ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

PAUL SCHERRER INSTITUT
PSI

The Langevin Approach to Discretize the Collision Operator

Master's Thesis

Department of Physics

ETH Zürich

written by

Tobia Claglüna

supervised by

Dr. A. Adelmann

scientific advisers

Sonali Mayani
Dr. S. Muralikrishnan
Dr. M. Frey
Dr. A. Cerfon

July 17, 2023

Abstract

We present a numerical solver for the Vlasov-Poisson equation to simulate plasma dynamics dominated by Coulomb collisions. The method makes use of a Langevin type operator to approximate interparticle collisions via Rosenbluth potentials. We employ a Particle-in-Cell approach to efficiently simulate the motion of electrons in a self-consistent manner. The governing stochastic differential equation introduces a velocity dependent dynamic friction and diffusion term accounting for the numerous small angle collisions due to Coulomb repulsion in form of a stochastic process.

We carry out a thorough study on suitable differential operators for computing the Rosenbluth potentials as well as the collisional coefficients. The implementation is subsequently tested on a disordered induced heating process in which particle collisions play an essential role for the system to attain an equilibrium state. Additionally, we demonstrate the correctness of the computed collisional terms on analytical test cases.

Contents

List of Figures

Acronyms

DIH	Disordered-Induced-Heating
FD	Finite Difference
FEL	Free-Electron-Laser
FFT	Fast Fourier Transform
FP	Fokker-Planck
IPPL	Independent Parallel Particle Layer
LDLT	Factorization of a Matrix $\underline{\underline{A}} = \underline{\underline{LDL}}^*$
MPI	Message Passing Interface
OPAL	Object Oriented Particle Accelerator Library
OpenMP	Open Multi-Processing API
P^{3}M	Particle-Particle-Particle-Mesh
PDE	Partial Differential Equation
PIC	Particle-in-Cell
PM	Particle-Mesh
PP	Particle-Particle
RMS	Root-Mean-Square
SDE	Stochastic Differential Equation
SIMD	Single Instruction Multiple Data
VPFP	Vlasov-Poisson-Fokker-Planck

Chapter 1

Introduction

Plasma, often referred to as the fourth state of matter, consists of a gas-like collection of charged particles (ions and electrons) that exhibit a collective behavior. Understanding its behavior under the influence of electric and magnetic fields is of critical importance for numerous areas in research as well as industry. A prime example are plasmas confined in particle accelerators where particle bunches are exposed to a range of interesting phenomena which depend on the present plasma state.

In this report, we aim to model a phenomenon arising in ultra-cold plasmas where the transport of the contained particles is strongly influenced by interparticle collisions. Modeling these scattering processes has been an active area of research for many decades. The individual scattering events are very weak and often happen over large distances. Thus, modeling them as binary collisions requires very small timesteps which is not feasible for the number of particles one usually encounters in dense plasmas.

Instead, researchers have explored methods emanating from plasma kinetic theory which model the scattering processes via the Fokker-Planck (FP) equation [3, 4]. The definition of the friction and diffusion term of the FP formalism is not straightforward, thus researches usually have to state assumptions on the system state in order to approximate them [5, 6, 7]. In our work we model the collisions via the Langevin formulation of the FP equation, allowing us to model them as a stochastic processes, providing a velocity dependent deterministic friction and stochastic diffusion term [8]. This formulation facilitates the use of the Fokker-Planck term as part of the well-established Particle-in-Cell (PIC) method [9].

1.1 Motivation

Free-Electron-Lasers (FEL) have the ability to create beams that emit very short coherent light pulses which exhibit wave lengths down to 0.1nm. This can be used to map the atomic structure of proteins with a high temporal resolution. An electron beam is emitted from an electron gun which is subsequently accelerated through an array of magnets causing a lumping of the beam into bunches. This process is prone to be negatively impacted by intra-beam scattering, causing the beam to widen over time. As this phenomenon negatively affects the achievable beam brightness, it is important to calibrate an accelerator to counter it as much as possible.

SwissFEL is an X-ray FEL at the Paul Scherrer Institute containing multiple beamlines for producing different types of X-ray pulses. Since its commissioning in 2016, the need arose for a method which is able to accurately model the inter-particle collisions in an efficient manner. The existing method [2] based on the P^3M algorithm (Particle-Particle-Particle-Mesh, see Section 3.2)

has been shown to model collisions appropriately, but it does so at a considerable computational cost. This limits its applicability to small simulation domains. The aforementioned method of modeling collisions stochastically via the Langevin formulation, promises better computational complexity and therefore allowing to simulate larger number of particles and simulation domains. We test the method on an experiment which is governed by disordered induced heating and has been shown to be resolved correctly only if individual particle collisions are considered (via the P^3M method). This provides us with a challenging test case for the implemented method.

1.2 Outline

We start by introducing the notation used throughout the report, followed by a succinct treatise of the underlying theory of plasma modeling in the context of kinetic theory, resulting in the Langevin formulation of the FP equation. Subsequently we discuss the numerical methods used to solve the previously introduced equations and propose an analytical test case that is used to verify the correctness of our implementation in the following chapter. We then apply it to the physical test case governed by disorder induced heating and explore what values the two collisional coefficients attain and how they impact the normalized emittance. We conclude this work by highlighting the key findings of our investigation. Additionally, we propose algorithmic improvements and suggest possible directions for further exploration of the collisional terms applied in the context of disordered induced heating.

Chapter 2

Background

In this chapter we introduce the necessary concepts to describe the dynamics of a fully ionized single species plasma under the influence of many simultaneous particle collisions. We conclude by presenting the disorder induced heating (DIH) process and highlight why it is necessary to model collisions in such a regime.

2.1 Notation

In this section we introduce the notation used throughout the report (it partly follows the one introduced in [10]). We denote vector quantities with bold font $\boldsymbol{b}$ and tensor quantities with a double underscore $\underline{\underline{C}}$. Individual components of vector or tensor quantities are displayed with subscripts, i.e. $\boldsymbol{b}_i$, $\underline{\underline{C}}_{i,j}$. Given a normed vector space in $\mathbb{R}^3$, we can write a rank 2 tensor with orthonormal unit vectors $\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3$, as $\underline{\underline{C}} = \sum_{i,j} \underline{\underline{C}}_{i,j} \boldsymbol{e}_i \boldsymbol{e}_j$. A dot between two vectors defines a contraction of the nearest indices $(\nabla \boldsymbol{b}) \cdot \boldsymbol{d} = \sum_j (\nabla_i \boldsymbol{b}_j) \boldsymbol{d}_j$. Analogously, a double dot product between tensors is the contraction $\underline{\underline{C}} : \underline{\underline{E}} = \sum_{i,j} C_{i,j} E_{i,j}$. We define the L^2-norm of vectors as $\|\boldsymbol{b}\|_2 = \sqrt{\sum_i |b_i|^2}$. To simplify formulas containing L^2-norms of the phase space position and velocity vectors $(\boldsymbol{r}, \boldsymbol{v})$, we choose to simplify the notation and use non-bold letters (r, v) instead.

To represent partial differential equations (PDEs) of vectorial quantities we rely on differential operators. We shortly list the operators found in this report. The operator subscript indicates on what subspace variables it acts on (i.e. $\nabla_{\boldsymbol{r}}$ acts in configuration space whereas $\nabla_{\boldsymbol{v}}$ acts in velocity space)

$$\nabla_{\boldsymbol{r}} = \left(\frac{\partial}{\partial \boldsymbol{r}_1}, \frac{\partial}{\partial \boldsymbol{r}_2}, \frac{\partial}{\partial \boldsymbol{r}_3} \right)^T, \tag{2.1}$$

$$\Delta_{\boldsymbol{r}} = \nabla_{\boldsymbol{r}} \cdot \nabla_{\boldsymbol{r}}, \tag{2.2}$$

$$\underline{\underline{H}}_{\boldsymbol{r}} = \begin{bmatrix} \frac{\partial^2}{\partial \boldsymbol{r}_1 \partial \boldsymbol{r}_1} & \frac{\partial^2}{\partial \boldsymbol{r}_1 \boldsymbol{r}_2} & \frac{\partial^2}{\partial \boldsymbol{r}_1 \boldsymbol{r}_3} \\ \frac{\partial^2}{\partial \boldsymbol{r}_2 \partial \boldsymbol{r}_1} & \frac{\partial^2}{\partial \boldsymbol{r}_2 \boldsymbol{r}_2} & \frac{\partial^2}{\partial \boldsymbol{r}_2 \boldsymbol{r}_3} \\ \frac{\partial^2}{\partial \boldsymbol{r}_3 \partial \boldsymbol{r}_1} & \frac{\partial^2}{\partial \boldsymbol{r}_3 \boldsymbol{r}_2} & \frac{\partial^2}{\partial \boldsymbol{r}_3 \boldsymbol{r}_3} \end{bmatrix}, \tag{2.3}$$

$$\mathrm{Tr}(\underline{\underline{C}}) = \sum_i \underline{\underline{C}}_{i,i}. \tag{2.4}$$

Part of the methods used in this report require a computational mesh as a datastructure to store field quantities. We define the extents of a field in 3 dimensions as $[0, L_x] \times [0, L_y] \times [0, L_z]$, while the mesh which lies on top of this domain contains a total of $N_x \times N_y \times N_z$ cells. Thus, the mesh width or cellwidth along each dimension is $h_x = L_x/N_x, h_y = L_y/N_y, h_z = L_z/N_z$. We use a cell centered approach to define the quantities on a mesh: $\boldsymbol{x}_{i,j,k} = (ih_x, jh_y, kh_z)$ with half indices $(i, j, k) \in [\frac{1}{2}, \ldots, N_x - \frac{1}{2}] \times [\frac{1}{2}, \ldots, N_y - \frac{1}{2}] \times [\frac{1}{2}, \ldots, N_y - \frac{1}{2}]$. If a field exhibits the same extent along each dimension, we use no subscript the introduced mesh properties ($[0, L]^3$ and h).

For mesh quantities in the context of a finite difference (FD) scheme, the superscript $\boldsymbol{x}^n = \boldsymbol{x}(t_n)$ defines the state of it at timestep n. The same quantity at the subsequent timestep is then $\boldsymbol{x}^{n+1} = \boldsymbol{x}(t_n + dt)$. Similarly, for neighboring mesh cells of $\boldsymbol{x}_{i,j,k}$ we use an index increment ($\boldsymbol{x}_{i+1,j,k}$ for its right neighboring cell along the first dimension).

In Tables 2.1 and 2.2 we summarize the nomenclature we use in this report.

Table 2.1: Nomenclature used throughout this report.

Symbol	Definition
N_o	Mesh size along one dimension, where $o \in \{x, y, z\}$
N_m	Total number of cells in a mesh ($N_m = N_x \times N_y \times N_z$)
N_p	Total number of particles
η	Relative approximation error $\eta(x, x_{\text{appr}}) = \left(\frac{\|x_{\text{appr}} - x\|_2}{\|x\|_2}\right)$

Table 2.2: Nomenclature commonly used in plasma physics.

Symbol	Definition
n_0	Electron number density
λ_D	Debye length
v_{th}	Thermal velocity
ω_p	Plasma frequency
τ_p	Plasma period
ε_0	Vacuum permittivity
m_e	Electron mass
e	Elementary charge

2.2 Coulomb Interactions

We need to specify the type of collisions under consideration. The plasma containing charged particles (in our case electrons) can exhibit very high density (up to 10^{30}m^{-3}). Due to Coulomb's inverse-square law, the particles create an electric field which makes collective interactions possible. With collective we mean, that every particle experiences electric field forces from neighboring particles while at the same time, they themselves are influenced by particles nearby.

2.3 Vlasov-Poisson Equation

The majority of phenomena arising from particles interacting in a plasma state can be modeled adequately with a description of fluids (i.e. with Navier-Stokes equation), allowing the analysis

of macroscopic properties such as density and mean energy. The weakness of this method is its inability to model small-scale structures which can become instrumental to the system's behavior for states away from the thermal equilibrium. In these regimes, the velocity distribution of the system plays an important role in how particles interact. Thus, using a description giving us information about both position and velocity in a fluid element is paramount. This can be achieved by expressing a 6d phase space $(\boldsymbol{r}, \boldsymbol{v})$ with the help of a distribution function $f(\boldsymbol{r}, \boldsymbol{v}, t)$, which characterizes the particle density at position and velocity $(\boldsymbol{r}, \boldsymbol{v})$ at time t.

Taking a particle description of a plasma containing N_p particles, we can define its distribution with the Dirac delta function:

$$f_K(\boldsymbol{r}, \boldsymbol{v}, t) = \sum_{i=1}^{N_p} \delta[\boldsymbol{r} - \boldsymbol{r}_i]\delta[\boldsymbol{v} - \boldsymbol{v}_i]. \tag{2.5}$$

This description is also known as the Klimontovich distribution function and grants us the insight of whether we can expect to find a particle at a certain position $(\boldsymbol{r}, \boldsymbol{v})$ at time t [11]. But it being zero almost everywhere except at the particle positions $(\boldsymbol{r}, \boldsymbol{v})$ can be inhibiting for subsequent computations due to its singular nature. A better description still, would exhibit at least some smoothness as we are more interested in averaged quantities. By integrating over a volume element $dV = \Delta\boldsymbol{r}\Delta\boldsymbol{v}$, containing $N_{dV}(\boldsymbol{r}, \boldsymbol{v}, t) \gg 1$ particles, we arrive at a description of the number density at this position in phase space:

$$f(\boldsymbol{r}, \boldsymbol{v}, t) = \frac{1}{\Delta\boldsymbol{r}\Delta\boldsymbol{v}} \int_{\Delta\boldsymbol{r}} d\boldsymbol{r} \int_{\Delta\boldsymbol{v}} d\boldsymbol{v} f_K = \frac{N_{dV}(\boldsymbol{r}, \boldsymbol{v}, t)}{\Delta\boldsymbol{r}\Delta\boldsymbol{v}}. \tag{2.6}$$

By simple integration over the components of the phase space density, we can define the configuration and velocity space density respectively as follows

$$n_0(\boldsymbol{r}, t) = f(\boldsymbol{r}, t) = \int d\boldsymbol{v} f(\boldsymbol{r}, \boldsymbol{v}, t), \qquad f(\boldsymbol{v}, t) = \int d\boldsymbol{r} f(\boldsymbol{r}, \boldsymbol{v}, t). \tag{2.7}$$

We define the continuity equation for f as

$$\frac{\partial f}{\partial t} + \boldsymbol{v} \cdot \frac{\partial f}{\partial \boldsymbol{r}} + \frac{\boldsymbol{F}}{m} \frac{\partial f}{\partial \boldsymbol{v}} = 0, \tag{2.8}$$

where we replaced the particle acceleration in a volume element dV with $\boldsymbol{F}/m$ and $\boldsymbol{F} = \boldsymbol{F}_{mf} + \boldsymbol{F}_{ext}$ representing all collisionless (macroscopic) forces acting on the particles (self-consistent field and external forces). Equation 2.8 is known as the collisionless kinetic equation.

