare the different shapes, it is obvious to identify that the monitors *Arc-Simple* as well as *Arc-Curv* are based on the *Arclength* one. The nodes are equidistributed alongside the contour of the solution. Only the slightly modified *Arc-Curv*-monitor shows little superior capturing of the sharp corners. A big disadvantage of all three variants is the high resolution on the long slope that is technically unnecessary. However, the implementations are very robust, showing a consistently good capturing even at later time steps.

The *Curvature*-monitor focuses more on sharp corners. However, the approximation of the second derivative is error-prone when having a piece-wise linear function. When examining the plot at a higher number of cells the nodes are placed in a way to induce additional 'ministeps' over the steep gradient. We guess that a little error on parts of high slope can be additionally magnified since this monitor just analyzes such features. Fig. 3.8 illustrates the performance of the different monitors. Even though the *Curvature*-monitor works better than the reference finite volume implementation on a fixed uniform grid it is still about one order of magnitude behind the best. Additionally, the simulation at cell counts higher than 2048 was not possible since the two-step scheme of (3.32) did not converge. Presumably, the influence of the monitor is too high, i.e., the monitor values themselves span over many orders of magnitude. Then, the additional adjustments are unable to smooth the grid and the simulation stops. We could also observe this behaviour with the *Arc-Curv*-monitor.

As for the novel monitor based on *Artificial Diffusivity* the first results are poor. Even if the monitor captures the initial condition (IC) superior compared to the other tested ones, it falls behind at later points in time. In the case of 32 finite volume cells, as shown in Fig. 3.7, the top right corner of the

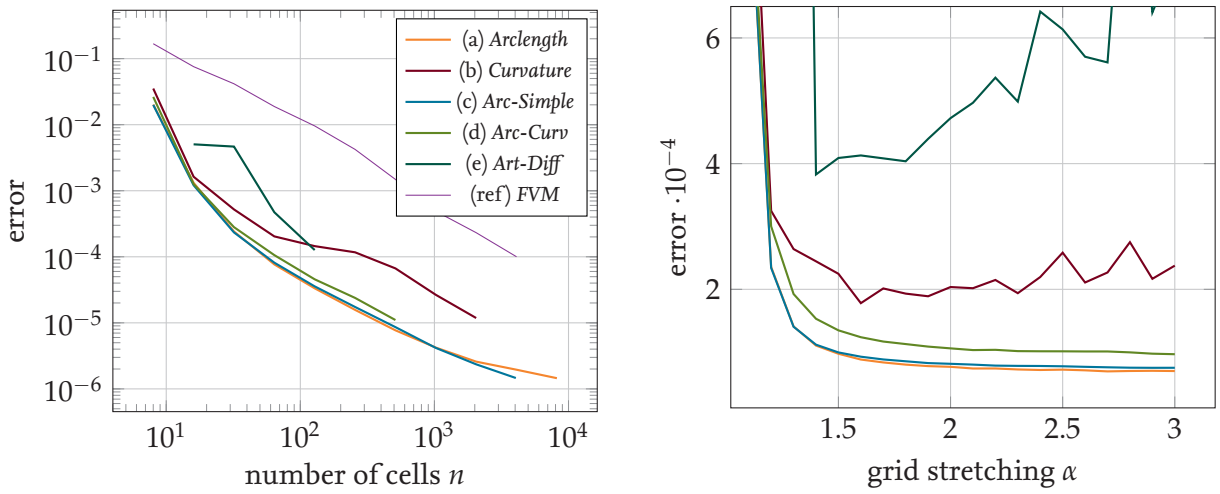


Figure 3.8: Analysis on the error (average deviation from analytical solution) over the number of finite volume cells at fixed grid stretching $\alpha = 2.0$ and over the grid stretching at 64 cells, both at time $t = 45s$.

step profile is captured better than the lower parts. When zooming out of the magnified image, a region of higher node density appears but it is not where it has to be. In reverse, this means that those are the locations the scheme finds the highest artificial diffusivity.

The connection between the way the points of the IC are placed for *Artificial Diffusivity* and *Curvature* is interesting and might show a connection between these two approaches. When it comes to accuracy, the *Artificial Diffusivity* monitor is still better than the reference implementation but it is very inconsistent. For higher cell counts than 128 we did not find any suitable values for the limiter η in (3.52), as a consequence the simulation did not run. Here this monitor shares the same problem as the one of *Curvature* because the two-step scheme does not converge.

In Fig. 3.8 we also plot the average deviation from the analytical solution at 64 finite volume cells with varying grid stretchings α. For the *Arclength* monitor and its descendants, the error soon converges. For the sole *Curvature* there are oscillations but they are mainly bounded. Based on this data, it could be assumed that α can also be set to values higher than 2.0 but we advise against this since this makes the adaptive grid sluggish. When it comes to the monitor on *Artificial Diffusivity*, 2.0 also seems to be a reasonable choice.

Tab. 3.3 compares the different orders of convergence where we chose an individual lower limit for the linear regression. By the slightly higher rate of the reference *FVM* implementation it can be seen that it will converge against or overtake the other monitors at very high cell counts. That seems reasonable since the fixed grid will then also have a high resolution everywhere.

Additionally, in Tab. 3.3 we summarize the CPU time per step on an Intel i7 8550u CPU. The total simulation spanned over 60 s, hence 6000 steps. As expected, the time for the reference implementation is less, to be exact just half of the rest since there is no Remap subroutine involved. The remaining five monitors share a CPU time that is in the same order of magnitude. The slightly faster time of *Arc-Simple* in combination with a superior performance in the convergence analysis makes this monitor the best choice for this kind of scenario. It seems that the neglected square of this monitor does influence the computing time because it might damp the overall monitor values and as a consequence the subsequent iterations of the two-step scheme (3.32) may take fewer steps.

Table 3.3: Additional information on the order of convergence for the different monitor functions with $\alpha = 2.0$, $n = 64$ and at time $t = 45s$. The third column denotes the average CPU per step averaged over the entire simulation.

Monitor	Order of convergence	CPU time per step in s
(a) Arclength	$\mathcal{O}(n^{-0.84})$, $n \geq 64$	2.05×10^{-5}
(b) Curvature	$\mathcal{O}(n^{-1.12})*$, $n \geq 256$	1.83×10^{-5}
(c) Arc-Simple	$\mathcal{O}(n^{-0.98})$, $n \geq 64$	1.16×10^{-5}
(d) Arc-Curv	$\mathcal{O}(n^{-1.07})*$, $n \geq 64$	1.99×10^{-5}
(e) Art-Diff	$\mathcal{O}(n^{-1.93})*$, $n \geq 16$	2.47×10^{-5}
(ref) FVM	$\mathcal{O}(n^{-1.21})$, $n \geq 8$	9.27×10^{-6}

* = problems at higher cell counts

This could also explain the fact that the CPU time of the *Arclength*-monitor is worse than the one of *Curvature* even if the calculation of the second derivative is more complex. Finally, the monitor of *Artificial Diffusivity* takes the longest since it incorporates a more complex scheme.

3.6 Population balance of pure coagulation

In a next test, we compare the monitor functions on a case of pure coagulation. The corresponding equation is given as the PBE without the growth term but with the source integral of (2.11). We consider binary coagulation with a constant kernel of $a(v', v - v') = 1$. This concludes into the equation

$$\frac{\partial n(v,t)}{\partial t} = \frac{1}{2}\int_0^v n(v',t)n(v-v',t)\,\mathrm{d}v' - \int_0^\infty n(v',t)n(v,t)\,\mathrm{d}v'. \tag{3.57}$$

The finite volume discretization of this reaction term will yield a double integral that is generally intensive to compute. In the solver BOFFIN this case is already implemented by Liu and Rigopoulos (2018). We therefore use this code.

In the case of an exponential initial condition

$$n(v,0) := n_0(v) = e^{-x} \quad v \in \Omega_v = [10^{-3}, 5 \times 10^1] \tag{3.58}$$

the analytical solution is then given according to Scott William T. (1968) as

$$n(v,t) = \frac{4}{(t+2)^2} exp\left(\frac{-2v}{t+2}\right). \tag{3.59}$$

The domain spans over multiple orders of magnitude, thus we use the preset of the grid stretching $\alpha = 1.5$. In the previous section, the *Arc-Simple* monitor was seen to be the best in the category of the *Arclength* descendants. Therefore, the class of this kind of monitors is from now on restricted to it. We add the regular *Arclength* in the convergence analysis. The regularization constant was kept at $\beta = 10^4$. As a reference, a standard finite volume scheme as in the case of step advection is employed. However, the fixed grid is set to be logarithmically uniform.

Theoretically, the monitor based on artificial diffusivity will not work because there is no growth incorporated. However, we set the growth rate to an arbitrary number, for example, $G = 0.05\,\mathrm{s}^{-1}$ due to the fact that the big measuring for the monitor on the artificial diffusivity takes place inside

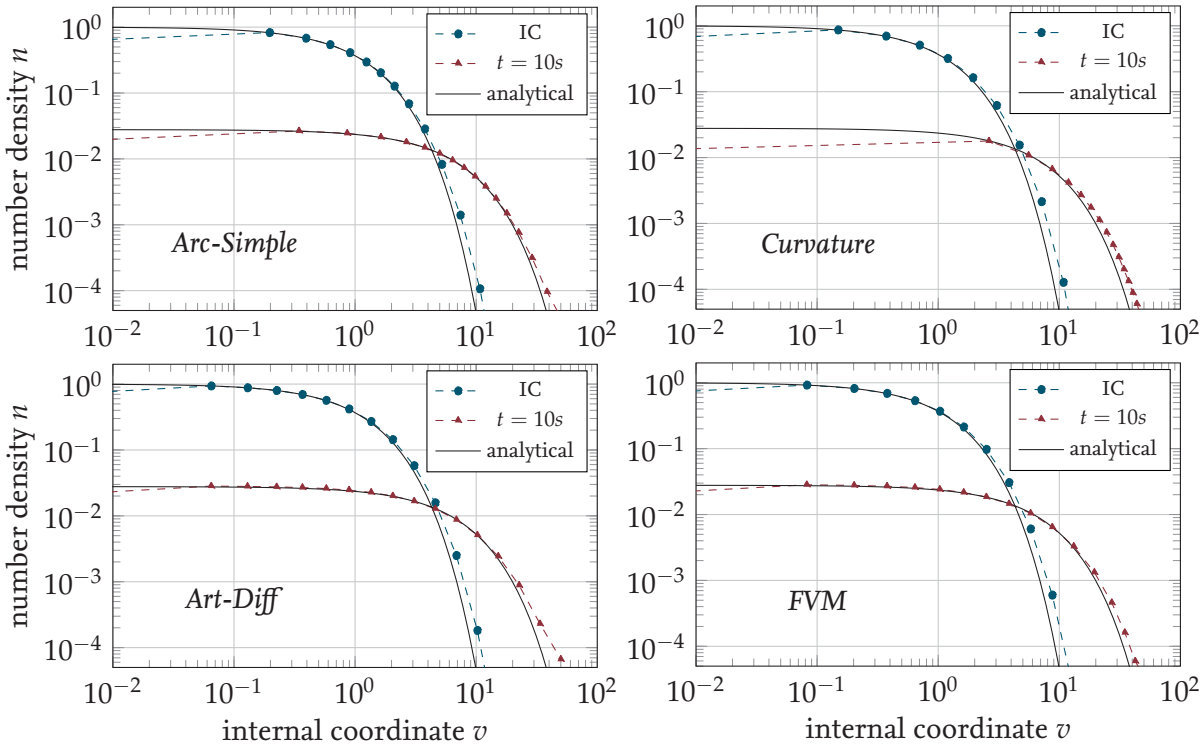


Figure 3.9: Displays how the numerical solution capture the solution at two different points time with grid stretching $\alpha = 1.5$ and 16 finite volume cells. There is also a reference FVM implementation on a fixed exponential grid.