By denoting $(\partial f/\partial t)_{\text{coll}}$ as the rate of change of f over time due to collisions and combining it with the continuity equation, we arrive at collisional kinetic equation or also known as the *Vlasov* equation [12]:

$$\frac{\partial f}{\partial t} + \boldsymbol{v} \cdot \frac{\partial f}{\partial \boldsymbol{r}} + \frac{\boldsymbol{F}}{m} \frac{\partial f}{\partial \boldsymbol{v}} = \left(\frac{\partial f}{\partial t}\right)_{\text{coll}}. \tag{2.9}$$

From this point onward we choose to directly simplify the introduced equations with assumptions made for the disorder induced heating process (see Section 2.7) to avoid listing them multiple times with only small adjustments. Equation 2.9 alone, also known as the Vlasov equation, does not give us the full picture yet, as the force term $\boldsymbol{F}$ contains both external and self-consistent electromagnetic fields emerging from the plasma motion. In order to compute the self-consistent fields, we need the Maxwell equations. We limit ourselves to the electrostatic case, meaning we

can omit time dependent terms as well as the magnetic self-fields. These assumptions leaves us with only the Poisson equation for defining the contributions to the force term $\boldsymbol{F}$:

$$\nabla^2_{\boldsymbol{r}}\phi(\boldsymbol{r}) = -\frac{\rho(\boldsymbol{r})}{\epsilon_0}. \tag{2.10}$$

Where the charge density of particles with charge q is given by $\rho(\boldsymbol{r}) = qf(\boldsymbol{r})$, and ε_0 being the vacuum permittivity. The coupled Equations 2.9 and 2.10 are also known as the Vlasov-Poisson equations.

So far we have not yet discussed the right-hand side term of the Vlasov Equation 2.9. Its approximation is an ongoing problem for which several approaches have been proposed ([5], [6], [4]). In the next section we discuss one possible approach of deriving such a collision operator.

2.4 Fokker-Planck Collision Operator

The collision term $(\partial f/\partial t)_{\text{coll}}$ should capture changes to the density function $f(\boldsymbol{r}, \boldsymbol{v})$ due to interactions between particles (e.g. collisions). In our case collisions in a fully ionized plasma, this reduces to Coulomb interactions (see Section 2.2). In fact, these collisions are responsible for the relaxation of the phase space towards equilibrium [13]. We can think of a test particle traveling through a background of equally charged particles ("scatterers"). On its path it gets influenced by the electric force of these nearby particles. In the context of a plasma beam, this is also called intrabeam scattering. However, the interactions are limited to a Debye radius λ_D, "shielding" it off from particles farther away. A binary collision model, such as the Boltzmann collision integral [14], seems intuitive at first. We illustrate why this is not sufficient when considering that the number of particles in a Debye sphere can grow very large.

These collisions exhibit two important properties: first, an individual collision (i.e. its potential energy) is weak compared to the thermal energy of the system [12], causing only small angle deflections. It can be shown that the accumulated effect of these far outweigh more rare strong interactions. Second, the considered time-scale is usually much larger than the collision time τ_c but still smaller than the dissipation time ν [15]:

$$\tau_c \ll \Delta t \ll \nu, \tag{2.11}$$

where the dissipation time ν is of the scale after which a small perturbation to the phase space density is observed to be below a fixed threshold [16].

Following the above argument, it is a reasonable assumption that the spatial position of the test particle is not substantially affected by the Coulomb collisions, thus we can speak of the collisional effects to be local in configuration space $(\boldsymbol{r}, t)$ [17], meaning that it is sufficient to only consider changes to the distribution function in velocity space.

These mentioned properties of collisions show a stark similarity to properties known from Brownian motion. Indeed, there exists the *Fokker-Planck* formulation of the collision operator which can be derived from the theory random particle motion or in our case random collisions which are akin to Markov processes (see Section 2.6). Meaning that they underlie a stochastic process which is independent of its past states.

In the Fokker-Planck approach we advance our phase space density in time by Δt via an integral over a probability function $\psi(\boldsymbol{v}, \Delta\boldsymbol{v})$ that defines how likely it is that a particle with velocity $\boldsymbol{v}$ experiences a change in velocity $\Delta\boldsymbol{v}$. Taking the truncated Taylor expansion up to order 2 of this integral results in the Fokker-Planck operator

$$\left(\frac{\partial f}{\partial t}\right)_{\text{coll}} = -\frac{\partial}{\partial \boldsymbol{v}} \cdot \left(\frac{f\langle \Delta \boldsymbol{v}\rangle}{\Delta t}\right) + \frac{1}{2}\frac{\partial^2}{\partial \boldsymbol{v}\partial \boldsymbol{v}} : \left(\frac{f\langle \Delta \boldsymbol{v}\Delta \boldsymbol{v}\rangle}{\Delta t}\right), \tag{2.12}$$

$$\begin{Bmatrix} \langle \Delta \boldsymbol{v}\rangle \\ \langle \Delta \boldsymbol{v}\Delta \boldsymbol{v}\rangle \end{Bmatrix} = \int \psi(\boldsymbol{v}, \Delta \boldsymbol{v}) \begin{Bmatrix} \Delta \boldsymbol{v} \\ \Delta \boldsymbol{v}\Delta \boldsymbol{v} \end{Bmatrix} d(\Delta \boldsymbol{v}),$$

where $\langle \Delta \boldsymbol{v}\rangle/\Delta t$ defines the change in velocity averaged over an ensemble moving with speed $\boldsymbol{v}$, and similarly $\langle \Delta \boldsymbol{v}\Delta \boldsymbol{v}\rangle/\Delta t$ is the average change due to diffusion of the velocities in an ensemble. This definition is still detached from the notion of describing the physics of collisions themselves. We shall take the definition of Rosenbluth et al. [4] to obtain an approximation to these terms. The authors make use of the inverse square law of Coulomb interactions and carry out their calculations in a center-of-mass frame of reference where $\boldsymbol{g}$ is the relative velocity with respect to its frame. In Figure 2.1 we can see an illustration of this frame of reference, where an incoming scatterer with relative velocity $\boldsymbol{g}$ is deflected by an angle θ.

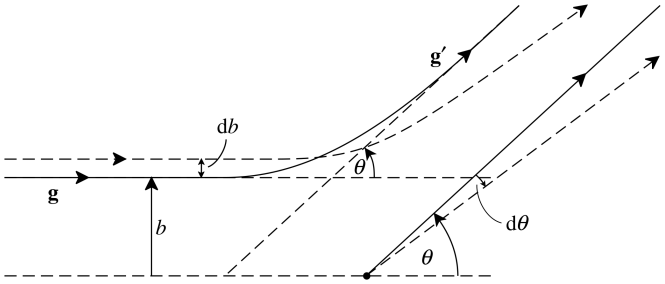


Figure 2.1: Scattering in the center of mass frame [1].

The scattering angle is directly related to the impact parameter b of the scatterer. As aforementioned, the majority of collisions within the range of a Debye sphere are happening at large impact parameters (i.e. small angles θ). With the Rutherford scattering cross-section, $\sigma(|\boldsymbol{g}|, \theta) = -\frac{b}{\sin\theta}\frac{db}{d\theta}$, one can then integrate over all possible scatterers of a certain velocity $\boldsymbol{v}_s$ and arrive at the so called Rosenbluth potentials (simplified for single species plasma):

$$h(\boldsymbol{v}) = 2\int \frac{f(\boldsymbol{v})}{\|\boldsymbol{v} - \boldsymbol{v}_s\|} d\boldsymbol{v}_s, \tag{2.13}$$

$$g(\boldsymbol{v}) = \int \|\boldsymbol{v} - \boldsymbol{v}_s\| f(\boldsymbol{v}) d\boldsymbol{v}_s. \tag{2.14}$$

We can now succinctly write the two ensemble averages defined in Equation 2.12 as

$$\boldsymbol{F}_d(\boldsymbol{v}) = \frac{\langle \Delta \boldsymbol{v}\rangle}{\Delta t} = \Gamma \frac{\partial h(\boldsymbol{v})}{\partial \boldsymbol{v}}, \tag{2.15}$$

$$\underline{\underline{D}}(\boldsymbol{v}) = \frac{\langle \Delta \boldsymbol{v}\Delta \boldsymbol{v}\rangle}{\Delta t} = \Gamma \frac{\partial^2 g(\boldsymbol{v})}{\partial \boldsymbol{v}\partial \boldsymbol{v}}, \tag{2.16}$$

with the common prefactor $\Gamma = \frac{q_e^4}{4\pi\epsilon_0^2 m_e^2} \ln\Lambda$ being a function of the Coulomb logarithm $\ln\Lambda \approx 10$. The term in Equation 2.15 is known as the *dynamic friction* coefficient and Equation 2.16 as the *dynamic diffusion* coefficient.

2.5 Properties of Rosenbluth Potentials

It may not be obvious at first how to compute the integrals in Equations 2.13 and 2.14. But we can find a stark similarity to potential theory (see [18]) which allows us to reformulate them as

$$\nabla_{\boldsymbol{v}}^2 h(\boldsymbol{v}) = -8\pi f(\boldsymbol{r}, \boldsymbol{v}), \tag{2.17}$$

$$\nabla_{\boldsymbol{v}}^2 \nabla_{\boldsymbol{v}}^2 g(\boldsymbol{v}) = -8\pi f(\boldsymbol{r}, \boldsymbol{v}). \tag{2.18}$$

As a result we can then directly employ well-established methods (as introduced in Chapter 3) for their computation. Observing the matching right-hand side terms, it is easy to show that from Equations 2.15 - 2.18 we can establish following relations (cf. [19])

$$\begin{aligned} \mathrm{Tr}(\underline{\underline{D}}(\boldsymbol{v})) &= \Gamma \nabla_{\boldsymbol{v}}^2 g(\boldsymbol{v}) \\ &= \Gamma h(\boldsymbol{v}), \\ \nabla_{\boldsymbol{v}} \cdot \underline{\underline{D}}(\boldsymbol{v}) &= \nabla_{\boldsymbol{v}} \cdot (\Gamma \nabla_{\boldsymbol{v}} \nabla_{\boldsymbol{v}} g(\boldsymbol{v})) \\ &= \Gamma \nabla_{\boldsymbol{v}} \nabla_{\boldsymbol{v}}^2 g(\boldsymbol{v}) \\ &= \boldsymbol{F}_d(\boldsymbol{v}) \\ &= \Gamma \nabla_{\boldsymbol{v}} h(\boldsymbol{v}). \end{aligned} \tag{2.19}$$

These yield an additional apparatus to further test whether an implementation is correct. We mention this since one has to implement two different solvers to compute either potential and also, as we will see, there exist multiple ways to numerically compute the Fokker-Planck coefficients from the potentials.

The coefficients also possess some interesting properties as described in Hinton [20] that will help us derive the numerical model explained in Chapter 3. The proofs of which are detailed therein, while we content ourselves by only summarizing the key points:

The distribution function cannot become negative ($f(\boldsymbol{r}, \boldsymbol{v}, t) \geq 0$) and as a consequence also the tensor $\partial^2 g(\boldsymbol{v})/\partial \boldsymbol{v} \partial \boldsymbol{v}$ is positive semi-definite. Therefore intuitively, one can think of the modeled Coulomb collisions "filling" up any existing areas close to zero in the phase space $f(\boldsymbol{r}, \boldsymbol{v}, t)$.

2.6 Langevin Equation

In the last section we touched upon the fact that the collision operator operates on a time-scale where macroscopic properties of the phase space experience slow changes as a result of many random short collisions. These dynamics can also be characterized by the Langevin equation giving us a different view on the problem, while also allowing us to integrate it with the PIC approach (see Section 3.2). Thus, it does not come as a surprise that we can show equivalence between the Fokker-Planck and Langevin description [21]. The Langevin equation describing the dynamics of $\boldsymbol{v}$ as a Markov process, takes on the following form [6]:

$$\frac{d\boldsymbol{v}(t)}{dt} = \boldsymbol{F}(\boldsymbol{v}) + \underline{\underline{Q}}(\boldsymbol{v})d\boldsymbol{W}(t), \tag{2.20}$$

where $\underline{\underline{Q}}$ follows from the factorization of $\underline{\underline{D}} = \underline{\underline{Q}}^T \underline{\underline{Q}}$ and $d\boldsymbol{W}(t)$ is the randomly fluctuating Langevin force with zero mean and correlation function:

$$\begin{cases} \langle d\boldsymbol{W}_i(t)\rangle = 0, \\ \langle d\boldsymbol{W}_i(t) d\boldsymbol{W}_j(t')\rangle = \delta_{ij}\delta(t-t'). \end{cases} \tag{2.21}$$

As we will see in Section 3.3.1 we can use this to define the time evolution of the coupled Vlasov-Poisson-Fokker-Planck (VPFP) equation.

2.7 Disorder Induced Heating

Ultracold plasma is governed by the very low kinetic energy of initially unstructured (disordered) particles. Due to interparticle forces (i.e. Coulomb repulsion) they then start to relax from a unstructured into a more structured state with lower potential energy. Hence causing an increase in kinetic energy in form of a heating process with increased particle velocities. For simulating a plasma undergoing this process, the particle interactions can therefore not be regarded as a small perturbation of the system dynamics anymore.

The DIH effect is known to limit the beam brightness of cold electron beams that emanate from a photoemission source [22]. In the case of the Free Electron Laser SwissFEL it has been observed by Prat et al. [23] that their current simulation method cannot completely replicate the observed behavior in the beamline. One suspicion that has come up as an attempted explanation, is that the current method does not capture the interparticle collisions well. Therefore we strive to implement a solver which enables to model these interactions in a more efficient manner than the existing P^3M method by Ulmer [2].

Beam Emittance

When simulating a particle bunch in an accelerator one needs a way to describe how it behaves over time. Beam emittance is a conserved quantity of motion that is based on the coordinates of the particles in phase space. Intuitively one can think of the emittance value as an area (by projection of the phase space) that exhibits an ellipsoidal shape.

We can define the RMS emittance and normalized emittance along the first dimension by the central moments $\langle\cdot\rangle$ of the particle distributions as follows:

$$\varepsilon_x = \sqrt{\langle \boldsymbol{r}_x^2\rangle\langle \boldsymbol{v}_x^2\rangle - \langle \boldsymbol{r}_x \boldsymbol{v}_x\rangle^2}, \tag{2.22}$$

$$\varepsilon_{x,n} = \sqrt{\langle \boldsymbol{r}_x^2\rangle\langle (\gamma\beta_x)^2\rangle - \langle \boldsymbol{r}_x \gamma\beta_x\rangle^2} = \varepsilon_x \gamma\beta_x, \tag{2.23}$$

where $\gamma = \left(1 - \left(\frac{v}{c}\right)^2\right)^{-\frac{1}{2}}$ is the Lorentz factor and $\beta_x = \frac{\boldsymbol{v}_x}{c}$ is the normalized transverse velocity. As a bunch accelerates, the transverse size shrinks. Though if we scale the RMS emittance by $(\gamma\beta_x)$, we can recover the invariant property of the emittance [24], hence the formulation of normalized emittance $\varepsilon_{x,n}$.