the ratio of consecutive solution gradients r that obviously still exists. Additionally, this could be seen as the case that would occur when the solution at a given point in time would suddenly change from pure coagulation to pure growth. Since the grid spans over multiple orders of magnitude the limiter η was set to the values according to the second row of 3.2.

In Fig. 3.9 the numerical solution with 16 finite volume cells is shown for the initial condition and the solution after $t = 10\,\mathrm{s}$. Once again, the marks indicate the location of the cell faces and the respective solution values that have been linearly interpolated from the cell averaged solution. The most left point (that can not be seen in the plots) is located at the left boundary 10^{-3}.

All the monitors present a good coverage of the initial condition even though they overshoot at values of $v > 5 \times 10^1$. When it comes to the solution at later time steps, they differ. The *Curvature* monitor fails to set grid nodes to small coordinate values. That might be caused by the fact that the solution is convex over the domain, hence there is no big change in curvature. Mathematically, this can be examined by taking the second derivative w.r.t. v of (3.59) that will strongly decrease at later points in time because the denominator in the coefficient will be proportional to t^3. We can further confirm this bad performance by looking at the convergence diagram of Fig. 3.10. Here we calculate the average deviation from the analytical solution in accordance to (3.56). Given the case of an exponential initial condition within pure coagulation, the *Curvature* monitor does not converge.

The *Arc-Simple* monitor works as expected.Additionally, it does shift its grid node locations slightly to the right but given the short simulation time, it is well capable of analyzing the solution. By looking at the convergence diagram, it performs the best. However, this should not be taken for granted

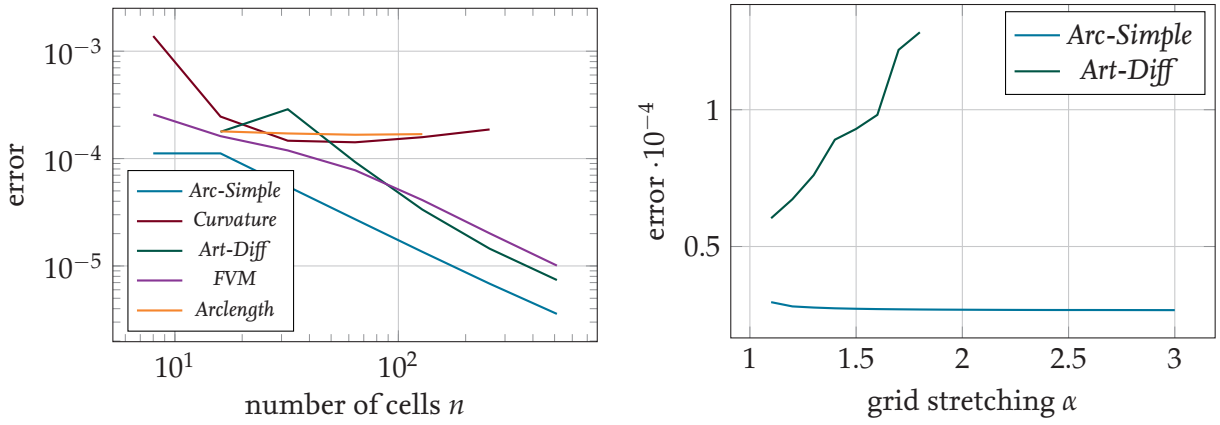


Figure 3.10: Analysis on the error (average deviation from analytical solution) over the number of finite volume cells at fixed grid stretching $\alpha = 1.5$ and over the grid stretching at 64 cells, both at time $t = 10s$.

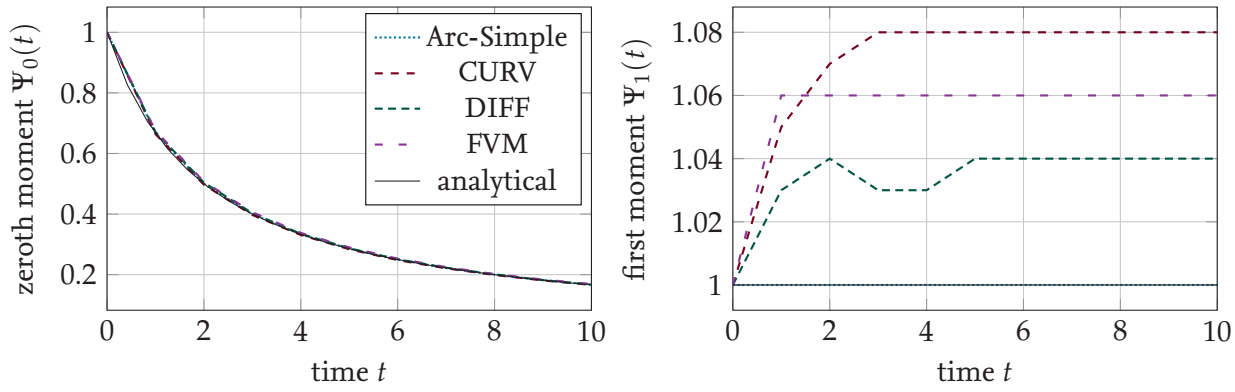


Figure 3.11: Capturing capabilities of monitors and the reference implementation for the first two moments of the solution.

since longer simulations might decrease the magnitude of arclength in the left parts of the domain and hence will make the points shift to the right like the *Curvature* monitor.

Most surprising is the performance of the monitor of *Artificial Diffusivity*. The capturing is comparable to the reference finite volume implementation and as seen in the convergence diagram it overtakes this implementation at cell counts higher than ≈ 90. As we set $\alpha = 1.5$, we chose a smaller one than in the previous section. Such a low value of α shifts the grid style more towards an uniform grid that we see in the reference implementation. This trend continues and in the case of $\alpha = 1.1$ it is close to a uniform grid. We think that this also explains the nearly similarly placed grid node locations in Fig. 3.9.

For the *Arc-Simple* monitor we can again see a robust performance over the span of the tested grid stretching, hence the choice of $\alpha = 1.5$ is justified. We also analyzed the *Curvature*-monitor on different values for α, however, the results of the error are again diverging, and since the accuracy is in a different order of magnitude, we decided to leave it out of the plot.

In contrast to the previous section, pure Coagulation involves a change in the zeroth moment while keeping the first moment fixed. By applying the appropriate integral, we can deduce the

Table 3.4: Additional information on the order of convergence for the different monitor functions with $\alpha = 1.5$, $n = 64$ and at time $t = 45s$. The third column denotes the average CPU per step over the entire simulation.

Monitor	Order of convergence	CPU time per step in s
(a) Arclength	n.c.	3.61×10^{-4}
(b) Curvature	n.c.	3.92×10^{-4}
(c) Arc-Simple	$\mathcal{O}(n^{-1.00})$, $n \geq 16$	4.37×10^{-4}
(e) Art-Diff	$\mathcal{O}(n^{-1.33})$, $n \geq 32$	4.12×10^{-4}
(ref) FVM	$\mathcal{O}(n^{-0.77})$, $n \geq 8$	4.24×10^{-4}

n.c. = not converging

following analytical expressions

$$\Psi_0(t) = \frac{2}{t+2} \quad \text{and} \quad \Psi_1(t) = 1 \quad \forall t \geq 0 \tag{3.60}$$

In Fig. 3.11 we plot the capturing of these moments. All of the tested monitors, as well as the *FVM*-implementation, yield a precise coverage of the total particle number. However, in the case of the first moment, only *Arc-Simple* fully captures the results which underlines its good performance so far. All the other monitors, as well as the *FVM*-implementation, have a slight overshoot but converge after a given time. Since this could be caused by the given code of Liu and Rigopoulos (2018) the relation to the monitor functions might not be safe.

Given the error at different number of cells, we can deduce the approximate order of convergence as listed in Tab. 3.4. The overtaking of the monitor of *Artificial Diffusivity* against the *FVM*-implementation is seen by its comparably higher convergence rate. By comparison with the order achieved in the case of step advection, this reveals a really good performance of the *Artificial Diffusivity*-monitor that, unfortunately, still heavily depends on the heuristically chosen values of the limiter η. Interesting is also the fact that the *Curvatue*-monitor, as well as the additionally tested *Arclength*-monitor, does not converge.

Moreover, in Tab. 3.4 we list the computation time necessary for the different monitors. Compared to the case of pure growth the times per step increased by one order of magnitude for all monitors we chose to analyze here. This is due to the evaluation of the double integral.

In general, the time contribution of evaluating the fractional step is higher than the time spent in the Remap subroutine. In contrast to the last section, the *FVM* reference implementation performs slightly worse than the best one which seems to be the Curvature monitor. This might also be caused by the given implementation of Liu and Rigopoulos (2018).

4 Modeling and Simulation of Cell Growth and Cell Fission

In this chapter, the adaptive mesh discretization scheme in combination with the different monitor functions is used to analyze the population dynamics of mammalian stem cells. A finite volume scheme for the contribution of breakage is derived. In a test case of simple breakage the scheme is validated with the help of analytical solutions. Then, kinetics found in the literature are transfered onto the previously analyzed numerical tool. Finally, the results of the simulation of combined breakage and growth are presented and discussed.

4.1 Considering the contribution of breakage

Similar to the aggregation term, the discretization of the breakage source term introduces a double integral. Depending on the precision chosen to compute the values, breakage can be very costly. Additional time might be necessary if the kernel values, or the kinetics in general, are difficult to evaluate.

Therefore, it is crucial that the numerical implementation of this source term is as efficient as possible. In the context of the chosen finite volume discretization, we present a scheme that is geometrically motivated. The domain of integration is analyzed and split up into basic two-dimensional geometrical object over which the integral is approximated by a set of Newton-Cotes formulae. We therefore fix the time, hence neglect any dependencies on it for the following derivations.

4.1.1 Numerical Implementation of the resulting source term

The contribution to the right-hand side of the system of ODEs of the breakage reaction term is given by (3.25). This integral is defined in the transformed τ-space, however, it would be more convenient to evaluate the integral in the real v-space since the kernel functions are given there. So we apply an integral transformation

$$\dot{S}_i := \frac{1}{\Delta\tau} \int_{\tau_{i-\frac{1}{2}}}^{\tau_{i+\frac{1}{2}}} \dot{s}(\bar{v}(\tau,t), \mathbf{y}, f(\cdot))\, \mathrm{d}\tau = \frac{1}{\Delta\tau} \int_{v_{i-\frac{1}{2}}}^{v_{i+\frac{1}{2}}} \dot{s}(v, \mathbf{y}, n(\cdot)) \cdot w(\bar{\tau}(v,t), t)\, \mathrm{d}v. \tag{4.1}$$

Over the domain of integration $[v_{i-\frac{1}{2}}, v_{i+\frac{1}{2}}]$ the Jacobian determinant $w(\bar{\tau}(v,t),t)$ is constant as $w_i = \Delta\tau/\Delta v_i$. If we factor it out, the normalization changes from $\Delta\tau$ to Δv_i. The discretized source term now becomes

$$\dot{S}_i = \frac{1}{\Delta v_i} \int_{v_{i-\frac{1}{2}}}^{v_{i+\frac{1}{2}}} \dot{s}(v, \mathbf{y}, n(\cdot))\, \mathrm{d}v \tag{4.2}$$

This one is generally valid for all kind of source phenomena. In the case of binary breakage we insert the model of (2.10). It consists of the Birth and Death of particles meaning we have two separate

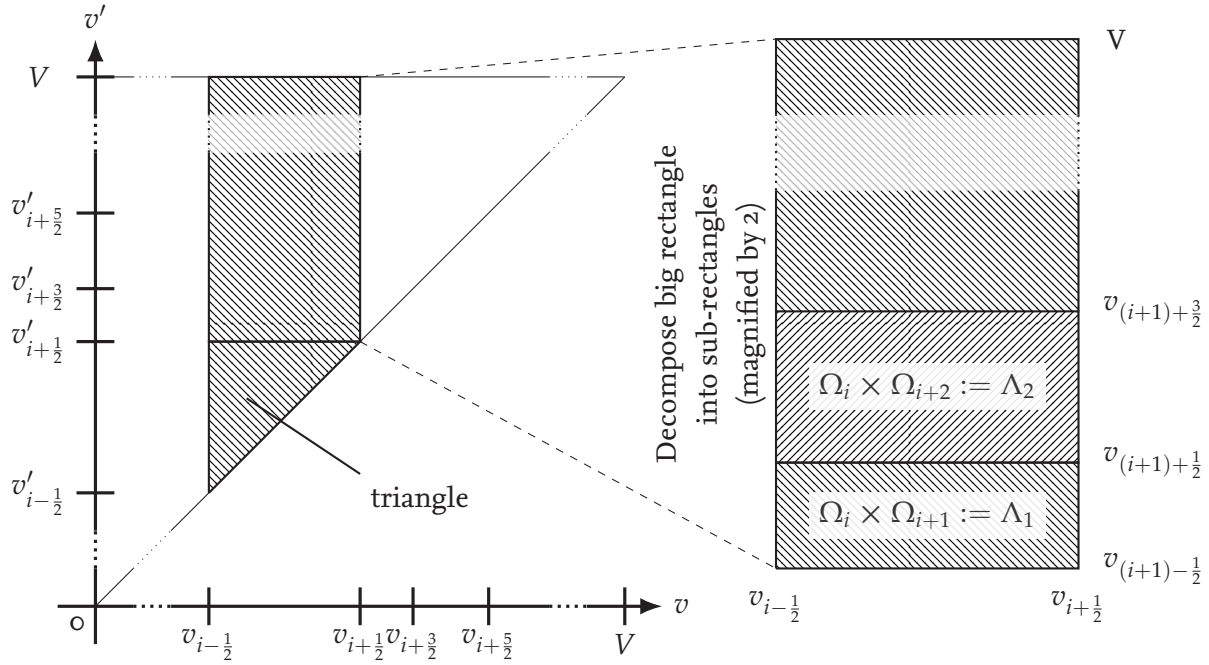


Figure 4.1: Visualization of the domain of integration. The two axes v and v' both represent the same discretized domain $\Omega_v = [0, V]$. The entire integration domain is divided up into $n-1$ trapezoids, each consisting of one triangle and $n-(i+1)$ rectangles.

integrals

$$\dot{S}_i = \underbrace{\frac{2}{\Delta v_i}\int_{v_{i-\frac{1}{2}}}^{v_{i+\frac{1}{2}}}\int_{v}^{V} n(v')\gamma(v',\mathbf{y})p(v,v')\,\mathrm{d}v'\mathrm{d}v}_{\mathcal{B}_i} + \underbrace{\frac{1}{\Delta v_i}\int_{v_{i-\frac{1}{2}}}^{v_{i+\frac{1}{2}}} n(v)\gamma(v,\mathbf{y})\,\mathrm{d}v}_{\mathcal{D}_i}. \tag{4.3}$$

Whereas the evaluation of the contribution of death $\mathcal{D}$ is a straightforward numerical integration (e.g., by using the trapezoidal rule)

$$\mathcal{D}_i = N_i\frac{\gamma(v_{i+\frac{1}{2}},\mathbf{y}) + \gamma(v_{i-\frac{1}{2}},\mathbf{y})}{2}, \tag{4.4}$$

the Birth term incorporates a double integral. Hereby the domain of integration is a trapezoid that can be decomposed into one triangle and a big rectangle. This can be seen on the left-hand side of Fig. 4.1. Mathematically, this results in a partitioning of the interval of the inner integral $v' \in [v, V]$ into two adjacent sub-intervals $v' \in [v, v_{i+\frac{1}{2}}]$ (with $v \in [v_{i-\frac{1}{2}}, v_{i+\frac{1}{2}}]$) and $v' \in [v_{i+\frac{1}{2}}, V]$. Consequentially, the integral of the contribution of Birth is split up into two separate parts.

$$\mathcal{B}_i = 2\Bigg\{\underbrace{\frac{1}{\Delta v_i}\int_{v_{i-\frac{1}{2}}}^{v_{i+\frac{1}{2}}}\int_{v}^{v_{i+\frac{1}{2}}} n(v')\gamma(v',\mathbf{y})p(v,v')\,\mathrm{d}v'\mathrm{d}v}_{\mathcal{B}_i^{triangle}} + \underbrace{\frac{1}{\Delta v_i}\int_{v_{i-\frac{1}{2}}}^{v_{i+\frac{1}{2}}}\int_{v_{i+\frac{1}{2}}}^{V} n(v')\gamma(v',\mathbf{y})p(v,v')\,\mathrm{d}v'\mathrm{d}v}_{\mathcal{B}_i^{rectangle}}\Bigg\} \tag{4.5}$$

Going further, the big interval of the inner integral can be naturally decomposed by the employed FV-discretization. This means the big rectangular domain of integration $\Omega_i \times [v_{i+\frac{1}{2}}, V]$ is sliced

into $n-(i+1)$ rectangular sub-domains Λ_j. Here we introduce a new index j that starts at $i+1$ and goes up to n, whereas the locations of the so created grid points $v_{j\pm\frac{1}{2}}$ obviously coincides with the ones down on the first discretization, simply shifted by one index point. This context can also be seen on the right-hand side of Fig. 4.1. In general, the resulting sub-rectangles will not be squares because of $\Delta v_i \neq \Delta v_j$.

Now the domain of integration has been analyzed and simplified. The next step is to solve the remaining two integrals. Here, we start with the Birth on the rectangular sub-domains. The limits of integration on a rectangle do not depend on each other. This allows for a change in the order of integration. Additionally, in the integrand only the partitioning probability $p(v,v')$ depends on v, the variable of the outer integral, leaving it solely exposed to it. Using the trapezoidal rule, the outer integral can be evaluated independently.

$$\begin{aligned}
\mathcal{B}_i^{rectangle} &= \frac{1}{\Delta v_i}\int_{v_{i-\frac{1}{2}}}^{v_{i+\frac{1}{2}}}\sum_{j=i+1}^{n}\int_{v_{j-\frac{1}{2}}}^{v_{j+\frac{1}{2}}} n(v')\gamma(v',\mathbf{y})p(v,v')\,\mathrm{d}v\,'\mathrm{d}v \\
&= \sum_{j=i+1}^{n}\int_{v_{j-\frac{1}{2}}}^{v_{j+\frac{1}{2}}} n(v')\gamma(v',\mathbf{y})\frac{1}{\Delta v_i}\int_{v_{i-\frac{1}{2}}}^{v_{i+\frac{1}{2}}} p(v,v')\,\mathrm{d}v\,\mathrm{d}v' \\
&= \sum_{j=i+1}^{n}\int_{v_{j-\frac{1}{2}}}^{v_{j+\frac{1}{2}}} n(v')\gamma(v',\mathbf{y})\frac{p(v_{i+\frac{1}{2}},v')+p(v_{i-\frac{1}{2}},v')}{2}\,\mathrm{d}v'
\end{aligned} \tag{4.6}$$

The normalization cancels with the distance employed by the trapezoidal rule.