We will use the normalized emittance in order to validate the correctness of our simulation in the DIH case (see Chapter 4).

Plasma Frequency

Cold plasma exhibits an oscillating behavior at a certain plasma frequency ω_p. It is a function of the number density n_0, the elementary charge e, the electron mass m_e and the vacuum permittivity ε_0:

$$\omega_p = \sqrt{\frac{n_0 e^2}{m_e \varepsilon_0}}. \tag{2.24}$$

Following this definition we can define the plasma period as $\tau_p = 2\pi/\omega_p$.

Chapter 3

Computational Model

In the following chapter we introduce the methods used for the numerical treatment of the governing equations stated in the previous chapter. We show that we can reformulate them in a succinct scheme which is able to resolve the dynamics of our system and allow us to advance it in time. At the end of this chapter we introduce an analytical test case that is used in Chapter 4 to verify that the terms emanating from the right-hand side term of Eq. 2.12 are correctly computed and show the expected convergence rate of our scheme.

3.1 Field Solvers

In this section we present the numerical methods to solve the elliptic PDEs introduced in the previous chapter. We introduce two solver types that are based on the Fast Fourier transform (FFT) and highlight their strengths and weaknesses. Both methods aim to solve the electrostatic Poisson's equation (2.10), or equations that show structural similarity with it. Albeit their approach differs slightly, both methods achieve to impose open boundary (free-space) conditions ($\lim_{\|\boldsymbol{r}\|_2 \to \infty} \rho(\boldsymbol{r}) = 0$). Instead of using the above formulation of the PDE we opt to formulate it as a convolutional integral using Green's function:

$$\phi(\boldsymbol{r}) = \int G(\boldsymbol{r} - \boldsymbol{r}')\rho(\boldsymbol{r}')d\boldsymbol{r}', \tag{3.1}$$

where $G(\boldsymbol{r}) = -\frac{1}{4\pi r}$ is the analytical Green's function for the Poisson problem in 3 dimensions given open boundary conditions. Similarly, $G(\boldsymbol{r}) = \frac{r}{8\pi}$ is its formulation for the biharmonic problem stated in Eq. 2.16. For their truncated spectral representations we direct the reader to Table 1 in [25].

The fact that Eq. 3.1 is a convolution, gives us the ability to employ the aforementioned Fast Fourier Transform. Another advantage of this integral form is that we can readily compute the $\boldsymbol{E}(\boldsymbol{r})$ field by employing differentiation in Fourier space, maintaining the same accuracy as the method itself without having to rely on an accurate finite difference stencil.

Hockney-Eastwood

The algorithm introduced by Hockney and Eastwood [26] (henceforth called Hockney's method) extends the charge distribution $\rho(\boldsymbol{r})$ by doubling the mesh along each spatial dimension. In this newly added part of the mesh we set the values to zero. Albeit this allows us to use efficient cyclic convolution, it comes with a higher memory footprint not only for the increased size of the

charge distribution ($(2N)^d$), but similarly as the Green's function domain has to be extended as well.

Due to the simple nature of this trick, it is easy to implement and also features good computational complexity: $\mathcal{O}((2N)^d \log(2N)^d)$ and an error convergence of up to $\mathcal{O}(N^{-2})$ [27].

Vico-Greengard

This comparatively new method, introduced in 2016 [25] by Vico et al. (henceforth called Vico's method), can be seen as a feasible alternative to the aforementioned Hockney method. Given sufficient smoothness of $\rho(\boldsymbol{r})$, it shows spectral convergence: $\mathcal{O}(\gamma^N)$, where $\gamma \in (0, 1)$. Due to the oscillatory nature of the truncated Green's function it requires a mesh of $(4N)^d$ (zero-padded as in Hockney's case). But luckily the authors showed that one can precompute the Green's function on $(4N)^d$ in an initialization step and subsequently restricting it to $(2N)^d$ without loosing fidelity in the solution. Thus for a fixed Green's function, solving the actual Poisson equation uses at most as much memory as Hockney's method.

We must point out that the authors not only provide a formulation of Green's function for the regular Poisson problem but also for several other commonly encountered PDEs, including the biharmonic equation which we also make use of for solving Equation 2.16.

3.2 Particle-in-Cell Method

As explained in the previous chapter, we are interested in modeling the time evolution of the phase space $f(\boldsymbol{r}, \boldsymbol{v})$ in a single-species plasma. In Equation 2.12, one of the dominating terms governing the dynamics of the particles in such a plasma is defined by long-range interactions via the self-generated electrical field. PIC methods are one of the most efficient and widely used methods to resolve such long-range interactions in areas of research such as fluiddynamics [28, 9], astro physics and plasma physics [29, 30].

Its strength lies in the ability to combine two traditional numerical techniques to describe the flow of a field, known as the Eulerian and Lagrangian description. PIC uses a computational mesh in combination with so called macroparticles, which are able to move freely between the cells of the mesh according to Newton's equation of motion. Meshes in a Eulerian system are suitable for resolving distortions in the medium but show an undesired diffusive property in moving discontinuities. The Lagrangian particle view on the other hand, allows to properly resolve the latter but shows deficiencies when strong distortions appear [31].

In our case, the particles implicity represent the phase space distribution $f(\boldsymbol{r}, \boldsymbol{v})$ and carry other macroscopic system properties important to our simulation. We will now illustrate two common operations that are essential to the PIC algorithm.

Given a charge density defined by the particle positions in configuration-space, we are interested in solving the Poisson equation (cf. Eq. 2.10) with the PIC method and subsequently computing the interaction forces induced by the self-generated electrical field $\boldsymbol{E}(\boldsymbol{r})$. Via the so called `Scatter` operation we are able to distribute charges, carried by the particles that are part of one cell, to itself and neighboring cell values. This charge assignment is treated in further detail in [32]. We can then solve for the electric potential $\phi(\boldsymbol{r})$ and electric field $\boldsymbol{E}(\boldsymbol{r})$ with any common mesh-based method (i.e. FFT or FD stencils). The $\boldsymbol{E}(\boldsymbol{r})$ field in this case, needs to be interpolated back (i.e. `Gather`) from cells to the particles in its immediate neighborhood in order to add force contributions to their trajectory of motion. In alternating these two steps we are then able to simulate how the system evolves over time.

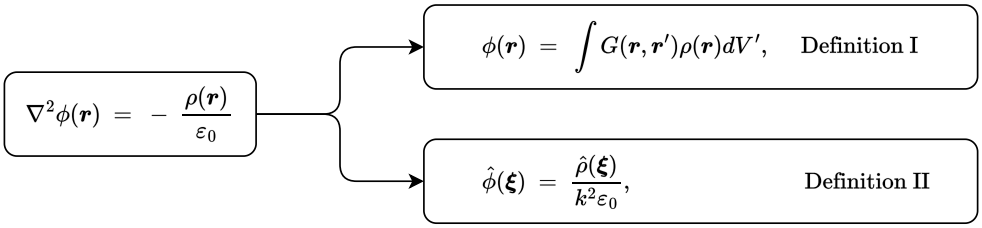


Figure 3.1: Two mathematically identical formulations of the solution to the Poisson problem 2.10. As used in Table 3.1.

P^{3}M

The P^{3}M method is an extension to the PIC scheme that additionally models **short-range** particle-particle (PP) interactions. Hence the name (Particle-Particle)-(Particle-Mesh) method (P^{3}M). The method devised by Hockney and Eastwood [26] has gained popularity in settings where we expect to encounter non-uniform particle distributions. The **long-range** Particle-Mesh (PM) forces acting on the particle's position in phase space are computed via a solving Poisson's equation with the PIC method (in Fourier space). This is identical to what we are doing in our collisionless solver. The **short-range** PP-interactions are computed in real-space by considering pairwise Coulomb interactions with particles contained in a sphere of radius r_c (chosen by the user). This approach introduces considerable a computational cost as we have to do bookkeeping for storing and updating the neighboring particles for each individual particle and quickly becomes infeasible for larger cut-off radii or higher density plasma. We direct the reader to [26] for a detailed treatise of the algorithmic details of the P^{3}M algorithm.

Table 3.1: Overview of explored methods for the various quantities needed by the Langevin solver.

PIC Type	**Quantity of Interest**	**Comp. Domain or Method**
Electrostatic PIC	$\phi(\boldsymbol{r})$	Definition I (see below)
		Definition II (see below)
	$-\nabla_{\boldsymbol{r}}\left[\phi(\boldsymbol{r})\right]$	Finite Difference Gradient: $\nabla^{\mathrm{fd}}_{\boldsymbol{r}}$
		Spectral Gradient: $\nabla^{\mathrm{sp}}_{\boldsymbol{r}}$
Velocity PIC	$\nabla_{\boldsymbol{v}}h(\boldsymbol{v})$, $g(\boldsymbol{v})$	Hockney, $\nabla^{\{\mathrm{fd,sp}\}}_{\boldsymbol{v}}$
		Vico, $\nabla^{\{\mathrm{fd,sp}\}}_{\boldsymbol{v}}$
	$\frac{\partial^2}{\partial\boldsymbol{v}\partial\boldsymbol{v}}g(\boldsymbol{v})$	Finite Difference Hessian: $\underline{\underline{H}}^{\mathrm{fd}}_{\boldsymbol{v}}$
		Spectral Hessian: $\underline{\underline{H}}^{\mathrm{sp}}_{\boldsymbol{v}}$

3.2.1 Methods Overview

We have access to a P^{3}M solver by Ulmer [2] and can therefore readily apply it to the DIH problem to observe how including short-range interactions in the force computation impacts the normalized emittance. One crucial difference of Ulmer's collisionless solver to ours is, that he

uses a Green's function directly defined in real space (see Definition I in Figure 3.1) to compute the potential ϕ. Whereas we define it via the integral formulation in Fourier space (Definition II). Although these two formulations are mathematically identical under the periodicity assumption, we will observe two quite different outcomes in the normalized emittance. Ulmer defines the Green's function as:

$$G(\boldsymbol{r}, \boldsymbol{r}') = \frac{1}{4\pi\varepsilon_0} \frac{\operatorname{erf}(\alpha|\boldsymbol{r} - \boldsymbol{r}'|)}{|\boldsymbol{r} - \boldsymbol{r}'|}, \tag{3.2}$$

where $\alpha = 1 \times 10^6$ is a parameter indicating the splitting of interactions into the PM and PP parts. Subsequently, we refer to a Langevin solver instance implementing Definition I as *Langevin-I* and analogously *Langevin-II* for an instance employing Definition II.

So far we introduced a variety of building blocks we can use for the Langevin solver. Thus, it might be difficult for the reader to follow the subsequently tested solver combinations. In Table 3.1 we collect all the methods of computation we explored for each quantity of interest (QoI) we need as part of our Langevin solver.

3.3 Discretization of the Rosenbluth Potentials

After having derived expressions for both Rosenbluth potentials in Section 2.5, we now show how we compute them numerically. As the name already suggests, they can be treated as potentials, similarly to what we would encounter computing an electric field defined by distributed particle charges (see Equations 2.17 and 2.18).

We have already touched upon the fact that it is a reasonable assumption to solve them in velocity space exclusively. Since we expect our distribution to respect $\lim_{\|\boldsymbol{v}\|_2 \to \infty} \int f(\boldsymbol{r}, \boldsymbol{v}) d\boldsymbol{r} = 0$ [20], the open-boundary solvers as introduced in Section 3.1 may be used here.

The friction and diffusion coefficients are readily computed from the potentials either by directly differentiating them in Fourier space (e.g. [33]) or using a second order centered finite difference stencil. It has to be noted that the FD approach would not respect the limit $\lim_{\|\boldsymbol{v}\|_2 \to \infty} g(\boldsymbol{v}) = 0$ (as to the argument of Hinton [20]).

Composable Finite Difference Operators

In a situation where it is necessary to respect these boundary conditions with a FD stencil, one could explore one-sided stencils along the dimension with open boundary conditions. We implemented a general interface for composing user defined stencils along either dimension which are generated at compile time. This allows us to define more complex operator structures such as the Hessian. Further details on this tool with additional convergence studies can be found in Appendix E. Although, as we will show with numerical experiments in Chapter 4, there is no need for such a flexible operator. The potential $g(\boldsymbol{v})$ is indeed vanishingly small at the mesh boundaries given the properties of the DIH problem (see Section 2.7).

After computing the coefficients on the grid, we scatter their values onto the macroparticles. This way, we can use them directly in the time integrator described in the next section.

3.3.1 Stochastic Description

In Chapter 2.4 we discussed how collisions can be thought of as a Markov process where a particle experiences a small change in velocity due to many weak coulomb collisions with other particles passing through its Debye sphere. This derivation lead us to a formulation of the right-hand side of the Vlasov Equation 2.9 containing a dynamic friction $\boldsymbol{F}_d$ and diffusion coefficient $\underline{\underline{D}}$ based

on the Langevin equation. The coefficients themselves are approximated with the help of the Rosenbluth potentials.

Based on these calculations, Risken [34] formulates the time evolution of the Vlasov-Poisson-Fokker-Planck (VPFP) equation as the following system of coupled differential equations governing the motion of the macroparticles in the PIC approach:

$$\begin{cases} \dfrac{\partial \boldsymbol{r}}{\partial t} = \boldsymbol{v} \\ \dfrac{\partial \boldsymbol{v}}{\partial t} = \dfrac{(\boldsymbol{F}_{mf} + \boldsymbol{F}_{ext})}{m} + \boldsymbol{F}_d + \underline{\underline{Q}} \cdot d\boldsymbol{W}(t)^T. \end{cases} \tag{3.3}$$

Diffusion Matrix Factorization

In Section 2.4 we have listed an important property of which we can now make use of to determine an appropriate factorization of $\underline{\underline{D}}$. We know that $\underline{\underline{D}}$ is at least semi-positive definite [8]. Thus, one feasible algorithm would be the LDLT factorization, which has the advantage that we can avoid computing square-roots on the immediate matrix entries of $\underline{\underline{D}}$ [35]. This means we can factorize $\underline{\underline{D}} = \underline{\underline{L}}\underline{\underline{S}}^2\underline{\underline{L}}^T$ such that $\underline{\underline{Q}} = \underline{\underline{S}}\underline{\underline{L}}^T$ with $\underline{\underline{S}}$ being a diagonal matrix. See Appendix C for the details on how to compute $\underline{\underline{Q}}$ from $\underline{\underline{L}}$ and $\underline{\underline{S}}$. We point out that the default Cholesky factorization would not be suitable, as for this method to work, the matrix has to be positive definite.