In a last step, the former inner integral is solved by trapezoidal rule. Consider that $n(v')$ is constant on the sub-domain of integration Ω_j as N_j. The last real cell Ω_i does not have a rectangle on top, it only spans a triangle. So there is no contribution by it.

$$\mathcal{B}_i^{rectangle} = \begin{cases} \displaystyle\sum_{j=i+1}^{n}\Delta v_j\frac{N_j}{4}\Big[\gamma(v_{j+\frac{1}{2}},\mathbf{y})\Big(p(v_{i+\frac{1}{2}},v_{j+\frac{1}{2}})+p(v_{i-\frac{1}{2}},v_{j+\frac{1}{2}})\Big)+ \\ \qquad\qquad \gamma(v_{j-\frac{1}{2}},\mathbf{y})\Big(p(v_{i+\frac{1}{2}},v_{j-\frac{1}{2}})+p(v_{i-\frac{1}{2}},v_{j-\frac{1}{2}})\Big)\Big] & i<n \\ 0 & i=n \end{cases} \tag{4.7}$$

The triangle part is not as easy to integrate but following the basic idea of Newton-Cotes numerical integration schemes, we can define an approximation of the integrand over the triangle by linear shape functions on all three vertices. This refers to a plain that is point-wise exact. The integral will yield the volume under this construction.

To do this, we consider a mapping of the isosceles triangle into a parent space as seen in 4.2. Over the domain of integration $n(v')$ will be constant N_i. The area of the triangle in transformed space is $1/2$. Due to the linear transformation into parent space the Jacobian determinant will simply become $\left|\frac{\partial(v,v')^T}{\partial(\xi_1,\xi_2)^T}\right| = 2A$, with A the area of the real triangle.

$$\mathcal{B}_i^{triangle} = \frac{N_i}{\Delta v_i}\int_0^1\int_0^{1-\xi_1}\gamma\big(\bar{v}'(\xi_1,\xi_2),\mathbf{y}\big)\,p\big(\bar{v}(\xi_1,\xi_2),\bar{v}'(\xi_1,\xi_2)\big)2A\,\mathrm{d}\xi_2\,\mathrm{d}\xi_1 \tag{4.8}$$

We can now approximate the integrand by a superposition of shape functions that have the value

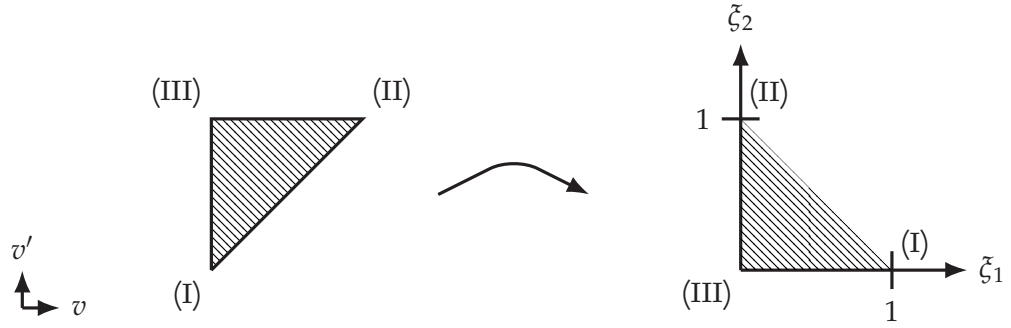


Figure 4.2: Transformation of the general isosceles triangle into parent space.

one at the designed node and will be zero at every other vertex of the triangle

$$\mathcal{B}_i^{triangle} = 2A\frac{N_i}{\Delta v_i}\int_0^1\int_0^{1-\xi_1}\left(\gamma(v'_{i-\frac{1}{2}},\mathbf{y})p(v_{i-\frac{1}{2}},v_{i-\frac{1}{2}})\cdot\xi_1\right)+\left(\gamma(v'_{i+\frac{1}{2}},\mathbf{y})p(v_{i+\frac{1}{2}},v_{i+\frac{1}{2}})\cdot\xi_2\right)+\left(\gamma(v'_{i+\frac{1}{2}},\mathbf{y})p(v_{i-\frac{1}{2}},v_{i+\frac{1}{2}})\cdot(1-\xi_2-\xi_1)\right)\,\mathrm{d}\xi_2\,\mathrm{d}\xi_2. \tag{4.9}$$

Integration with the area of an isosceles triangle being calculated as $A = (\Delta v_i)^2/2$ yields

$$\mathcal{B}_i^{triangle} = N_i\frac{\Delta v_i}{6}\Big[\underbrace{\gamma(v_{i-\frac{1}{2}},\mathbf{y})p(v_{i-\frac{1}{2}},v_{i-\frac{1}{2}})}_{(I)}+\underbrace{\gamma(v_{i+\frac{1}{2}},\mathbf{y})p(v_{i+\frac{1}{2}},v_{i+\frac{1}{2}})}_{(II)}+\underbrace{\gamma(v_{i-\frac{1}{2}},\mathbf{y})p(v_{i-\frac{1}{2}},v_{i+\frac{1}{2}})}_{(III)}\Big]. \tag{4.10}$$

In conclusion, the total contribution of the breakage reaction term on cell i is

$$\frac{1}{\Delta v_i}\int_{v_{i-\frac{1}{2}}}^{v_{i+\frac{1}{2}}} \dot{s}(v,\mathbf{y},n(\cdot))\,\mathrm{d}v := \dot{S}_i = \mathcal{D}_i + 2\left(\mathcal{B}_i^{triangle}+\mathcal{B}_i^{rectangle}\right). \tag{4.11}$$

For a high efficiency, it is useful to fetch all necessary kernel values in advance of each fractional step since, for example, the value at one vertex of the sub-rectangles is also used by up to three other rectangles. This additionally comes in handy in the case of complex kinetics as presented in the section on cell growth and cell fission.

4.1.2 Validation on pure breakage

In this section, we test our finite volume scheme for breakage with simple kinetics. Additionally, we sum up the differences between the various monitors compared to the inverse phenomenon of coagulation presented in the previous chapter.

Therefore, a case of pure breakage is considered for which analytical solutions are available. Starting off with the source term of (2.10), the dependency of the division rate on any additional scalars $\mathbf{y}(t)$ is neglected. By keeping the size-dependent kernel functions, the population balance equation for pure breakage reads

$$\frac{\partial n(v,t)}{\partial t} = \int_v^V n(v',t)\gamma(v')p(v,v')\,\mathrm{d}v' + n(v,t)\gamma(v) \quad v\in\Omega_v,\ t>0. \tag{4.12}$$

Table 4.1: Analytical solutions of pure breakage for two different cases due to Ziff and McGrady (1985).

Case	Domain Ω_v	$\Gamma(v,\mathbf{y})$	$p(v,v')$	$n_0(v)$	$n(v,t)$	$\Psi_0(t)$	$\Psi_1(t)$
(1) Lin	$[10^{-6},10^2]$	v	$1/v'$	e^{-v}	$(1+t)^2e^{-v(1+t)}$	$t+1$	1
(2) Quad	$[10^{-4},10^2]$	v^2	$1/v'$	e^{-v}	$[1+2t(1+v)]e^{-v(tv+1)}$	*	1

* no closed form available

We imprint the exponential initial condition

$$n(v,0) := n_0(v) = exp(-v) \quad v \in \Omega_v, \tag{4.13}$$

that was already used in the case of pure coagulation.

In the context of depolymerization Ziff and McGrady (1985) derived an analytical solution. They have been adapted and simplified by Qamar et al. (2009) which we will use here. Tab. 4.1 collects the analytical solutions as well as the domains that differ between the Linear Case (1) and the Quadratic Case (2). By (2.8) the moments due to the analytical solution can be deduced that are found in the two right columns of the table.

Naturally, breakage forms the inverse operation to coagulation. It increases the total amount of particles while keeping the total particle mass fixed. In Fig. 3.9, the initial profile was smeared to the right as the coagulation created particles of greater volume. For pure breakage, we expect the opposite. The initial profile will be compressed and shifted to the left. In the limit, only particles of volume close to zero exist.

Since the initial condition is the same, we do not print the capturing by the different monitors again. Fig. 4.3 just presents the solution at the end time for both test cases. In accordance to the coagulation case, the *Arc-Simple*-monitor again showed the best performance. Its points reconstruct the analytical solution the best. However, the monitor is incapable of capturing points below a threshold of a certain number of cells. It improves when the number of cells increases as illustrated in the convergence diagram of Fig. 4.4. The reference *FVM*-implementation on a fixed exponential grid does obviously not experience this kind of lower threshold.

Due to the fixed halo cell at the very left, a significant mismatch of the *Arc-Simple*-monitor as well as the monitor *Curvature* can be seen. By the same argumentation as in the section of coagulation, this gap naturally increases stronger for the *Curvature*-monitor which can be retraced by taking the second derivative w.r.t. v of the analytical solution.

Despite this gap, the accuracy of *Arc-Simple* is still better than the *FVM*-implementation. A reason for this is the higher node density in the region the solution experiences its steep gradient. Points left of this location fall under small scales in internal coordinate and count less when it comes to the calculation of the average deviation.

Generally speaking, low cell counts are not well captured by the proposed breakage scheme. The overshoot of the *FVM*-reference implementation is directly visible in the case of 32 finite volume cells. This might explain the use of the conservation form that can be seen in Qamar et al. (2009). However, for more cells, the scheme shows a good representation of the moments. The right side of Fig. 4.3 presents the plot of the moments for the case of 64 finite volume cells.

The monitor of *Curvature* again shows an inconsistent performance. It is only applicable for a certain number of finite volume cells. Above this upper limit, the iteration of the two-step scheme

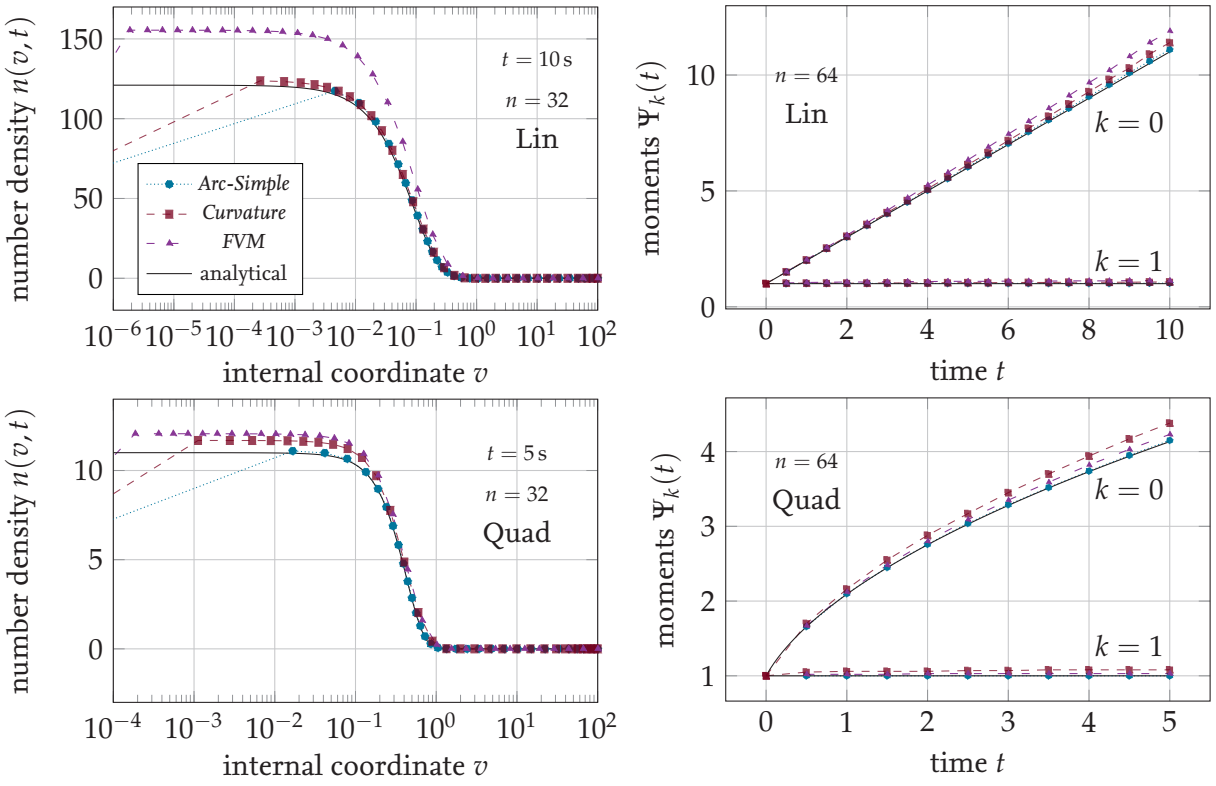
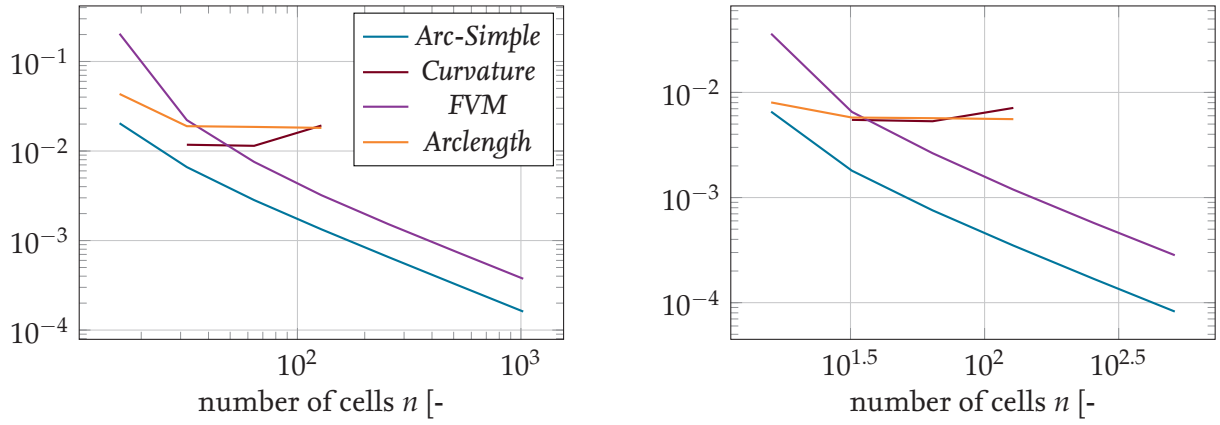


Figure 4.3: Capturing capability of the distribution and the first two moments of the tested monitors. The top row presents the case (1) of a linear division rate and the bottom row the case (2) of a quadratic division rate.



(a) Linear case (1)

(b) Quadraric case 2

Figure 4.4: Convergence diagram for the average deviation from analytical solution for both cases at a grid stretching of $\alpha = 1.5$.

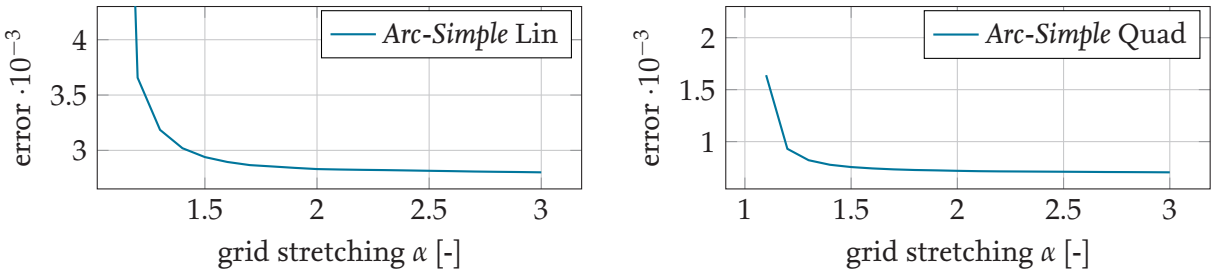


Figure 4.5: Convergence diagrams for the average deviation from analytical solution over the grid stretching α for both cases at 64 finite volume cells.

Table 4.2: Information on the order of convergence at a grid stretching of $\alpha = 1.5$ and CPU time per step with 64 finite volume cells averaged over all steps.

	Order of convergence		CPU time per step in s	
Monitor	(1) Lin	(2) Quad	(1) Lin	(2) Quad
Arclength	n.c.	n.c.	1.07×10^{-4}	9.13×10^{-4}
Arc-Simple	$\mathcal{O}(n^{-1.06})$, $n \geq 32$	$\mathcal{O}(n^{-1.11})$, $n \geq 32$	1.11×10^{-4}	1.97×10^{-3}
Curvature	n.c.	n.c.	1.04×10^{-4}	1.63×10^{-3}
FVM (ref)	$\mathcal{O}(n^{-1.08})$, $n \geq 64$	$\mathcal{O}(n^{-1.13})$, $n \geq 32$	1.23×10^{-4}	1.97×10^{-3}

n.c. = not converging

(3.32) exceeds its maximum steps. By the four points we were able to compute, the monitor seems to be converged. However, the performance is comparably bad.

On top of that, we performed a test on the monitor of *Artificial Diffusivity* but in contrast to its good results in the case of pure coagulation, it was not able to capture the pure breakage case. Even if both examples share the same initial condition, their underlying mechanics are only similar in theory. For the case of pure coagulation, we assumed the kernel function to be a constant. Here however, both kernels are size-dependent. Together with the problem of heuristically fitting the limiter η, the performance of the monitor of *Artificial Diffusivity* is inconsistent, given the artificial test cases we used it in.

With Fig. 4.5 and the analyses made in the previous chapter, we conclude that there is a limit for the grid stretching α above which the solution error converges for the *Arc-Simple* monitor. Again, we recommend to not set it too high since it would make the grid sluggish.

In Tab. 4.2 we summarize the convergence order, if available. As already mentioned in the previous test cases, the slightly faster convergence of the *FVM*-implementation indicates a point of very high number of cells at which both accuracies coincide.

4.2 Simulation of the proliferation of mammalian stem cells

Generally speaking, the application of the population balance equation to the simulation of cell populations was an early transfer of the idea. With Hulburt and Katz (1964) introducing the concept of the PBE and the layout of the various source terms in the field of crystallization, Fredrickson et al. (1967) continued by changing the interpretation into a biological spectrum. Many of the components can be defined there and, by design, share a similar stochastic background. In addition to

this first article, Fredrickson (2003) reviewed the first concepts about 35 years later. Here, he also adapted the concept of a multistage PBE to account for the different life phases a cell passes. Basically, the idea is to solve multiple population balance equations in conjunction, all responsible for the different stages in the life of a cell. The transition between two phases is modeled akin to nucleation by a Birth and Death term at the source of each equation. Breakage then reduces the number of cells in the last stage and increases it in the first one. Overall, multistage PBEs allow for a more precise representation of complex population dynamics.

Here, we restrict ourselves to a single stage PBE and use deterministically modeled constitutive parameters (growth rate, division rate, and partitioning probability) that are all size-dependent.

4.2.1 Kinetics of cell fission and growth in the context of the PBE

The task of the next two subsections is to test our explicit adaptive grid scheme and the various monitor functions on an example with realistic kinetics. The simulation of the evolution of sheep bone stem cells we are about to set up incorporates two phenomena simultaneously, growth and breakage. The latter one is called cell fission in this biological context. For the underlying model, the ideas of Mantzaris et al. (1999) are used and for the necessary parameters we refer to Mancuso et al. (2009) and the references therein. The latter one has been chosen since many of the parameters are tuned by real-world experimental data of sheep-bone stem cells evolving in a petri-dish.

In contrast to the literature cited in which the model was mass-based, the internal coordinate will be cell volume $v = [\mathrm{mm}^3]$. However, since the mass density of the cells is assumed to be constant, it is possible to easily switch between both representations. The unit of the number density is then set to

$$n(v,t) = \left[\frac{\textit{number of cells}}{\textit{cell culture volume} \times \textit{internal coordinate}}\right] = \left[\frac{1}{\mathrm{mm}^3 \times \mathrm{mm}^3}\right] = \left[\frac{1}{\mathrm{mm}^6}\right]. \tag{4.14}$$

These biological phenomena span over a much broader period of time, hence the unit of time is chosen to be hours $t = [h]$. In all plots, the axes are scaled to be in the measure of days $[d]$.

To a good approximation, the cells can be seen as spheres. Therefore, we can deduce the relationship between the radius and the volume to

$$r = \left(\frac{3v}{4\pi}\right)^{\frac{1}{3}}. \tag{4.15}$$

Three kinetic relations have to be determined, the two kernel functions and an expression for the growth rate. We start off with the latter one. Growth in general is limited by the space a population is allowed to develop in. In the case of mammalian stem cells growing in a petri-dish, this area is a limiting factor. To incorporate the idea, the area currently occupied on this dish has to be examined. We therefore assume that the cell culture only consists of one layer.