3.4 Resulting Scheme

We have now successfully brought the VPFP equation into a form that we can start thinking about an appropriate timestepping scheme. We opt for a simple but frequently used timestepping scheme that facilitates the Stochastic Differential Equation (SDE) form of our coupled system of Equations 3.3. It is based on the Euler-Maruyama scheme in conjunction with integrator splitting:

Algorithm 1 Time Integrator Procedure

1: **procedure** ADVANCE PARTICLES IN TIME BY dt
2: $\boldsymbol{r} \leftarrow \boldsymbol{r} + \frac{dt}{2}\boldsymbol{v}$.
3: Compute $\boldsymbol{F}(\boldsymbol{r})$; $\boldsymbol{v} \leftarrow \boldsymbol{v} + \frac{dt}{2}\frac{\boldsymbol{F}}{m}$.
4: Compute $\boldsymbol{F}_d(\boldsymbol{v})$ and $\underline{\underline{D}}(\boldsymbol{v})$
5: Factorize $\underline{\underline{D}}(\boldsymbol{v})$; $\underline{\underline{Q}} \leftarrow \underline{\underline{S}}\underline{\underline{L}}^T$
6: $\boldsymbol{v} \leftarrow \boldsymbol{v} + dt\boldsymbol{F}_d + d\boldsymbol{W}(t) \cdot \underline{\underline{Q}}$.
7: Compute $\boldsymbol{F}(\boldsymbol{r})$; $\boldsymbol{v} \leftarrow \boldsymbol{v} + \frac{dt}{2}\frac{\boldsymbol{F}}{m}$.
8: $\boldsymbol{r} \leftarrow \boldsymbol{r} + \frac{dt}{2}\boldsymbol{v}$
9: **end procedure**.

We have chosen this scheme as it have been shown to work well for integrating the VPFP equation, albeit in a different setting [8]. It exhibits a symmetric update structure w.r.t. step 4 through 6. This would enable subcycling on the collisional terms if necessary (similar to Adam [36]).

Boundary Conditions

When solving PDEs, boundary conditions are of utmost importance for achieving uniqueness and physical correctness. In our problem setting we solve the PDEs arising from the Fokker-Planck equation in velocity phase space with open boundaries. For the computation of the self-generated electric field we choose periodic boundary conditions. This was mainly chosen to facilitate a direct comparison to Ulmer's results [2] (see Section 4.1).

3.4.1 Advantages over P^{3}M

Having introduced all important parts of the Langevin method, we can elaborate on the advantages it brings compared to the P^{3}M method. The user of the P^{3}M method by Ulmer would need to set two hyperparameters: α, defining the splitting ratio of Green's function interaction potential into a long-range and short-range part, as well as the cut-off radius r_c which defines the range of the PP collisions. Although not strictly a hyperparameter but still impactful, there exist a multitude of charge-shaping functions to choose from (i.e. Gaussian, polynomials). This can make it a delicate endeavor to properly tune the method for a certain problem.

The Langevin method on the other hand regulates the effect of collision by choosing the Coulomb logarithm $\ln \Lambda$. It is used in the derivation of the Fokker-Planck coefficients for defining a cut-off for weak Coulomb interactions at large impact parameters. Thus, in a sense it serves a similar purpose as r_c in the P^{3}M method. Depending on the state of plasma one can choose its magnitude appropriately as it is dependent on the system's temperature.

Computational Complexity

The computational complexity is an important criterion to understand in what cases one algorithm might be superior to another. The Langevin approach significantly improves upon the computational complexity of the P^{3}M method.

For P^{3}M it can be split up into the PP and the PM part:

$$C_{\mathrm{P^3M}}(N_p, N_m, \delta) = \underbrace{\mathcal{O}(N_p^2 \delta^3)}_{\text{Particle-Particle}} + \underbrace{\mathcal{O}(N_p) + \mathcal{O}(N_m \log(N_m))}_{\text{Particle-Mesh}}, \tag{3.4}$$

with $\delta = r_c/L$ for a uniform charge distribution, where L is the domain length. The $\mathcal{O}(N_p)$ term accounts for the scatter/gather operations.

The PM term is also present in the Langevin method, while the term for the particle particle interaction changes due to the computation of the Fokker-Planck coefficients. We can omit constant coefficients, resulting in it being asymptotically almost identical to the regular PIC solver, with the exception of the added $\mathcal{O}(N_m)$ term for the decomposition of the diffusion coefficients $\underline{\underline{D}}(\boldsymbol{v})$ used in the time integrator:

$$\begin{aligned} C_{\mathrm{Langevin}}(N_p, N_m) &= \underbrace{\mathcal{O}(N_p) + \mathcal{O}(N_m \log(N_m))}_{\boldsymbol{F}_d} \\ &\quad + \underbrace{\mathcal{O}(N_p) + \mathcal{O}(N_m) + \mathcal{O}(N_m \log(N_m))}_{\underline{\underline{D}}} \\ &\quad + \underbrace{\mathcal{O}(N_p) + \mathcal{O}(N_m \log(N_m))}_{\text{Particle-Mesh}} \\ &= \mathcal{O}(N_p) + \mathcal{O}(N_m \log(N_m)). \end{aligned} \tag{3.5}$$

3.5 Test Cases

To evaluate the correctness of our implementation we use simple initial conditions to which we know the analytical solution of the potentials and coefficients discussed in Chapter 2. This also gives us the ability to carry out studies on the error convergence with respect to decreasing mesh width.

Rate of Convergence

When evaluating the performance of numerical methods, an important criterion can be to determine the achieved rate of convergence. With this we mean, how much better of an approximation we can expect to obtain when increasing the discretization level (i.e. number of grid points) our method uses. There exist two types of convergence we will encounter in Chapter 4. We illustrate their asymptotic behavior on the example of the relative approximation error η with increasing grid points N_o (see Table 2.1):

$$\eta(x, x_{\text{appr}}) = \mathcal{O}(N_o^{-r}), \qquad \text{algebraic convergence with rate } r, \tag{3.6}$$

$$\eta(x, x_{\text{appr}}) = \mathcal{O}(\gamma^{N_o}), \qquad \text{exponential convergence with } \gamma \in (0, 1), \tag{3.7}$$

where x denotes the ground truth value (i.e. the analytical solution) and x_{appr} the approximation given by the applied numerical method.

3.5.1 Centered Gaussian

We use a centered Gaussian probability density function in with curvature σ as our starting point to emulate the phase space density $f(\boldsymbol{v})$ in three-dimensional velocity space. Each grid point in this space is associated with a Cartesian velocity vector $\boldsymbol{v} = [v_x, v_y, v_z]^T$ and a respective euclidean norm $v = \|\boldsymbol{v}\|_2 = \sqrt{v_x^2 + v_y^2 + v_z^2}$. The resulting density is then described as

$$f(\boldsymbol{v}) = \frac{1}{\sqrt{8\pi^3}\sigma^3} \exp\left(-\frac{v^2}{2\sigma^2}\right). \tag{3.8}$$

The potentials by Rosenbluth (Eq. 2.17 and Eq. 2.18), as well as their derivatives, can be analytically derived and are listed in Equations 3.9. In the following formulation of the solutions we have already accounted for the prefactor of 8π that appears in Eqs. 2.17 and 2.18. They reflect what has been implemented in the computer code for this test case:

$$h(\boldsymbol{v}) = \frac{2}{v}\mathrm{erf}\left(\frac{v}{\sqrt{2}\sigma}\right) \tag{3.9a}$$

$$g(\boldsymbol{v}) = \sigma\sqrt{\frac{2}{\pi}}\exp\left(-\frac{v^2}{2\sigma^2}\right) + (v + \frac{\sigma^2}{v})\mathrm{erf}\left(\frac{v}{\sqrt{2}\sigma}\right) \tag{3.9b}$$

$$\nabla h(\boldsymbol{v}) = \left[\sqrt{\frac{2}{\pi}}\frac{1}{\sigma}\exp\left(-\frac{v^2}{2\sigma^2}\right) - \frac{1}{v}\mathrm{erf}\left(\frac{v}{\sqrt{2}\sigma}\right)\right]\boldsymbol{v} \tag{3.9c}$$

$$\begin{aligned}\underline{\underline{D}}_{i,i}(\boldsymbol{v}) = \Gamma\sqrt{\frac{2}{\pi}}\frac{1}{\sigma}\exp\left(-\frac{v^2}{2\sigma^2}\right) \\ \left[\frac{\boldsymbol{v}_i - \sigma^2}{\sigma^2} + \frac{(\sigma^2/v + v)(\sigma^2 v^2 - \boldsymbol{v}_i^2 v^2 - \sigma^2\boldsymbol{v}_i^2)}{\sigma^2 v^3} + \frac{2\boldsymbol{v}_i^2}{r^4}(v^2 - \sigma^2)\right] \\ + \frac{1}{v^5}\mathrm{erf}\left(\frac{v}{\sqrt{2}\sigma}\right)\left[v^4 - \boldsymbol{v}^2(\boldsymbol{v}_i^2 + \sigma^2) + 3\sigma^2\boldsymbol{v}_i^2\right], \quad \forall i \in [1,3]\end{aligned} \tag{3.9d}$$

$$\underline{\underline{D}}_{i,j}(\boldsymbol{v}) = \Gamma\left[\left(\frac{3\sigma^2}{v^2} - 1\right)\frac{\boldsymbol{v}_i\boldsymbol{v}_j}{v^3}\ \mathrm{erf}\left(\frac{v}{\sqrt{2}\sigma}\right) - 3\sigma\sqrt{\frac{2}{\pi}}\frac{\boldsymbol{v}_i\boldsymbol{v}_j}{v^4}\exp\left(-\frac{v^2}{2\sigma^2}\right)\right],\ \forall i \neq j. \tag{3.9e}$$

Maxwellian

Multiplying Eq. 3.8 with the number density n_0 and choosing $\sigma = v_{th}$, we obtain a centered Maxwellian distribution. As we expect the velocity distribution function to relax to a Maxwellian equilibrium distribution, it equips us with another important test case that closer resembles the problem at hand, which is the disordered induced heating of a cold electron sphere (cf. Chapter 3.2.4.2 in Callen [14]).

Chapter 4

Results

In this Chapter we present the results on our Langevin solver presented in the previous chapter. At first, we summarize the hardware and software tools we have used for our experiments, then we introduce the reference implementation of a Vlasov-Poisson solver that pursues a different ansatz for modeling the collisions. We continue with a section where we test the collisionless solver, followed by a thorough examination of both collisional terms (friction and diffusion) by means of the analytical test case from Section 3.5. Lastly we apply our Langevin solver to the DIH problem and investigate the computed collisional terms emanating from it. Throughout this Chapter we refer to our implementation as "Langevin solver", regardless of its current experiment type (with or without collisions enabled).

4.1 Simulation Setup

The solver was programmed entirely in C++. For post-processing we used Python. We use the Independent Parallel Particle Layer (IPPL) [37], a backend of the accelerator simulation tool OPAL [38], providing many tools and data structures instrumental to PIC codes such as ours. Through the usage of the Kokkos Performance Portability Ecosystem [39] we achieve hardware independent code. Meaning, we can compile our code for a multitude of computing hardware (CPU, GPU, ...) without necessary changes to the code itself.

The default experiment type uses a mesh size of 256^3 and 64^3 for discretizing the configuration and velocity space respectively (denoted by $[256]^3$, $[64]^3$ in the figure labels). We choose to use periodic boundary conditions in configuration space to facilitate a direct comparison between our PIC solver and Ulmer's. As Ulmer correctly states, this is a reasonable assumption due to the large ratio between the chosen box length and the sphere radius (see A.2). In velocity space we use open boundary conditions, as motivated in Section 3.3. Due to the velocity domain being fairly large we found that Hockney's method was not able to converge to an acceptable solution for the Rosenbluth potentials given our chosen mesh size. This is why we adopted domain normalization to $[-1, 1]$ in order to solve for the Rosenbluth potentials. If there are cases where we deviate from this setup we will state it explicitly. Further simulation parameters are listed in Appendix A.1.

We run our experiments on a cluster of the Paul Scherrer Institut containing Nvidia GDX A100 GPUs. Each unit contains 40GB of memory and works in conjunction with an AMD Rome 7742 CPU with 1 TB of system memory. An experiment of the Langevin solver for 5 plasma periods takes approx. 1 minute. Therefore the implementation didn't require parallelization over multiple GPUs.

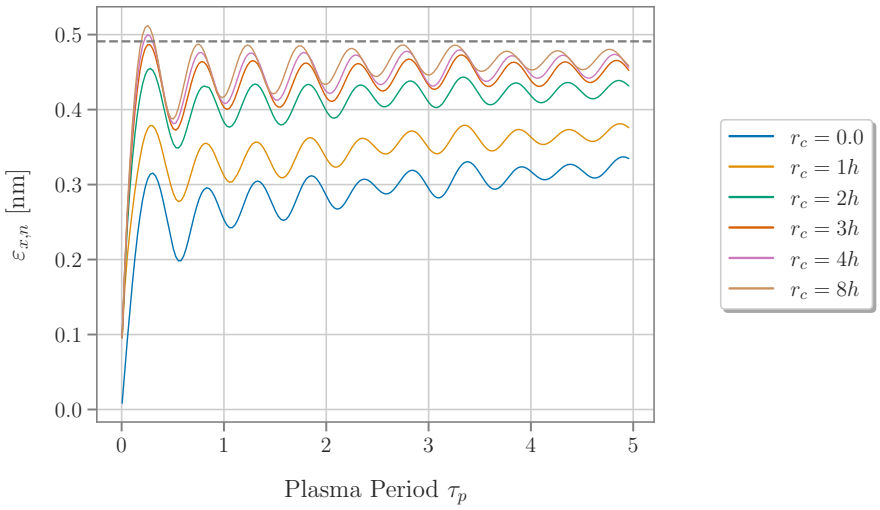


Figure 4.1: Normalized emittance computed by the P^{3}M method [2] for varying cut-off radii $r_c(h)$, where $h = 0.39\mu$m is the mesh width. The dashed gray line signifies the expected limit for the normalized emittance.

4.1.1 Disorder Induced Heating in a Cold Sphere

We strive to replicate the same benchmark case as in [40] and [2]. This benchmark simulates a cold electron sphere with radius $R = 17.74\mu$m containing $N_p = 156055$ electrons and produces a bunch density of $n_0 = 6.67 \times 10^{18}\text{m}^{-3}$. The bunch is initially at rest (zero velocity distribution), but quickly starts to expand due to the self-generated electric field. By applying a linear focusing force given by the average electric field multiplied by a focusing factor $f_c = 1.5$, we can restrict this expansion and cause it to approach an equilibrium state. Given the aforementioned properties of the system, the plasma oscillates at a frequency of $\omega_p = 1.45 \times 10^{-11}\text{s}^{-1}$, equating to a period of $\tau_p = 4.31 \times 10^{-11}$s. Mitchell [40] states the analytical value for the normalized emittance in this equilibrium as being $\varepsilon_{x,n}^{\text{eq}} = 0.491$nm. We choose to run our experiments for 1000 timesteps over 5 plasma periods. This results in a timestep size of $dt = 2.15623 \times 10^{-13}$s, which is coincides with the one chosen by Ulmer [2].

In Figure 4.1 we show the normalized emittance for varying r_c for the DIH problem using the P^{3}M method. It is evident that considering a larger neighborhood for the PP interaction pushes the fluctuating normalized emittance to the expected value of $\varepsilon_{x,n}^{\text{eq}} = 0.491$nm.

In the subsequent sections of this chapter we refer to this benchmark solely as the "DIH problem".

4.2 Collisionless Simulation

In the following section we analyze a few selected statistics of our collisionless solver to assert that the implementation of the DIH problem indeed functions as we expect it to.