Mancuso et al. (2009) introduced a bi-dimensional projection that approximately resembles the area a each spherical cell and its interstices occupy to

$$a = \left(\frac{4}{\pi}\right) r = \sqrt[3]{\frac{36}{\pi}} v^{\frac{2}{3}}. \tag{4.16}$$

To get the total area the entire population occupies, we integrate over the domain of all possible cell volumes $v \in \Omega_v$ with their respective number density $n(v,t)$ and multiply by the total culture

volume V. This yields a somehow modified 2/3-moment of the number density

$$\phi(t) = V\sqrt[3]{\frac{36}{\pi}} \int_{\Omega_v} v^{2/3} n(v,t)\, \mathrm{d}v \approx V\sqrt[3]{\frac{36}{\pi}} \sum_{i=1}^{n} \frac{3N_i}{5} \left[v_{i+\frac{1}{2}}^{5/3} - v_{i-\frac{1}{2}}^{5/3} \right]. \tag{4.17}$$

With the help of the occupied area, a geometrical limiting factor $L(t)$ can be defined by using the petri-dish area A

$$L(t) = \left[1 - \frac{\phi(t)}{A} \right]^{\omega}. \tag{4.18}$$

This power-law equation is outfitted with the tuned exponent ω.

Another limiting aspect of the growth rate would be the dependencies on additional scalars $\mathbf{y}(t)$ that could incorporate, e.g., nutrient concentrations. In the case of sheep bone stem cells, Mancuso et al. (2009) set this variable to the concentration of oxygen C_{O_2}. However, given the laboratory conditions, it is assumed to be a constant value.

For the aspects of growth, two phenomena are considered. The anabolic part will describe the transformation of nutrients into biomass, hence it adds to the growth rate. On the other side, there is the catabolic part that defines the nutrient consumption, i.e., how much of its available resources the cell uses to produce its necessary energy. Consequentially, this part reduces the growth rate. The latter one is set proportional to the cell volume since heavier cells need to consume more nutrients. Anabolism is set to be proportional to the cells surface o. This yields

$$G = C_1 \cdot o - C_2 \cdot v. \tag{4.19}$$

By the relation between the surface of a sphere and its volume

$$o = \frac{3}{r} v = 3v \left(\frac{4\pi}{3v} \right) = \sqrt[3]{36\pi} \cdot v^{2/3}, \tag{4.20}$$

the growth rate is given as

$$\begin{aligned} G(v, C_{O_2}) = \underbrace{\sqrt[3]{36\pi} \cdot v^{2/3} \cdot \lambda_{max}}_{\text{maximum volumetric growth rate}} \cdot \underbrace{\frac{C_{O_2}}{C_m + C_{O_2}} L(t)}_{\text{limiters}} - \underbrace{g_{cat} v}_{\text{catabolic consumption}} = \\ \underbrace{\sqrt[3]{36\pi} \cdot \lambda_{max}}_{\text{anabolic rate } \lambda_{an}} \cdot \underbrace{\frac{C_{O_2}}{C_m + C_{O_2}}}_{\text{oxygen limiter } \theta_{O_2}} \cdot L(t) \cdot v^{2/3} - g_{cat} \cdot v. \end{aligned} \tag{4.21}$$

Here, we introduce the anabolic rate λ_{an}. This naturally captures the effects of the logistical behaviour since given a high occupied area $\phi(t)$ the geometric limiter $L(t)$ will be small and presumably the catabolic part will be predominant, resulting in lower growth rates or even negative growth rates, i.e., the cell shrinks.

Given the model for the growth rate, we now define the two kernel functions. The division rate $\gamma(v, C_{O_2})$ describes the amount of cells that will commit fission in a given time $\gamma = [1/h]$. The deterministic aspect of this phenomenon is related to the growth rate, making the division rate proportional to it. Additionally, we have to incorporate a stochastic component. That can be done by implementing a Gaussian normal distribution around a given cell volume μ that shows the highest

Table 4.3: Kinetic parameters used in the simulation. They are based on Mancuso et al. (2009) and the references therein. We translated them from a mass-dependent formulation to a volume-based one.

Name	Parameter	Value	Unit
Concentration of oxygen	C_{O_2}	2.03×10^{-7}	$\mathrm{mmol/mm^3}$
C_{O_2} at half maximal consumption	C_m	6.0×10^{-9}	$\mathrm{mmol/mm^3}$
Mass density of given cells	ρ_c	1.14×10^{6}	$\mathrm{ng/mm^3}$
Partitioning parameter	q	39	—
Total culture volume	V	800	$\mathrm{mm^3}$
Power-Law exponent	ω	10.8	—
Petri-dish area	A	800	$\mathrm{mm^2}$
Initial number of cells	N_0	800	—
Mean of Gaussian	μ	3.333×10^{-6}	$\mathrm{mm^3}$
Standard deviation of Gaussian	σ	9.8684×10^{-7}	$\mathrm{mm^3}$
Maximum growth rate	λ_{max}	8.54386×10^{-5}	$\mathrm{mm^3/(mm^2 h)}$
Beta-Function	$\beta(q+1, q+1)$	$4.650850914 \times 10^{-25}$	—
Oxygen Limiter	θ_{O_2}	0.971291	—
anabolic rate	λ_{an}	4.13179×10^{-4}	$\mathrm{mm^3/(mm^2 h)}$
catabolic rate	g_{cat}	1.0×10^{-3}	$1/\mathrm{h}$

probability to commit fission. Since the integral over this distribution does not encompass $(-\infty, \infty)$ the distribution has to be normalized. Given the Gaussian

$$\mathcal{N}(v|\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\left(\frac{-(v-\mu)^2}{2\sigma^2}\right), \tag{4.22}$$

we set the division rate to be proportional to the growth rate and the normalized distribution

$$\gamma(v, C_{O_2}) = G(v, C_{O_2}) \cdot \frac{\mathcal{N}(v|\mu,\sigma)}{1 - \int_0^v \mathcal{N}(\tilde{v}|\mu,\sigma)\ \mathrm{d}\tilde{v}}. \tag{4.23}$$

Here, it is possible to approximate the integral over the Gaussian with the help of the error function

$$\int_0^v \mathcal{N}(\tilde{v}|\mu,\sigma)\ \mathrm{d}\tilde{v} = \frac{1}{2}\left(erf(\frac{v-\mu}{\sqrt{2}\sigma}) - erf(\frac{-\mu}{\sqrt{2}\sigma})\right). \tag{4.24}$$

The last part is to model the partitioning probability $p(v', v) = [-]$ that defines how the biomass (or to say the volume) is divided between the two daughter cells arising in fission. Mancuso et al. (2009) followed the idea of an unequal partitioning function, first proposed by Hatzis et al. (1995),

$$p(v, v') = \frac{1}{\beta(q+1, q+1)} \frac{1}{v'} \left(\frac{v}{v'}\right)^q \left(1 - \frac{v}{v'}\right)^q. \tag{4.25}$$

Here, q is a tuned parameter which they set to 39. $\beta(\xi, \xi)$ is the Beta-function that is symmetrical due to identical arguments. It is defined with the help of Gamma-Function as

$$\beta(\xi, \xi) = \frac{\Gamma(\xi)^2}{\Gamma(2\xi)} \quad \text{with} \quad \Gamma(\xi) = \int_0^\infty s^{\xi-1} e^{-s}\ \mathrm{d}s. \tag{4.26}$$

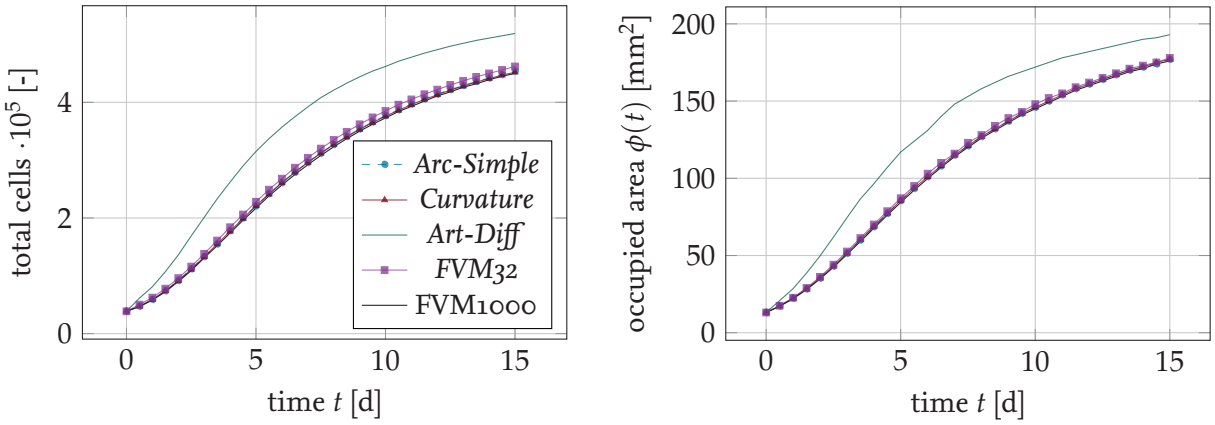


Figure 4.6: Capturing of the zeroth moment (total number of cells) and 2/3-moment (area occupied on the petri-dish) with 32 finite volume cells and grid stretching of $\alpha = 2.0$ against a FVM1000 reference solution.

The kinetic parameters used are summarized in Tab. 4.3.

For the domain, we follow Mancuso et al. (2009) and set the upper limit to $m = 16ng$ which can be translated into our context to approximately yield the domain with appropriate boundary conditions to

$$v \in \Omega_v = [0, 1.5 \times 10^{-5}] \quad \text{with} \quad n(0,t) = 0 \; \forall t \geq 0. \tag{4.27}$$

As for the initial condition, a log-normal distribution

$$n(v, t=0) := n_0(v) = \frac{N_0}{V} \frac{1}{\sqrt{2\pi}\sigma_0 \cdot v} e^{-\frac{1}{2}\left(\frac{ln(\rho_{cell} v) - ln(\mu_0)}{\sigma_0^2}\right)^2} \tag{4.28}$$

was used in which we applied the following presets for the mean and standard deviation (not to confuse with those used for calculating the division rate)

$$\mu_0 = 2ng \quad \text{and} \quad \sigma_0 = 0.4ng. \tag{4.29}$$

The mean μ_0 is the averaged cell mass Mancuso et al. (2009) observed at the beginning of their in-vitro experiments.

The numerical implementation of these kinetics is straightforward. We used the built-in *erf*-function of the gfortran compiler. The initial condition was imprinted with the help of the trapezoidal rule.

4.2.2 Results and Interpretation

In the case of cell proliferation in a petri-dish, we expect a sigmoidal behaviour meaning that the total cell mass (first moment) as well as the total cell count (zeroth moment) will converge against an upper limit if the simulation takes long enough.

For our purpose, we want to test the explicit adaptive grid scheme with the different monitors from the literature and the one proposed. Fortunately, the monitor on *Artificial Diffusivity* can be used with size-dependent growth here.

As a reference solution, a fixed uniform grid FVM-implementation with 1000 finite volume cells was used that we call FVM1000. For the simulation time, we followed Mancuso et al. (2009) and

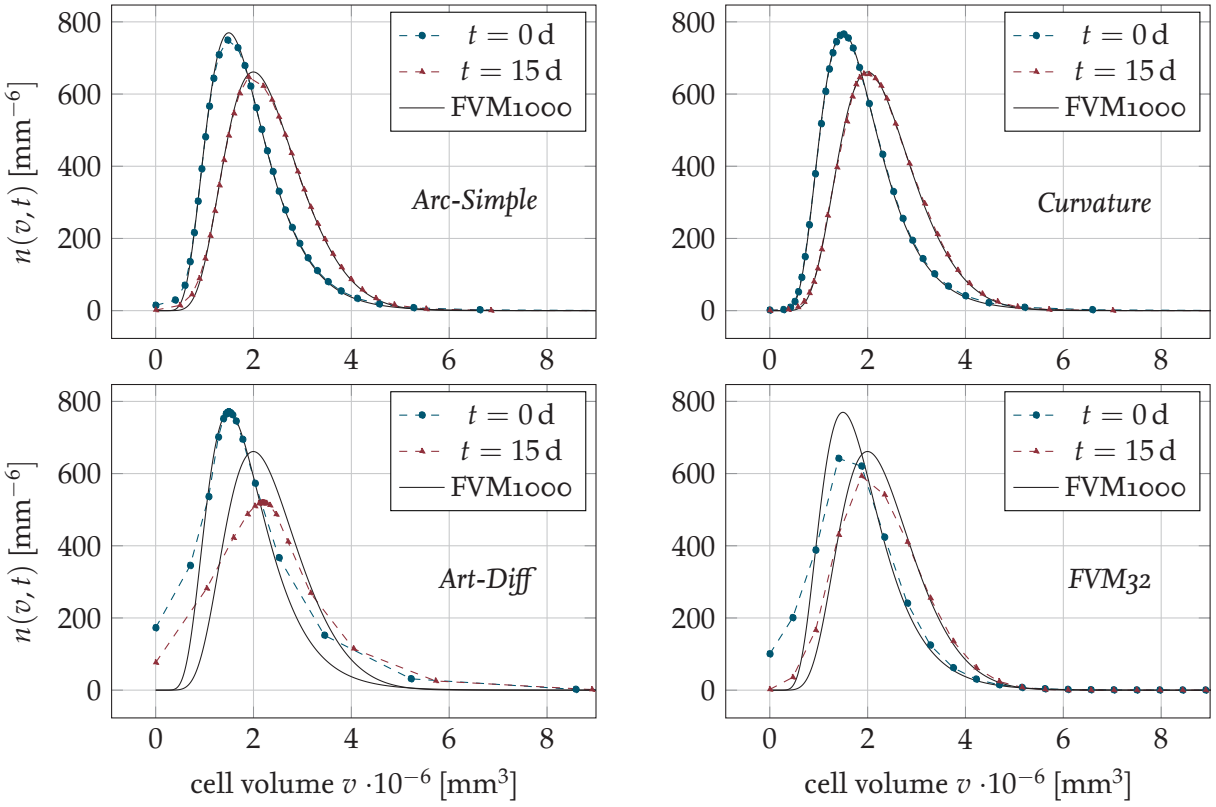


Figure 4.7: Capturing of the solution at beginning and end of cell proliferation, normalized by the current total amount of cells with a grid stretching of $\alpha = 2.0$.

set it to $t = 15\,\mathrm{d} = 360\,\mathrm{h}$. With the unchanged time horizon of $t_k = 1/100\,\mathrm{h}$ this results in 36000 computed time steps. Since the domain only spans over two orders of magnitude the grid stretching is set to $\alpha = 2.0$ for the time being. Similar to breakage, the monitors have been restricted to *Arc-Simple*, *Curvature*, *Art-Diff* and a *FVM32* standard implementation.

In Fig. 4.6 we present the coverage of the moments with the various monitor functions at 32 cells against our reference solution. Compared to the analyzes made in Mancuso et al. (2009) our results differ. To our best knowledge, we assume that the calculation of the occupied area $\phi(t)$ made within their simulation comprises a mistake. This can be retraced by analytically solving for the limiter $L(t)$ with the help of the initial condition and comparing with their plotted findings. To this end, we decide to omit an inclusion of their results and solely summarize the performance against the FVM1000-solution.

Generally speaking, the classical implementation of *FVM32* shows a reasonably good performance for only using 32 fixed uniformly spaced finite volume cells. *Arc-Simple* and *Curvature* render almost similar results and are, as expected, slightly better than *FVM32*. That is due to the improved capturing of the one mode of the distribution. The monitor of *Artificial Diffusivity* overshoots both moments. A reason for this is found in the plots of Fig. 4.7. The monitor sets its nodes only at the top of the distribution hence being incapable of resolving the other regions precisely enough. All the other representations capture the state of balanced growth well. As in Mancuso et al. (2009) this point in time is reached when cell growth and cell fission form a state of equilibrium. The initial

Table 4.4: Information on accuracy and timing for each tested monitor. Deviation was calculated with $n = 32$ and $\alpha = 2.0$ against the FVM1000 reference solution. CPU time per step presents an average over all 36000 steps undertaken.

Monitor	Average deviation in mm^{-6}	CPU time per step in s
Arclength	1.918168×10^{6}	3.98×10^{-4}
Arc-Simple	1.646119×10^{6}	3.85×10^{-4}
Curvature	2.029393×10^{6}	3.87×10^{-4}
Art-Diff	1.175658×10^{7}	3.86×10^{-4}
FVM32	5.475675×10^{6}	3.85×10^{-4}
FVM1000	—	3.48×10^{-1}

profile with its original feature at $v \approx 1.75 \times 10^{-6}\,\mathrm{mm}^3$ is shifted only slightly to the right at the end of the proliferation when normalizing by the current total number of cells.

Given the FVM reference solution at 1000 cells, we can calculate an average deviation from it for all the monitors similar to the L_1-error used in the previous examples

$$error(t) = \frac{1}{1.5 \times 10^{-5}} \int_0^{10^{-5}} \left| n^{numerical}(v,t) - n^{FVM1000}(v,t) \right| \mathrm{d}v, \tag{4.30}$$

where both solutions are constructed based on linear approximation between the discrete values.

In Tab. 4.4 the different accuracies are summarized in the case of 32 finite volume cells. It has to be considered that the shapes presented in Fig. 4.7 may induce different orders of magnitude. However, the solutions are not normalized by their zeroth moment before calculating this deviation. Recall that the unit of the deviation is the unit of the number density distribution $error = [\mathrm{mm}^{-6}]$. To get the deviation in, e.g., total cells a multiplication by $800\,\mathrm{mm}^3 \cdot 1.5 \times 10^{-5}\,\mathrm{mm}^3$ would be necessary. However, this data still matches the plots of Fig. 4.7 and once again favors the *Arc-Simple*-monitor. We can conclude that, based on our tests, the *Arc-Simple*-monitor is applicable for all scenarios and presented the best overall performances. By contrast, the monitor of *Artificial Diffusivity* is about one order of magnitude worse. Maybe it has to be adapted for the case of multiple phenomena occurring simultaneously.

Aside from the deviations, the CPU time per step is also denoted in Tab. 4.4. The durations for every implementation are almost similar, leading to the conclusion that for more sophisticated simulations in which the evaluation of the kinetics is more intensive the additional time the remap subroutine takes is neglectable. However, the solution becomes more precise given a reasonable choice of the monitor function. The accuracy improves by up to three times in this given scenario.

Fig. 4.8 shows that there are still some fluctuations over the grid stretching α for the *Curvature*-monitor but, in general, they do not affect the overall performance that much. Together with the convergence of the *Arc-Simple*-monitor, a general recommendation for α is still difficult. Too high grid stretchings might make the grid become too sluggish for more sophisticated simulations while setting the grid stretching too low results in a grid that has no adaptivity. But given the previous examples, a good rule of thumb would be $\alpha = 2.0$ for grids that only span over one or two orders of magnitude, i.e., where one prefers a fixed uniform over a fixed exponential grid. If it is the opposite

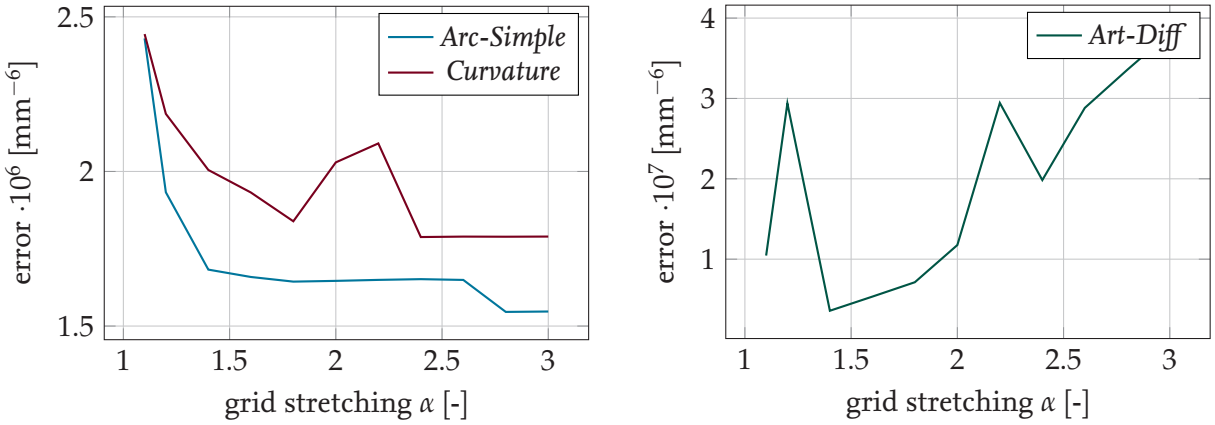


Figure 4.8: Convergence diagram of the different monitors against the FVM1000 solution over various values of the grid stretching α at 32 finite volume cells.

case, and the domain spans over a wider range, $\alpha = 1.5$ might help the grid nodes to adapt more precisely.