At first, we show that the Lorentz factor for the DIH problem stays close to 1 (see Figure 4.2), meaning that our particles do not accelerate indefinitely (remaining in the non-relativistic regime), and thus also indicating that we chose an appropriate focusing strength ($f_c = 1.5$).

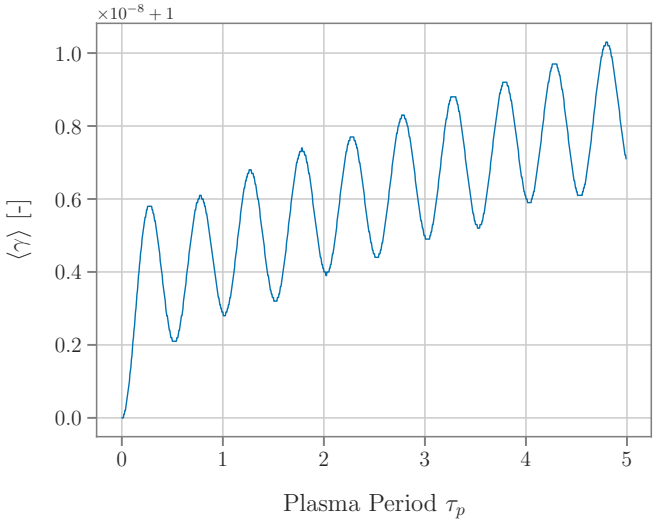


Figure 4.2: Ensemble averaged Lorentz factor γ. Note the multiplicative factor of $\times 10^{-8}$ shown at the top left corner. Plasma period $\tau_p = 4.31 \times 10^{-11}$s.

As we have seen in Table 3.1, the Langevin solver can consist out of any combination of the given sub-methods. We shall now analyze a subset of combinations in the electrostatic PIC solver. The emittance values shown in Figure 4.3a appear to be impacted heavily by either how we choose to solve for the potential ϕ, as well as the way we compute its gradient. It comes unexpectedly that the Langevin-II solver in conjunction with a spectral gradient computation ($\nabla^{\text{sp}}_{\boldsymbol{v}}$) shows an upwards trend, and even surpasses the expected limit of normalized emittance $\varepsilon^{\text{eq}}_{x,n} = 0.491$nm. Apart from this behavior, it agrees very well with the analytical value of the plasma period τ_p. The Langevin-II type solver generally results in larger amplitudes whereas the Langevin-I appear to be more damped. Nevertheless, it is hard to judge which combination results in the most "correct" behavior of the emittance. Ulmer's results also have to be considered as just an approximation of the real behavior.

Thus we would like to conclude, that the choice of these two building blocks have a non-negligible impact resulting in strongly different behaviors for a longer simulation time. We would suggest further investigation along these lines if one is interested in a solver optimally tuned for the DIH problem. The justification for this decision being, that we would be able to observe an impact of Coulomb collisions on the normalized emittance irrespective of the chosen structure of the electrostatic PIC solver.

The choice of the appropriate mesh size can also prove to be delicate, as we have to find a trade-off between a small enough mesh width h that we are able to resolve small-scale effects and at the same time respecting given memory or compute time constraints. It can be seen in Figure 4.3b that a spatial discretization of size 256^3 is necessary for both solvers to show the oscillatory behavior we would expect in the normalized emittance.

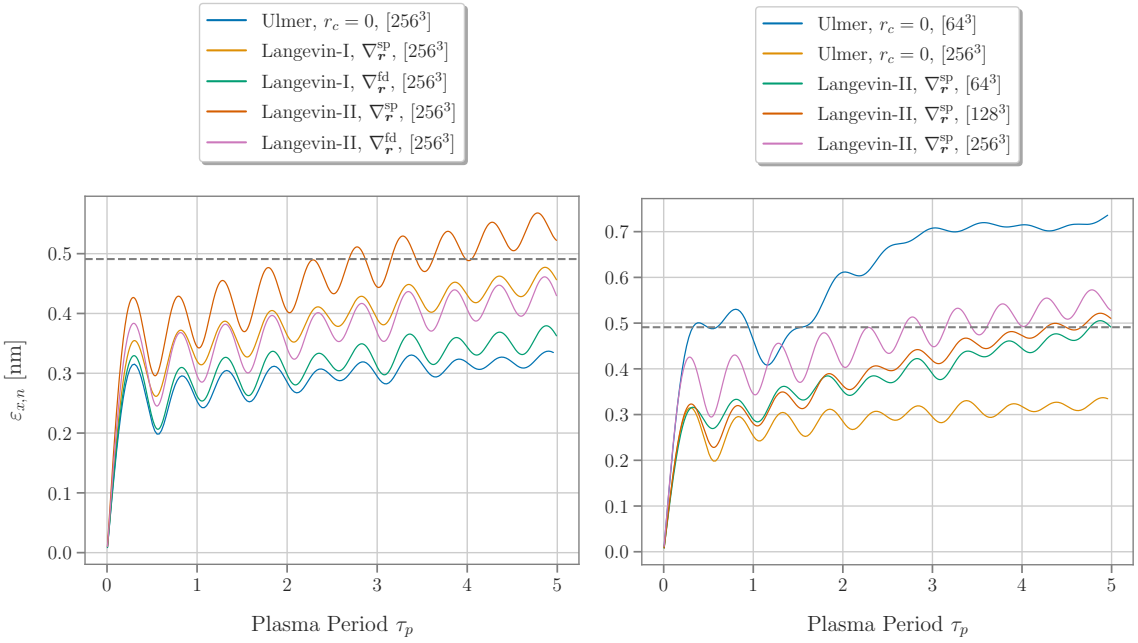


Figure 4.3: Study on how the Poisson solver type impact on the normalized emittance $\varepsilon_{x,n}$ (4.3a). Impact of varying spatial discretization on normalized emittance computed by the reference P^{3}M solver [2] ($r_c = 0.0$) and the Langevin electrostatic PIC solver (4.3b). The dashed horizontal line indicates $\varepsilon^{\text{eq}}_{x,n}$.

4.3 Gaussian Test Case

Via the formulations in Equations 3.9 we are able to test individual components arising from the Rosenbluth potentials in a controlled setting and verify their correctness on an initial velocity density distribution given by Eq. 3.8. We choose initial conditions that are similar to a snapshot of the three-dimensional velocity distribution at iteration 100 ($t = 0.0215$ns) of the DIH problem as then we expect the collisions to be a dominant contribution to the system dynamics. The resulting velocities are sampled from $\mathcal{N}(0, \sigma)$ with $\sigma = 0.05 v_{max}$.

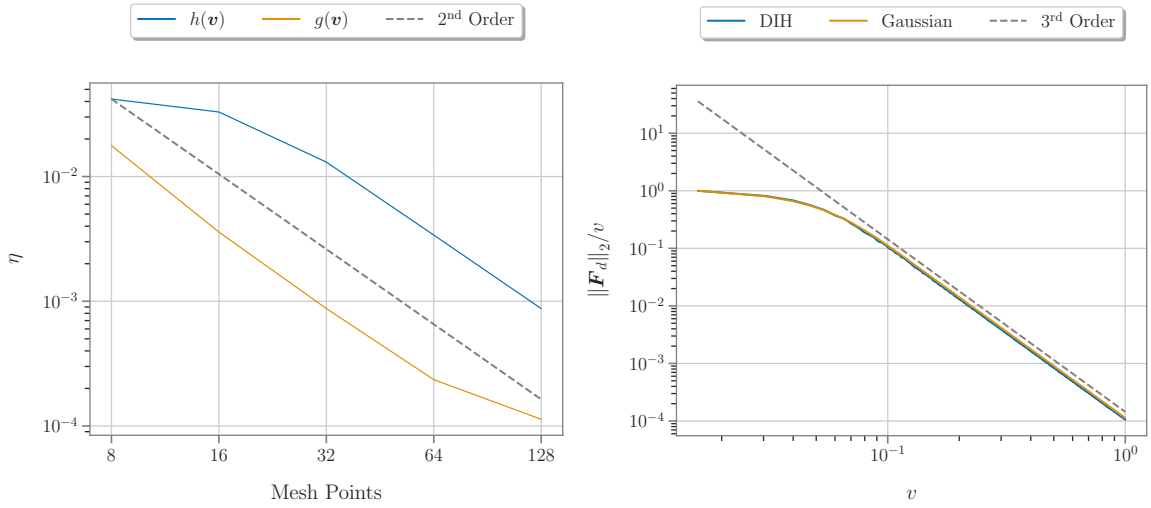
The potentials were solved for with Vico's method and exhibit a consistent order 2 convergence of the relative error (see Figure 4.4a). The same holds for Hockney's method (Appendix B). This might come as a surprise as we would expect exponential convergence for Vico's method. But as mentioned in Subsection 3.1 this only holds for sufficiently smooth functions. In our case this is not given as on the one hand, our particle distribution is peaked around the origin and on the other, we use the particle scatter operation to obtain the particle density in velocity space. This causes a non-smooth density due to the finite mesh size and particle number. Although we could either increase the particle density or use a higher order interpolation method (for the scatter/gather operations) for further smoothness, the computation would quickly become too expensive in terms of memory usage and computation time. In particular, we have observed when the diffusion term was enabled in the time integrator. As we need multiple temporary datastructures during the computation of $d\boldsymbol{W} \cdot \underline{\underline{Q}}$.

So far we cannot conclusively say which method is preferable. Regardless, we propose to use

Vico's method to be on the safe side. If one would ever encounter a smoother velocity distribution (i.e. due to the aforementioned improvements to the PIC routines), the method is proven to show better convergence. Conveniently, it also exhibits similar computational complexity as Hockney's method.

Hereafter we may only consider results computed with Vico's method to avoid distracting the reader with too many figures for both solver types. We will state explicitly if there is some data that nonetheless used Hockney's method due to the results bearing unexpected or interesting behavior.

Having shown the expected convergence rate on the potentials we can move on and check it for the individual collisional coefficients, $\boldsymbol{F}_d$ and $\underline{\underline{D}}$.



(a) (b)

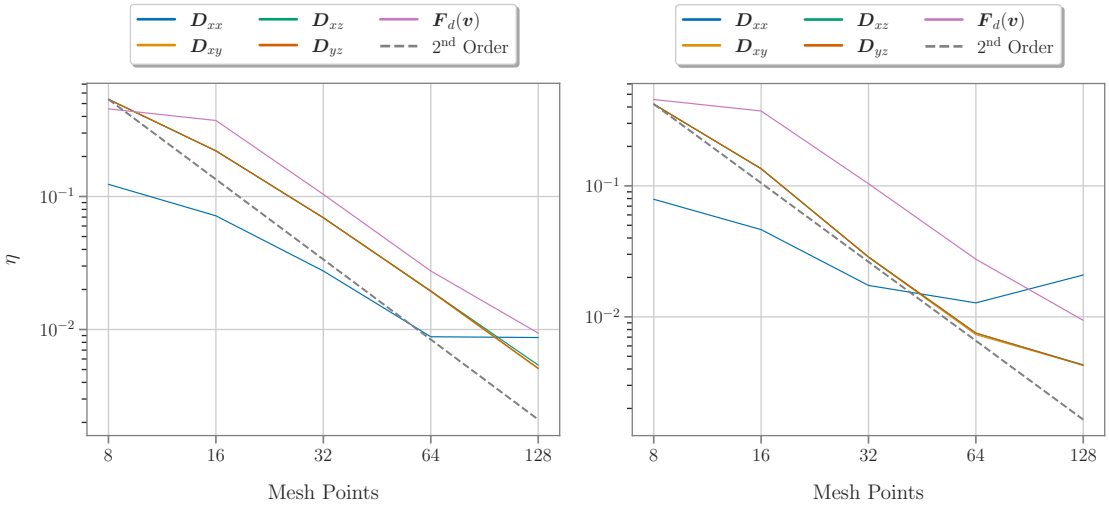
Figure 4.4: Convergence study of Rosenbluth potentials solved with Vico's method (4.4a). The normalized friction coefficient distributions for both the Gaussian test case and the DIH problem after 5 plasma periods (4.4b). Both exhibit the expected asymptotic $1/v^3$ fall-off (dashed gray line).

4.3.1 Friction & Diffusion Coefficient

At first, we are interested in how the values of the friction coefficients are distributed. We expect large friction terms for small deflections/scattering. Furthermore it should decrease at an asymptotic rate of $1/v^3$ at large scatterer velocity as specified by [5]. The expected asymptotic fall-off is distinctly visible on a normalized log-log scale in Figure 4.4b. The same behavior has been seen in by Qiang (cf. Figure 4 of [41]) for their specific test case of a Maxwellian velocity distribution.

As for the convergence (Figure 4.5): $\boldsymbol{F}_d(\boldsymbol{v})$ shows a similar order 2 error convergence, but we observe a difference for computing $\underline{\underline{D}}(\boldsymbol{v})$ via finite difference or via the spectrally accurate method via the Fourier transform (as a byproduct of the computation of the Rosenbluth potential itself). Interestingly enough the spectral computation performs worse, at least for this case of the velocity distribution. After increasing our mesh size above 64^3 grid points, especially the accuracy of the diagonal elements of the Hessian starts to deteriorate. This suggests choosing

the finite difference approximation of the Hessian, as the diagonal elements tend to be dominant ones [5] as well as to use 64^3 as our default mesh size in velocity space. We follow up on the unexpected behavior of the diffusion coefficient with a more in depth analysis in Section 4.4.



(a) Finite difference computation of coefficients.

(b) Spectral computation of coefficients.

Figure 4.5: Convergence study of collisional coefficients for a Gaussian velocity distribution which models the distribution of the DIH problem.

Cholesky Decomposition

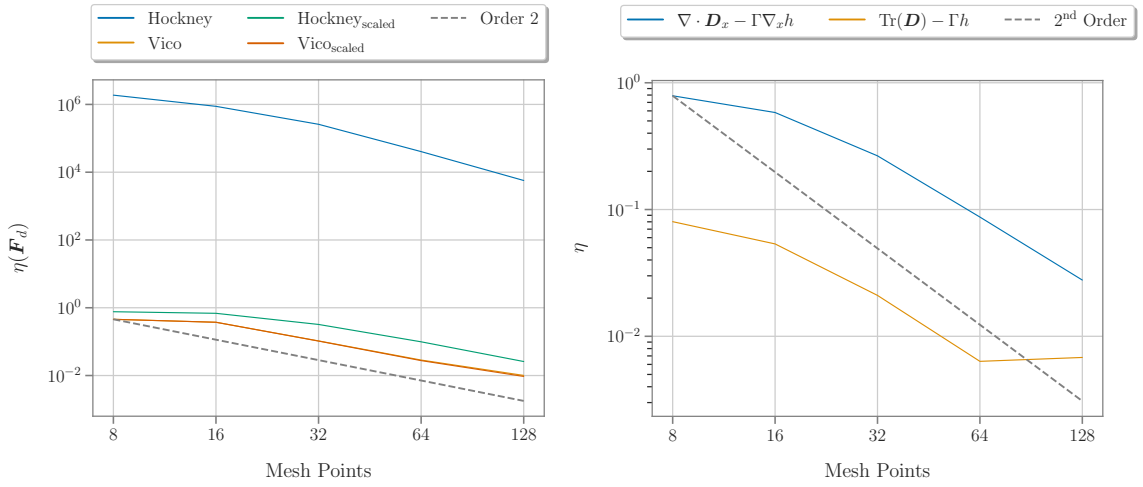
We implemented the LDLT algorithm listed in Appendix C to achieve the Cholesky decomposition of the Diffusion matrices $\underline{\underline{D}}(\boldsymbol{v})$. Using the fact that for any real matrix $\underline{\underline{A}}$, $\underline{\underline{A}} \cdot \underline{\underline{A}}^T$ is semi-positive definite, we generated 10^4 random semi-positive definite matrices and computed their decompositions. We then compared it to the decompositions obtained with the SciPy [42] Python module `scipy.linalg.ldl`. We found that all decompositions coincided.

Coefficient Identities

In Section 2.5 we derived identities that relate $\boldsymbol{F}_d$ and $\underline{\underline{D}}$ with each other. We observe the expected error convergence of second order for both identities (Figure 4.6b). Although it has to be pointed out that the magnitude of the relative error values are large as compared to the errors we have encountered so far. We also replicated this behavior by implementing it in Matlab [43]. The code of this implementation by A. Cerfon may be found on [44]. We ascribe this to the repeated application of the numerical differential operators causing a non-negligible accumulation of numerical errors.

4.4 Simulation with the Collisional Operator

In this section we test how our Langevin collisional terms are performing in case of the DIH problem. Following the thorough study of potential combinations of solver components outlined



(a) (b)

Figure 4.6: Unlike Vico's method, Hockney's method does not converge on the unnormalized (see subscript "scaled") velocity space (4.6a). Convergence study of the identities defined in Eq. 2.19 computed with Vico's method (4.6b).

Table 4.1: Solver setup for the DIH experiments.

PIC Type	**Quantity of Interest**	**Comp. Domain or Method**
Electrostatic PIC	$\phi(\boldsymbol{r})$	Definition I
	$-\nabla_{\boldsymbol{r}}\left[\phi(\boldsymbol{r})\right]$	Spectral Gradient: $\nabla^{\text{sp}}_{\boldsymbol{r}}$
Velocity PIC	$\nabla_{\boldsymbol{v}} h(\boldsymbol{v})$, $g(\boldsymbol{v})$	Vico + $\nabla^{\text{sp}}_{\boldsymbol{v}}$
	$\frac{\partial^2}{\partial \boldsymbol{v} \partial \boldsymbol{v}} g(\boldsymbol{v})$	Spectral Hessian: $\underline{\underline{H}}^{\text{fd}}_{\boldsymbol{v}}$

in the preceding sections, we have opted to employ the solver setup as stated in Table 4.1 for the subsequent experiments on the DIH problem.

In the course of this investigation we encounter several unexpected outcomes which we each address and try to give the reader an intuition as to why this might be the case.

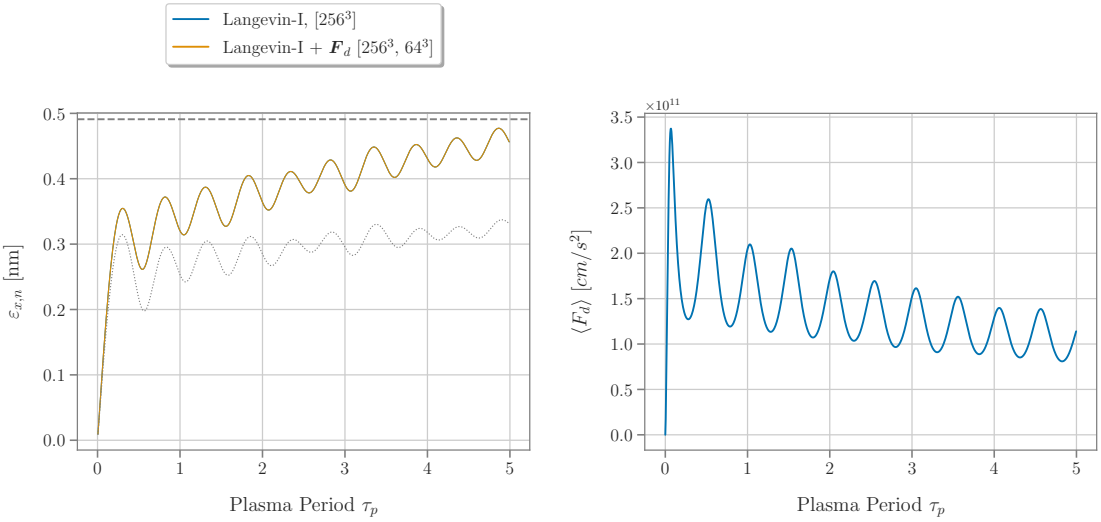
4.4.1 Friction Coefficient

With the help of the Gaussian test case we gained two important insights: $\boldsymbol{F}_d$ indeed exhibits the correct distribution (Fig. 4.4b). Also we showed that with Hockney's method one needs to normalize the velocity domain in order to observe the correct values (Fig. 4.6a).

The integration of the friction coefficient to the DIH time integrator as stated in Alg. 1 does not appear to have an impact on the normalized emittance (see Figure 4.7a). The computed values are in the range of 10^{11} (Fig. 4.7b), which is far too small to have an impact on the particle velocities during one timestep ($dt = 2.15 \times 10^{-13}$s). This finding is unexpected, given that in the previous section, we have demonstrated that the friction coefficient matches well with the analytical solution of the test case. We can think of several possible explanations for this to

happen:

1. We have introduced a unit error, making the coefficient several magnitudes too small.
2. The DIH problem's initial condition violates an assumption made during the derivation of the Rosenbluth potentials or the Fokker-Planck coefficients.
3. The combination of timestep / mesh width does not allow to properly resolve the collisions in velocity space.



(a) (b)

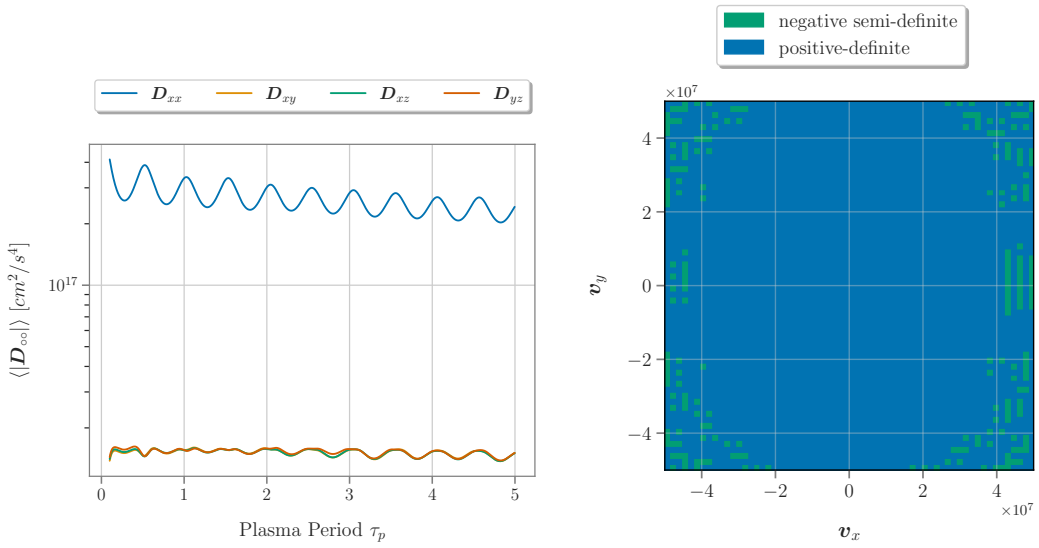
Figure 4.7: Langevin simulation with friction enabled (4.7a). The dotted line represents the emittance with Ulmer's collisionless solver and the dashed gray line the emittance limit. Average norm of the friction coefficient over 5 plasma periods (4.7b).

4.4.2 Diffusion Coefficient

The computation of the diffusion matrix $\underline{\underline{D}}$ works as expected. We plot the behavior of the averaged first diagonal entry and the off-diagonal elements over 5 plasma periods (Fig. 4.8a). Due to the coefficients being zero at $dt = 0.0$, we excluded the first 20 timesteps from the plot. This way we can better inspect the small-scale behavior after the warm up period.

This element-wise analysis confirms that the diagonal values are indeed dominant as we predicted and show that they attain a distinct periodic behavior already early in the simulation. All diagonal elements of the diffusion matrix coincide due to the symmetry of the problem (Fig. D.1). The off-diagonal elements show higher sensitivity to the initial conditions in form of noise. The periodic behavior thus only starts to appear after 3 plasma periods.

Despite the promising results so far, we encounter issues when attempting to find a Cholesky decomposition for the diffusion matrices at the particle locations. Upon analyzing the matrices themselves, it turns out that a small number do not obey the semi-positive definiteness property postulated in Section 2.5.



(a) (b)

Figure 4.8: Elements of computed diffusion coefficients over 5 plasma periods (4.8a). Slice through the diffusion matrix field at $\boldsymbol{v}_z = 0$ and $t = 4\tau_p$ indicating the matrix property for each entry (4.8b).

We state the hypothesis that the property does not hold due to problems in our method and not the mathematical proof being flawed. Hence, we are interested to find in what cases the matrices are not semi-positive definite.

We take a snapshot of the matrix field at the last iteration ($t = 5\tau_p$) and assign each entry a label according to its matrix property. In Figure 4.8b we show a slice through this field at $\boldsymbol{v}_z = 0$ after 4 plasma periods. An unanticipated accumulation of negative definite matrices emerges along the boundaries of the domain. We might speculate that this has to do with the fact that we are using FD to compute $\underline{\underline{D}}$. This rationale is quickly refuted by comparing it with a field computed with the spectrally accurate Hessian which satisfies the open boundary conditions of the velocity domain (see D.2). In fact, these numerical remnants are even more pronounced in the diffusion matrices when computed the Hessian of spectral accuracy.

When plotting these slices at periodic intervals of length τ_p (see Fig. 4.9), we observe that the halo of negative-definite matrices starts to disappear after $t = 3\tau_p$. This corresponds well with our previous observation of the off-diagonal elements reaching stable periodic behavior after the first three plasma periods (Fig. 4.8a). An additional finding, confirming the off-diagonals as the cause for some matrices not being factorizable, is that if we only use the diagonal values of $\underline{\underline{D}}$ for the factorization, all matrices are indeed positive-definite.

There exist a few plausible explanations for observing such behavior in the diffusion matrices. The DIH problem starts from an initial condition (zero velocity distribution) which cannot be assumed to be physical. Therefore we expect the dynamics to behave in a way which may violate one or more assumptions on our diffusion tensor computation. After the system has started to come close to the equilibrium state we see that this issue disappears and our assertions hold.

The pattern which remains after 5 plasma periods appears to consist of concentric circles. Since the probability distribution is peaked around the origin of the domain, and we consider a one-to-one correspondence between macro-particles and electrons, the particle density can

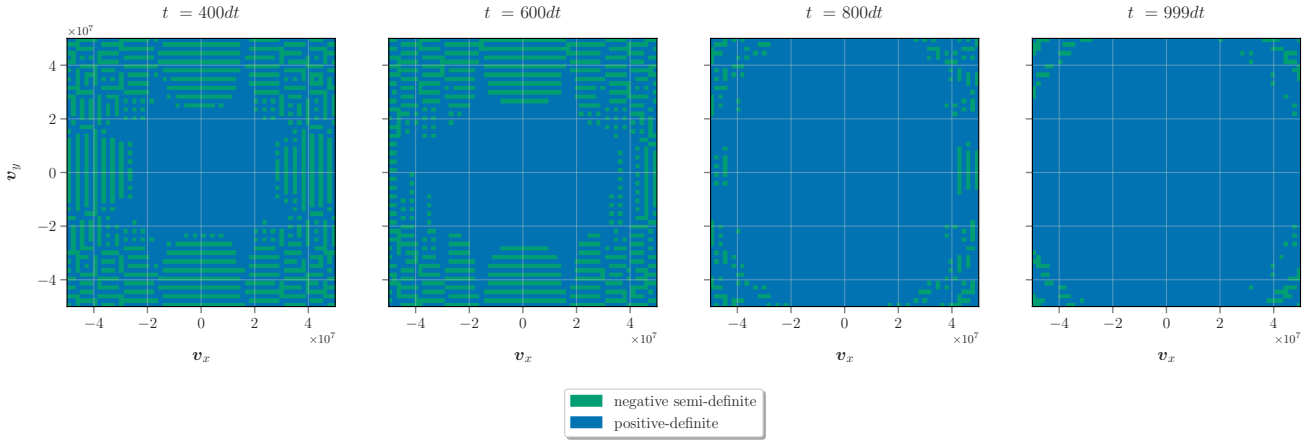


Figure 4.9: Evolution of slice through the diffusion matrix field over time at $\boldsymbol{v}_z = 0$.

become very small towards the boundaries. Due to the large magnitude difference with respect to the density at the origin, it could then be that solving for the Rosenbluth potentials introduces large relative numerical errors in the vicinity of the boundary.

4.4.3 Summary

We showed the correctness of our Langevin type collisional operator for the FP equation on analytical test cases, demonstrating second order convergence for both the dynamical friction $\boldsymbol{F}_d$ and diffusion term $\underline{\underline{D}}$.

Subsequently we applied the operator on the cold sphere DIH problem stated in 4.1.1 and found that the collisional effects introduced by the friction coefficients being too small to impact the normalized emittance of the electron sphere. We propose three hypotheses why we think this might be the case. The diffusion coefficients were shown to occasionally inhibit the decomposition necessary for the time integrator due to negative definiteness. We observed that they exhibit large noise levels on the off-diagonal entries over the course of the first three plasma periods after which they show a regular oscillatory behavior. This change is also reflected in the definiteness property of the computed diffusion matrices, which causes the number of negative definite matrices to decrease drastically from this point onward. We ascribe this observed behavior of the system to the unusual initial conditions where the particles have zero velocity.

Chapter 5

Conclusion

In this work we presented an implementation of the Langevin collision operator via a Particle-in-Cell scheme for modeling the Vlasov-Poisson-Fokker-Planck equation. We explored multiple methods for solving the electrostatic and Rosenbluth potentials by comparing their performance on analytical test cases. Additionally, we demonstrated 2^{nd} order algebraic error convergence for both the dynamical friction $\boldsymbol{F}_d$ and diffusion term $\underline{\underline{D}}$.