5 Conclusion

This work presented a comparison of different monitor functions in the context of an explicit adaptive grid scheme for solving the population balance equation. Therefore, test cases of pure growth, pure coagulation, and pure breakage have been considered. Finally, a real-world example of cell proliferation was analyzed. Tab. 5.1 shows a comprehensive overview of the performance of each monitor compared to the *FVM* reference implementation when it comes to average deviation from the analytical or from a reference solution.

Given the results of the convergence analysis as well as the capturing of the distribution and moments, the following conclusions can be drawn:

- The novel monitor that is based on *Artificial Diffusivity* shows a similar node placement as the *Curvature*-monitor. *Artificial Diffusivity* was able to capture coagulation well but was incapable at breakage. *Curvature* showed a bad performance in both test cases.
- There are cases in which a bad choice of monitor results in no convergence of the solution at all.
- The overall recommendation for the presented explicit adaptive grid scheme is the *Arc-Simple*-monitor.
- For *Arc-Simple*, the error over the grid stretching α converged already after 1.2 or 1.3 in every tested case.
- For more sophisticated simulations with more complex kinetics, the additional time it takes to calculate the remapping is mostly neglectable.
- The monitor of *Artificial Diffusivity* is also applicable to non-advection problems by using a 'virtual' growth rate. Unfortunately, its overall performance depends on a yet heuristic choice

Table 5.1: Summary of the improvements in accuracy for the different monitors over the tested cases at their respective times, all at 64 elements and varying grid stretching α. A negative sign corresponds to a decrease in deviation hence a better solution. Cases: (1) Step Advection, (2) Pure Coagulation, (3) Pure Breakage (Lin), (4) Pure Breakage (Quad), (5) Cell Proliferation

Case	*(ref) FVM*	(a) *Arclength*	(b) *Curvature*	(c) *Arc-Simple*	(d) *Arc-Curv*	(e) *Art-Diff*
(1)	1.885×10^{-2}	−99.6 %	−98.9 %*	−99.6 %	−99.4 %*	−97.5 %*
(2)	7.792×10^{-5}	+114.1 % n.c.	+82.0 % n.c.	−65.0 %	n.t.	+19.3 %
(3)	7.562×10^{-3}	+145.8 % n.c.	+51.2 % n.c.	−62.6 %	n.t.	n.t.
(4)	2.641×10^{-3}	+115.5 % n.c.	+101.4 % n.c.	−71.4 %	n.t.	n.t.
(5)	2.387×10^{6}	−61.3 %	−58.5 %	−68.7 %	n.t.	+389.3 % n.c.

n.c. = not converging, n.t. = not tested, * = problems at higher cell counts
no convergence information for the case of cell proliferation, performed against FVM(1000 cells)

of the limiter η.

- The proposed FV-scheme for covering the contribution of breakage is geometrically motivated and does a good job of representing the moments. When using with the *Arc-Simple* monitor it shows comparable precision to the mass conservation form of Qamar et al. (2009).
- If the given monitors converge, they do so at approximately the same order. By contrast, the *FVM*-reference implementation converges only slightly faster meaning that it will reach the superior accuracy of the monitors at high cell counts.

For future perspectives, we would find it interesting to compare the different monitors on a test case that involves nucleation. Since this would complete the phenomena occurring in population balance modeling a final decision on the choice of the monitor function can be made. However, it should be kept in mind that the actual performance of the monitor is closely linked to the adaptive scheme it works with. A more general comparison between this method and, e.g., the scheme of Tang and Tang (2003) can reveal further connections.

Additionally, we think that there is still potential in the monitor of *Artificial Diffusivity* that we were not able to find yet. Maybe the analytical behaviour can be further analyzed and the connection between it and the monitor of *Curvature* can be resolved.

Bibliography

Blom, J. G. and J. G. Verwer (1989), "On the use of the arclength and curvature monitor in a moving-grid method which is based on the method of lines", *Adaptive Methods for Partial Differential Equations*, pp. 160–175.

Ceniceros, H. D. and T. Y. Hou (2001), "An efficient dynamically adaptive mesh for potentially singular solutions", *Journal of Computational Physics* 172.2, 609–639.

Dorfi, E. and L. Drury (1987), "Simple adaptive grids for 1-D initial value problems", *Journal of Computational Physics* 69.1, 175–195.

Filbet, F. and P. Laurençot (2004), "Numerical simulation of the Smoluchowski coagulation equation", *SIAM Journal on Scientific Computing* 25.6, 2004–2028.

Fredrickson, A. G. (2003), "Population balance equations for cell and microbial cultures revisited", *AIChE journal* 49.4, 1050–1059.

Fredrickson, A. G., D. Ramkrishna, and H. M. Tsuchiya (1967), "Statistics and dynamics of procaryotic cell populations", *Mathematical Biosciences* 1.3, 327–374.

Godunov, S. K. (1959), "A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics", *Matematicheskii Sbornik* 89.3, 271–306.

Hatzis, C., S. Friedrich, and A. Fredrickson (1995), "Multistaged corpuscular models of microbial growth: Monte Carlo simulations", *Biosystems* 36.1, 19–35.

Hulburt, H. and S. Katz (1964), "Some problems in particle technology: A statistical mechanical formulation", *Chemical Engineering Science* 19.8, 555–574.

Jones, A. G. (2002), *LES-BOFFIN: User's Guide*.

Koren, B. (1993), "A robust upwind discretization method for advection, diffusion and source terms", *Notes on Numerical Fluid Mechanics* 45.

Kumar, J., G. Warnecke, M. Peglow, and S. Heinrich (2009), "Comparison of numerical methods for solving population balance equations incorporating aggregation and breakage", *Powder Technology* 189.2, 218–229.

Kumar, S. and D. Ramkrishna (1996a), "On the solution of population balance equations by discretization–I. A fixed pivot technique", *Chemical Engineering Science* 51.8, 1311–1332.

— (1996b), "On the solution of population balance equations by discretization–II. A moving pivot technique", *Chemical Engineering Science* 51.8, 1333–1342.

— (1997), "On the solution of population balance equations by discretization–III. Nucleation, growth and aggregation of particles", *Chemical Engineering Science* 52.24, 4659–4679.

Lim, Y., J. Le Lann, X. Meyer, and X. Joulia (2001), "Dynamic simulation of batch crystallization process by using moving finite difference method", *European Symposium on Computer Aided Process Engineering - 11*, ed. by R. Gani and S. B. Jrgensen, vol. 9, Elsevier, 201–206.

Liu, A. and S. Rigopoulos (2018), "A sectional seeking discretization technique for the coagulation process in the population balance equation", *Unpublished*.

Mancuso, L., M. Liuzzo, S. Fadda, et al. (2009), "Experimental analysis and modelling of in vitro proliferation of mesenchymal stem cells", *Cell Proliferation* 42.5, 602–616.

Mantzaris, N. V., J.-J. Liou, P. Daoutidis, and F. Srienc (1999), "Numerical solution of a mass structured cell population balance model in an environment of changing substrate concentration", *Journal of Biotechnology* 71.1, 157–174.

Miller, K. and R. Miller (1981), "Moving Finite Elements. I", *Journal on Numerical Analysis* 18.6, 1019–1032.

Omar, H. M. and S. Rohani (2017), "Crystal Population Balance Formulation and Solution Methods: A Review", *Crystal Growth & Design* 17.7, 4028–4041.

Qamar, S., M. Elsner, I. Angelov, G. Warnecke, and A. Seidel-Morgenstern (2006), "A comparative study of high resolution schemes for solving population balances in crystallization", *Computers & Chemical Engineering* 30.6, 1119–1131.

Qamar, S., A. Ashfaq, G. Warnecke, et al. (2007), "Adaptive high-resolution schemes for multidimensional population balances in crystallization processes", *Computers & Chemical Engineering* 31.10, 1296–1311.

Qamar, S., G. Warnecke, and M. P. Elsner (2009), "On the solution of population balances for nucleation, growth, aggregation and breakage processes", *Chemical Engineering Science* 64.9, 2088–2095.

Ramkrishna, D. (2000), *Population balances: Theory and applications to particulate systems in engineering*, Academic press.

Ramkrishna, D. and M. R. Singh (2014), "Population Balance Modeling: Current Status and Future Prospects", *Annu. Rev. Chem. Biomol. Eng.* 5.1, 123–146.

Rigopoulos, S (2010), "Population balance modelling of polydispersed particles in reactive flows", *Progress in Energy and Combustion Science* 36.4, 412–443.

Rigopoulos, S. and A. G. Jones (2003), "Finite-element scheme for solution of the dynamic population balance equation", *AIChE Journal* 49.5, 1127–1139.

Scott William T. (1968), "Analytic Studies of Cloud Droplet Coalescence I", *Journal of the Atmospheric Sciences* 25.1, doi: 10.1175/1520-0469(1968)025<0054:ASOCDC>2.0.CO;2, 54–65.

Sewerin, F. and S. Rigopoulos (2017), "An explicit adaptive grid approach for the numerical solution of the population balance equation", *Chemical Engineering Science* 168, 250–270.

— (2018), "An LES-PBE-PDF approach for predicting the soot particle size distribution in turbulent flames", *Combustion and Flame* 189, 62–76.

Stockie, J., J. Mackenzie, and R. Russell (2001), "A Moving Mesh Method for One-dimensional Hyperbolic Conservation Laws", *Journal of Scientific Computing* 22.5, 1791–1813.

Sweby, P. (1984), "High Resolution Schemes Using Flux Limiters for Hyperbolic Conservation Laws", *SIAM Journal on Numerical Analysis* 21.5, 995–1011.

Tang, H. and T. Tang (2003), "Adaptive Mesh Methods for One- and Two-Dimensional Hyperbolic Conservation Laws", *Journal of Numerical Analysis* 41.2, 487–515.

Ziff, R. M. and E. D. McGrady (1985), "The kinetics of cluster fragmentation and depolymerisation", *Journal of Physics A: Mathematical and General* 18.15, p. 3027.