Subsequently, we applied the solver to the challenging problem of simulating the disorder induced heating process in an initially cold sphere. Analyzing the impact of the coefficients on the normalized emittance, we found the friction coefficient to be too small to cause any effect. A possible explanation as to why this might be the case, is that the DIH problem starts off from an unphysical zero velocity initial condition. Although we thoroughly compared our method's intermediate and final results with a reference implementation of another type of collisional solver (P^3M), we ultimately cannot rule out the possibility that there is a unit mismatch or similar issue which causes a difference in magnitude of the coefficients. We showed that the diffusion matrices during the first few plasma periods tend to be negative definite when close to the simulation boundary. This property inhibits the Euler-Maruyama time integration method we employed, as it makes use of the matrices' Cholesky decomposition. Further investigation showed that the frequency at which such negative definite matrices occur rapidly decreases after the off-diagonal elements start to show an oscillatory behavior. This indicates that there is some warm-up time in which assumptions on the collisional operator could be violated, leading to these unphysical diffusion matrices.

Through our investigations into appropriate numerical operators for the Rosenbluth potentials, we have also developed a tool that enables the composition of user-defined operators into a single callable instance that is generated at compile time.

5.1 Outlook

The investigation on the Langevin collisional operator gave rise to many unexpected and interesting questions which could be explored in a potential continuation of this project.

We divide this section into suggestions serving two differing purposes:

1. Further investigation on the correctness of the operator when applied on the DIH problem.
2. Possible algorithmic and general performance improvements.

We suggest a continuation of the investigation on the apparent negative definiteness of the diffusion matrices in the early stages of the DIH simulation. Especially, we think it might be beneficial to better understand the origins of the noise seeming to impact the off-diagonal entries. It could help to inspect how the eigenvalues are distributed between the matrices. As an intermediate solution, one could use a diagonal restriction of the afflicted matrices, making them factorizable. The LDLT decomposition we currently use would allow for this to be implemented without much overhead. Although, we point out that one should be aware of the possible negative impact this linearization brings along.

Another direction of research could be, to better understand how the interplay between mesh spacing and timestep influences the collisional coefficient's values. This has partly already been explored by Stoel [8], though we expect it to be of even higher importance in our problem due to the peculiar initial conditions. We suggest exploring this with methods akin to the one used by Adam [36], also known as subcycling.

Since we have shown the correctness of our collisional operator on analytical test cases, it would be interesting to test it on a simpler physical test case to examine if we encounter similar issues as with the DIH problem or not.

This report has been a case study of the applicability of a collisional operator to the DIH problem. Our main focus did not lie on implementing a highly optimized and efficient solver before we could prove its correctness. Nevertheless, during the development process we have found many ways how one could optimize it. The current implementation of the solver supports shared-memory parallelism with OpenMP and SIMD parallelism via single GPU execution. Distributed memory parallelism (via MPI) is currently not supported due to the domain decomposition being defined solely by the particle distribution on the spatial grid. One could implement the concept of super-cells introduced by Qiang [41] which uses separate velocity grids for each MPI rank to circumvent these limitations. Another potential performance improvement concerns the computation of the two collisional terms. It could be carried out completely asynchronously on the GPU via the CUDA streams concept, as their computation is independent up to the point where the collisional coefficients are used to update the particle velocities.

Appendix A

Simulation Parameters

In this section we list the parameters and units used in the Langevin solver. Addtionally, we list the ones used in the reference P^3M implementation by Ulmer [2].

We use partially normalized units in our Langevin solver where it makes sense. They are listed in the Tables A.2 and A.3.

Table A.1: Unit convention used in the Langevin solver.

Type	**Symbol**	**Unit value**	**Normalizing Constant**
Length	[L]	10^{-2}m	-
Time	[T]	1s	-
Mass	[M]	9.109×10^{-31}kg	m_e
Charge	[Q]	1.602×10^{-19}C	q_e

A.1 Simulation Parameters of the Langevin solver

Table A.2: Default parameters of the Langevin Solver.

Parameter	Description	Default value	Unit
r_{beam}	Beam radius	0.001774	[L]
L_{box}	Box length	0.01	[L]
N_p	Number of Macroparticles	156055	-
N_r	Spatial grid points per dimension	256	-
dt	Timestep size	2.15623×10^{-13}	[T]
N_t	Number of timesteps	1000	-
q_e	Particle charge (normalized)	-1.0	[Q]
m_e	Particle mass (normalized)	1.0	[M]
f_c	Focusing strength	1.5	-
ϵ^{-1}	Inverse vacuum permittivity	3.182609×10^{9}	$\frac{[\mathrm{L}]^3[\mathrm{M}]}{[\mathrm{T}]^2[\mathrm{Q}]^2}$
Γ	Prefactor Rosenbluth potentials	8.0604×10^{18}	$[\mathrm{L}]^6/[\mathrm{T}]^4$
N_v	Velocity grid points per dimension	64	-
v_{max}	Maximum size of velocity quadrant	5.0×10^{7}	[L]/[T]

A.2 P^{3}M Reference Implementation

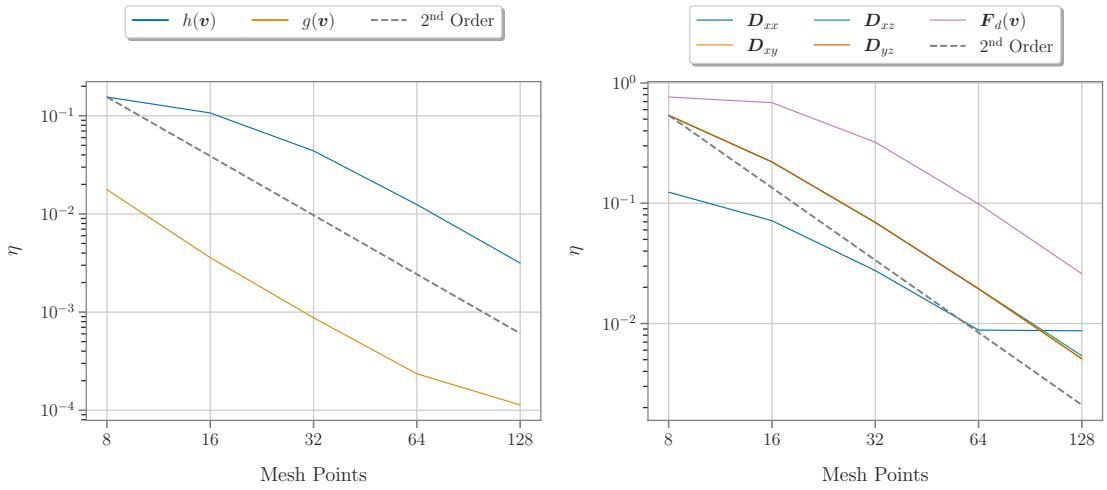
Table A.3: Default parameters of the DIH solver by Ulmer.

Parameter	Description	Default value	Unit
r_{beam}	Beam radius	0.001774	[L]
L_{box}	Box length	0.01	[L]
N_p	Number of Macroparticles	156055	-
N_r	Spatial grid points per dimension	256	-
dt	Timestep size	2.15623×10^{-13}	[T]
N_t	Number of timesteps	1000	-
q_e	Particle charge (normalized)	-1.0	-
m_e	Particle mass (normalized)	1.0	-
f_c	Focusing strength	1.5	-
r_c	Cut-off radius for PP interactions	3.125×10^{-4}	[L]
ε	regularization for PP interactions	0	[L]
α	Green's function splitting parameter	1.0×10^{6}	-

Appendix B

Convergence Study

B.1 Gaussian Testcase



(a) (b)

Figure B.1: Convergence study of Rosenbluth potentials (B.1a) solved with Hockney's algorithm and their corresponding collisional coefficients (B.1b) for a Gaussian velocity distribution following the same statistics as apparent in the DIH problem.

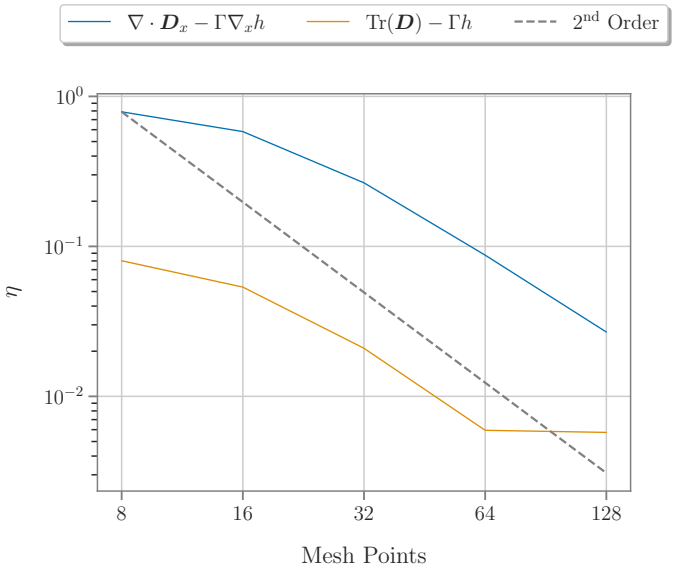


Figure B.2: Convergence study of the identities defined in Equation 2.19.

Appendix C

LDLT Decomposition

For our Langevin formulation of the Vlasov-Poisson-Fokker-Planck equation we need to factorize the 3x3 diffusion matrices $\underline{\underline{D}}$ with a Cholesky decomposition. Due to aforementioned reasons (see Section 3.3), this is most efficiently achieved with a LDLT decomposition. Our implementation is a straighforward unrolled implementation of this recursive algorithm. In the following listing of the implementation we omit invariants that require the matrix to be semi-positive definite ($\mathtt{D} \geq 0$ on line 23). They are present in the simulation code.

Listing 1 C++ function computing $\underline{\underline{Q}}$ from a given 3x3 matrix via the LDLT decomposition.

```
// Cholesky decomposition for semi-positive definite matrices
// Avoids sqrt of negative numbers by pivoting
// Computation is inplace
MatrixD_t LDLtCholesky3x3(const MatrixD_t &M) {
  MatrixD_t Q;
  VectorD_t row_factors;

  // Compute first row multiplicators
  row_factors[0] = M[1][0] / M[0][0];
  row_factors[1] = M[2][0] / M[0][0];

  // Eliminate value at [1,0]
  M[1] = M[1] - row_factors[0] * M[0];

  // Eliminate value at [2,0]
  M[2] = M[2] - row_factors[1] * M[0];

  // Eliminate value at [2,1]
  row_factors[2] = M[2][1] / M[1][1];
  M[2] = M[2] - row_factors[2] * M[1];

  // Check that the resulting matrix is semi-positive definite
  VectorD_t D = {M[0][0], M[1][1], M[2][2]};

  // Compute Q = sqrt(D) * L^T
  // Where D is diag(M) and `row-factors` are the lower triangular values of L^T
  // Loop is unrolled as we only ever do this for 3x3 Matrices
  Q[0][0] = Kokkos::sqrt(M[0][0]);
  Q[1][0] = row_factors[0] * Kokkos::sqrt(M[1][1]);
  Q[1][1] = Kokkos::sqrt(M[1][1]);
  Q[2][0] = row_factors[1] * Kokkos::sqrt(M[2][2]);
  Q[2][1] = row_factors[2] * Kokkos::sqrt(M[2][2]);
  Q[2][2] = Kokkos::sqrt(M[2][2]);

  return Q;
}
```

Appendix D

Supplementary Material of the DIH Experiments

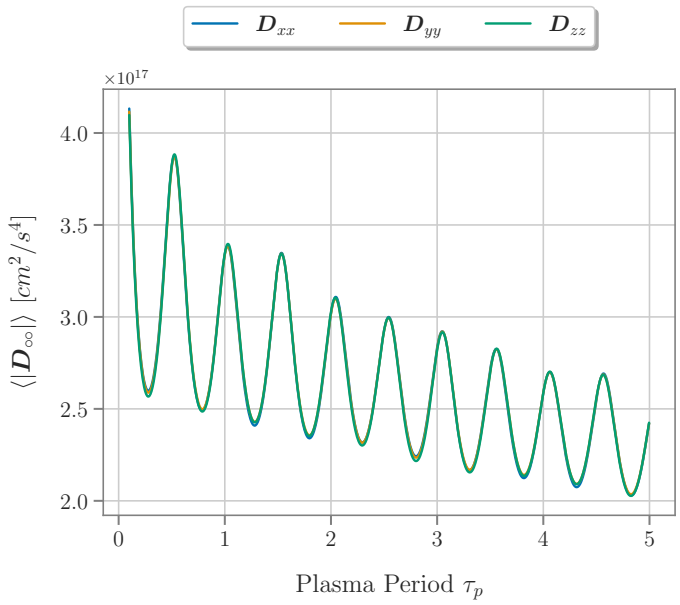


Figure D.1: Average absolute value of the diagonal diffusion coefficients over 5 plasma periods.

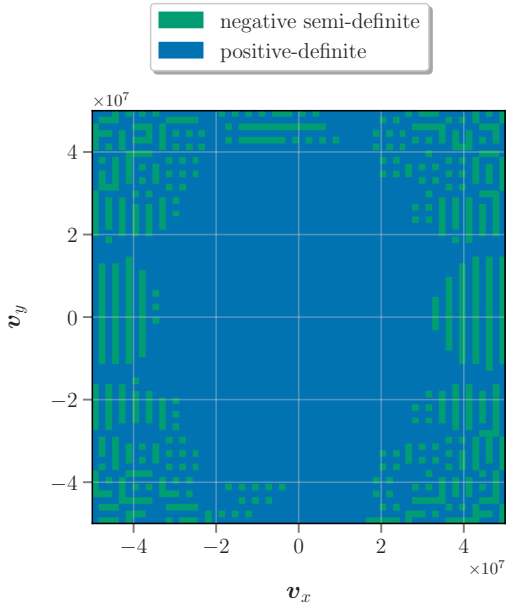


Figure D.2: Slice through the diffusion matrix field computed with the Hessian operator with spectral accuracy at $\boldsymbol{v}_z = 0$ and $t = 4\tau_p$ indicating the matrix property for each entry.

Appendix E

Chainable Differential Operators

Efficient and versatile stencil operators are at the heart of numerical simulation code. Thus, it does not come as a surprise that there exist a wealth of stencil libraries for almost any existing computer language (i.e. [45], [46] or [47]).

For our work we relied on a FD implementation of a Hessian operator (Eq. 2.3) in IPPL [38] for the diffusion coefficient computation. It implements a compact 2^{nd} order centered stencil. Unfortunately this comes with the drawback that the computed Hessian matrices along either system boundary cannot provide an accurate result, as they access values at the ghost-cells that have no physical meaning (due to our decision to model the velocity space with open boundary conditions). We want to create a flexible interface for defining differential operators by a combination of FD stencils (i.e. forward-, backward difference) which avoids code duplication as much as possible.

A differential operator can consist of one or multiple mathematically chained operators. We define a multi-index $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_n)$ and its cardinality as $|\alpha| = \sum_{i=1}^{n} \alpha_i$, such that we are able to express a chained operator

$$\mathcal{G}^{\alpha} = \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \cdots \partial x_n^{\alpha_n}}. \tag{E.1}$$

For simplicity we set $n = 3$, as all our simulations are carried out in 3 dimensions and set the requirement that the user can choose a specific stencil type for either dimension.

As an example we would like to create a 2^{nd} order accurate operator for the first derivative that does not access values outside the system boundaries on the lower x-axis. A potential candidate stencil defining a 1D forward FD scheme looks as follows:

$$\frac{\partial}{\partial x} f(x_k) = \frac{-3x_{k-1} + 4x_k - 2x_{k+1}}{2h_x}. \tag{E.2}$$

Template metaprogramming provides us with a powerful tool to construct and chain such operators at compile time. At run time the user can then apply the operator on a certain index of the field of interest to emulate his custom differential operator.

We continue to give the reader an insight into how we implemented this. For brevity, we omit some details regarding base classes and utility functions. The full code is available on [44]. The 1D stencil of Eq. E.2 is implemented in C++ as shown in Listing 2.

This or other stencils can then be encapsulated as an operator which either call a field at a certain index `(i,j,k)` or another `Callable` (i.e. a subsequent operator, exhibiting the same signature). This behavior is exemplified in Listing 3.

Listing 2 Generic first order centered FD stencil in 1D.

```
enum DiffType { Centered, Forward, Backward, CenteredDeriv2 };
enum OpDim { X, Y, Z };

template <OpDim D, typename T, class Callable>
inline T centered_stencil(const T &hInv, const Callable &F, size_type i,
                          size_type j, size_type k) {
  return 0.5 * hInv *
         (-shiftedIdxApply<D>(F, -1, i, j, k) +
          shiftedIdxApply<D>(F, 1, i, j, k));
}
```

Listing 3 Stencil operator calling a specific stencil of type `enum DiffType Diff` along `OpDim D`.

```
template <OpDim D, unsigned Dim, typename T, DiffType Diff, class Callable>
class DiffOpChain : public BaseDiffOp<D, Dim, T, Diff, Callable> {
public:
  typedef T value_type;
  typedef typename Field_t<Dim>::view_type FView_t;

  DiffOpChain(const FView_t &view, Vector_t<Dim> hInvVector)
      : BaseDiffOp<D, Dim, T, Diff, Callable>(view, hInvVector),
        leftOp_m(view, this->hInvVector_m) {}

  // Specialization to call the stencil operator on the left operator
  inline T operator()(size_type i, size_type j, size_type k) const {
    return this->template stencilOp(leftOp_m, i, j, k);
  }

private:
  // Need additional callable which might contain other operators
  const Callable leftOp_m;
};
```

Following this idea, it is a straightforward endeavor to construct more sophisticated operators that are based on multiple such chained constructs. A simple pseudo-code example computing $\frac{\partial^2}{\partial x \partial y} f(x, y)$ in 2 dimensions is shown in Listing 4. We conclude our excursion by constructing a Hessian operator given our stencil formulation, hence allowing the user to choose individual stencils along either dimension. In Figure E.1 we show that this operator exhibits the correct order of convergence (averaged over all matrix entries) for a selection of implemented stencils.

Listing 4 Pseudo-code for a chained operator (equivalent to $\frac{\partial^2}{\partial x \partial y} f(x, y)$).

```
constexpr int Dim = 2;

typedef double T;
// Inverse mesh-spacing
ippl::Vector<Dim, T> hInv = {40.0, 40.0};

// Field of type double and size [100]^2
Field<Dim, T> field(100, 100, 1.0 / hInv);

// Define the stencils applied along the x and y dimension
DiffType DiffX = DiffType::Forward;
DiffType DiffY = DiffType::Backward;

// Operator that is applied first
typedef DiffOpChain<OpDim::Y, Dim, T, DiffY, FView_t> firstOperator;
// Operator that is applied after the first
DiffOpChain<OpDim::X, Dim, T, DiffX, firstOperator> diff_xy(field, hInv);

// Compute curvature at index (42,42)
double result = diff_xy(42, 42);
```

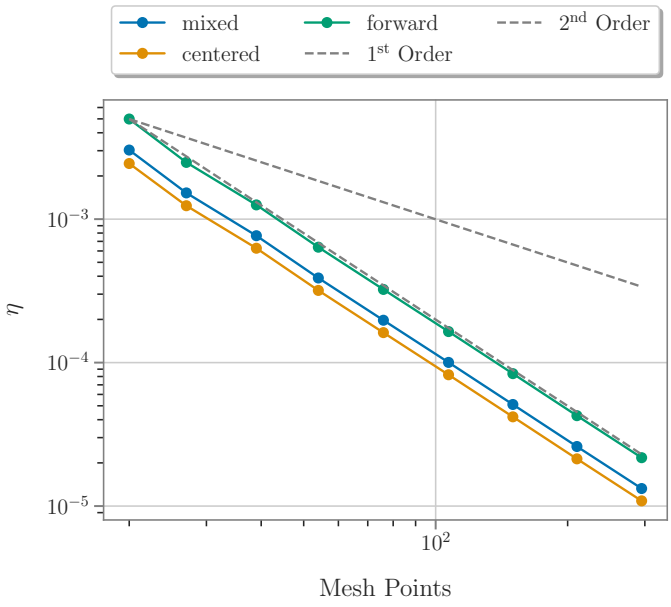


Figure E.1: Convergence study of our versatile Hessian operator. `mixed` signifies an operator which applies centered difference along the x-, forward difference along y- and backward difference along the z-dimension.

Acknowledgements

First of all, I would like to thank Andreas Adelmann and Sonali Mayani for guiding me through the entire duration of this thesis. Your valuable inputs as well as the numerous discussions we shared really helped me keeping my work on the right path, while also giving me the perseverance to press on in times I felt stuck.

Next, I would like to thank Sriramkrishnan Muralikrishnan, Matthias Frey and Antoine Cerfon for their regular inputs and sharing with me their wast experience with PIC codes and plasma physics. I greatly appreciated you taking the time for all these discussions and feedback.

Not to forget, I would like to thank all the AMAS group members I was able to spend many lunches and coffee breaks with. It was always a much welcomed change from working on the thesis.

Lastly, I am greatly indebted to my family for giving me sustained support during the course of my studies.

Bibliography

[1] T. J. M. Boyd and J. J. Sanderson, *The Physics of Plasmas*, 2003, ch. 8.4 Fokker-Planck Equation.

[2] B. Ulmer, "The p3m model on emerging computer architectures with application to microbunching," Master's thesis, ETH Zürich, 2016.

[3] S. Chandrasekhar, "Stochastic problems in physics and astronomy," *Rev. Mod. Phys.*, vol. 15, pp. 1–89, Jan 1943. [Online]. Available: https://link.aps.org/doi/10.1103/RevModPhys.15.1

[4] M. N. Rosenbluth, W. M. MacDonald, and D. L. Judd, "Fokker-planck equation for an inverse-square force," *Phys. Rev.*, vol. 107, pp. 1–6, Jul 1957. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRev.107.1

[5] W. M. Manheimer, M. Lampe, and G. Joyce, "Langevin representation of coulomb collisions in pic simulations," *Journal of Computational Physics*, vol. 138, no. 2, pp. 563–584, 1997.

[6] M. G. CADJAN and M. F. IVANOV, "Langevin approach to plasma kinetics with coulomb collisions," *Journal of Plasma Physics*, vol. 61, no. 1, p. 89–106, 1999.

[7] M. E. Jones, D. S. Lemons, R. J. Mason, V. A. Thomas, and D. Winske, "A grid-based coulomb collision model for pic codes," *Journal of Computational Physics*, vol. 123, no. 1, pp. 169–181, 1996. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999196900145

[8] L. Stoel, "The numerical solution of the vlasov-poisson-fokker-planck equation in the context of accelerator physics," Master's thesis, Utrecht University, 2015.

[9] F. H. Harlow, "The particle-in-cell computing method for fluid dynamics," *Methods Comput. Phys.*, vol. 3, pp. 319–343, 1964.

[10] A. Thomas, M. Tzoufras, A. Robinson, R. J. Kingham, C. Ridgers, M. Sherlock, and A. Bell, "A review of vlasov–fokker–planck numerical modeling of inertial confinement fusion plasma," *Journal of Computational Physics*, vol. 231, no. 3, pp. 1051–1079, 2012.

[11] D. R. Nicholson, *Introduction to Plasma Theory*. New York: Wiley, 1983.

[12] T. J. M. Boyd and J. J. Sanderson, *The Physics of Plasmas*. Cambridge University Press, 2003.

[13] A. H. Sørensen, "Intrabeam scattering," in *Frontiers of Particle Beams: Intensity Limitations*, M. Dienes, M. Month, and S. Turner, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 467–487.

[14] J. D. Callen, "Plasma kinetic theory," 2018, accessed: 2023-01-29.

[15] D. R. Nicholson, *Introduction to Plasma Theory*, 1983, ch. Appendix B.

[16] A. Fannjiang and L. Wołowski, "Noise induced dissipation in lebesgue-measure preserving maps on d-dimensional torus," *Journal of Statistical Physics*, vol. 113, pp. 335–378, 2003.

[17] J. D. Callen, *Plasma Kinetic Theory*, 2018, ch. Coulomb Collision Operator, accessed: 2023-01-29.

[18] ——, *Plasma Kinetic Theory*, 2018, ch. 2.2.3 Analogy with Electrostatics, accessed: 2023-01-29.

[19] A. Hakim, "The fokker-planck collision operator in gkeyll," published as a note (year unknown). Correctness FPO verified. [Online]. Available: https://ammar-hakim.org/sj/_static/files/fpo-notes.pdf

[20] T. J. M. Boyd and J. J. Sanderson, *Collisional transport in plasma.* North-Holland Amsterdam, 1983, vol. 1, no. 147, ch. 8.4 Fokker-Planck Equation, p. 331.

[21] M. R. R. Tabar, *Equivalence of Langevin and Fokker–Planck Equations.* Cham: Springer International Publishing, 2019, pp. 61–68. [Online]. Available: https://doi.org/10.1007/978-3-030-18472-8_7

[22] J. M. Maxson, I. V. Bazarov, W. Wan, H. A. Padmore, and C. E. Coleman-Smith, "Fundamental photoemission brightness limit from disorder induced heating," *New Journal of Physics*, vol. 15, no. 10, p. 103024, oct 2013. [Online]. Available: https://dx.doi.org/10.1088/1367-2630/15/10/103024

[23] E. Prat, P. Craievich, P. Dijkstal, S. Di Mitri, E. Ferrari, T. G. Lucas, A. Malyzhenkov, G. Perosa, S. Reiche, and T. Schietinger, "Energy spread blowup by intrabeam scattering and microbunching at the swissfel injector," *Physical Review Accelerators and Beams*, vol. 25, no. 10, p. 104401, 2022.

[24] K. Floettmann, "Some basic features of the beam emittance," *Phys. Rev. ST Accel. Beams*, vol. 6, p. 034202, Mar 2003. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevSTAB.6.034202

[25] F. Vico, L. Greengard, and M. Ferrando, "Fast convolution with free-space green's functions," *Journal of Computational Physics*, vol. 323, pp. 191–203, 2016.

[26] R. Hockney and J. Eastwood, *Computer Simulation Using Particles.* CRC Press, Mar. 2021. [Online]. Available: https://doi.org/10.1201/9780367806934

[27] J. Zou, E. Kim, and A. J. Cerfon, "Fft-based free space poisson solvers: why vico-greengard-ferrando should replace hockney-eastwood," *arXiv preprint arXiv:2103.08531*, 2021.

[28] M. W. Evans, F. H. Harlow, and E. Bromberg, "The particle-in-cell method for hydrodynamic calculations," LOS ALAMOS NATIONAL LAB NM, Tech. Rep., 1957.

[29] R. Morse, "Multidimensional plasma simulation by the particle-in-cell method." Los Alamos Scientific Lab., N. Mex., Tech. Rep., 1970.

[30] J. M. Dawson, "Computer simulation of plasmas (part of the proceedings of the iau colloquium no. 10, held in cambridge, england, august 12-15, 1970.)," *Astrophysics and Space Science*, vol. 13, pp. 446–467, 1971.

[31] F. H. Harlow, "Pic (particle-in-cell) and its progeny," Los Alamos National Lab.(LANL), Los Alamos, NM (United States), Tech. Rep., 1987.

[32] R. Hockney and J. Eastwood, *Computer Simulation Using Particles*. CRC Press, 2021, ch. 2-2-3 Charge Assignment and Force Interpolation.

[33] S. G. Johnson, "Notes on fft-based differentiation," *MIT Applied Mathematics, Tech. Rep.*, 2011.

[34] H. Risken, *Fokker-Planck Equation*. Springer, 1984.

[35] A. Krishnamoorthy and D. Menon, "Matrix inversion using cholesky decomposition," 2013.

[36] J. Adam, A. Gourdin Serveniere, and A. Langdon, "Electron sub-cycling in particle simulation of plasma," *Journal of Computational Physics*, vol. 47, no. 2, pp. 229–244, 1982. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0021999182900766

[37] AMAS Group, "Ippl (independent parallel particle layer)." [Online]. Available: https://gitlab.psi.ch/OPAL/Libraries/ippl

[38] A. Adelmann, P. Calvo, M. Frey, A. Gsell, U. Locans, C. Metzger-Kraus, N. Neveu, C. Rogers, S. Russell, S. Sheehy, J. Snuverink, and D. Winklehner, "Opal a versatile tool for charged particle accelerator simulations," 2019.

[39] C. R. Trott, D. Lebrun-Grandie, D. Arndt, J. Ciesko, V. Dang, N. Ellingwood, R. Gayatri, E. Harvey, D. S. Hollman, D. Ibanez *et al.*, "Kokkos 4: Programming model extensions for the exascale era," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 4, pp. 805–817, 2021.

[40] C. Mitchell, J. Qiang *et al.*, "A parallel particle-particle, particle-mesh solver for studying coulomb collisions in the code impact-t," in *6th Int. Particle Accelerator Conf.(IPAC'15), Richmond, VA, USA , May 3-8, 2015*. JACOW, Geneva, Switzerland, 2015, pp. 593–595.

[41] J. Qiang, R. D. Ryne, and S. Habib, "Self-consistent langevin simulation of coulomb collisions in charged-particle beams," in *SC'00: Proceedings of the 2000 ACM/IEEE Conference on Supercomputing*. IEEE, 2000, pp. 27–27.

[42] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. n. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[43] T. M. Inc., "Matlab version: 9.13.0 (r2022b)," Natick, Massachusetts, United States, 2022. [Online]. Available: https://www.mathworks.com

[44] Claglüna, Tobia , "Student repository (psi)," 2023. [Online]. Available: https://gitlab.psi.ch/AMAS-students/claglu-msc

[45] J. Dahm, E. Davis, T. Wicky, M. Cheeseman, O. Elbert, R. George, J. J. McGibbon, L. Groner, E. Paredes, and O. Fuhrer, "Gt4py: Python tool for implementing finite-difference computations for weather and climate," in *101st American Meteorological Society Annual Meeting*. AMS, 2021.

[46] Opdenhövel, Jan-Oliver and Alt, Christoph , "Stencilstream." [Online]. Available: https://github.com/pc2/StencilStream

[47] M. Bianco and U. Varetto, "A generic library for stencil computations," *arXiv preprint arXiv:1207.1746*, 2012.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

The Langevin Approach to Discretize the Collision Operator

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Claglüna	Tobia

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
Zürich, 17.07.2023	

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.