Multilevel Vortex Particle Method for Aerodynamic Simulations

vorgelegt von
M. Sc.
Joseph Saverin

an der Fakultät V – Verkehrs- und Maschinensysteme
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
- Dr.-Ing. -

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender:	Prof. Dr.-Ing. Julien Weiss
Gutachter:	Prof. Dr.-Ing. Christian Oliver Paschereit
Gutachter:	Dr.-Ir. Arne van Garrel
Gutachter:	Prof. Dr.-Ing. Jens Nørkær Sørensen
Gutachter:	Dr.-Ing. Néstor Ramos García

Tag der wissenschaftlichen Aussprache : 5. November 2021

Berlin 2022

Abstract

The ability to make predictions about the flow in the wake of a lift-generating body has a range of important applications. One example is the wake of an aircraft, where this flow significantly affects the proximity with which a trailing aircraft can take off, cruise or land. Another application is the placement and operation of wind turbines in a wind farm, where wake effects can significantly impact the performance of the turbines and the loading they experience. Offshore wind energy technology is progressing rapidly and it is expected to play a significant role in the transition to a clean energy grid. In offshore wind, wake interactions potentially play an even larger role than in onshore cases.

The designing engineer must currently resort to the use of low-fidelity models in the treatment of wake physics due to the computational expense of using high-fidelity models. Medium-fidelity tools are a good compromise between both fidelities as they offer better resolution of the problem without a significant increase in computational cost.

The present work introduces a method which aims to bridge the gap between medium- and high-fidelity wake treatments. This method is based on the vortex particle method, which treats the flow field as a superposition of vorticity-carrying elements. The approach is inherently grid-free which reduces computational overhead, simplifies the problem setup and allows for higher order effects such as viscous and turbulent diffusion to be accounted for.

Direct evaluation of a particle problem with N elements has computational complexity of $\mathcal{O}(N^2)$. The present method uses the *multilevel multi-integration cluster* method, which reduces the complexity to $\mathcal{O}(N)$. Two solvers have been implemented which employ this method. The first uses a Green's function treatment and is more suitable for medium-fidelity investigations. The second is based on the Poisson equation and is more suitable for high-fidelity studies.

The solvers have been validated against analytical and numerical results from the literature and the reduction of the computational complexity to $\mathcal{O}(N)$ is demonstrated. This work includes numerous simulated cases to demonstrate the range of application of the method: the wake behind an elliptic airfoil, the four-vortex wake system of an aircraft, the helical wake system of a wind turbine and the modes of instability of a helical vortex.

Zusammenfassung

Die Vorhersage der Strömungsverhältnisse im Nachlauf von Auftrieb erzeugenden Körpern ist in vielen Anwendungsbereichen von hoher Relevanz. Eine Anwendung betrifft beispielweise den Nachlauf von Flugzeugen in der Luftfahrt, bei denen die Nachlaufwirkung maßgeblich den Minimalabstand zwischen Flugzeugen bei Start und Landung sowie im Reiseflug bestimmt. Eine andere Anwendung betrifft die Windkraft. Bei der Platzierung und beim Betrieb von Windkraftanlagen in einem Windpark kann die Interaktion mit dem Nachlauf von stromauf platzierten Anlagen einen signifikanten Einfluss auf Wechsel- und Dauerlasten haben. Da die Windenergiebranche sich zunehmend auf Offshore-Anlagen konzentriert, bei denen die Nachlaufinteraktion eine noch größere Rolle als bei Onshore-Anlagen spielen kann, wird in Zukunft die Untersuchung von Nachlaufinteraktionen weiter an Bedeutung gewinnen.

Numerische Methoden höherer Ordnung können den Nachlauf zufriedenstellend abbilden, benötigen jedoch einen hohen Rechenaufwand. Daher werden zur Auslegung von aerodynamischen Komponenten aktuell hauptsächlich Methoden niedriger Ordnung angewendet, die allerdings wichtige physikalische Phänomene vernachlässigen. Diese Arbeit präsentiert eine Methode, die einen guten Kompromiss zwischen Rechenaufwand und realistischer Abbildung der wichtigen Phänomene schafft. Diese Methode basiert auf der Wirbelteilchen-Methode (vortex particle method), die das Strömungsfeld als Überlagerung von Wirbelelementen modelliert. Im Gegensatz zu etablierten numerischen Methoden wird hierbei grundsätzlich kein Berechnungsgitter benötigt, was zu einer Senkung des Rechenaufwands, Vereinfachung der Anwendung sowie Inklusion physikalischer Effekte höherer Ordnung wie molekularer und turbulenter Diffusion führt.

Der Rechenaufwand für die direkte Berechnung eines Systems aus N Wirbelteilchen ist $\mathcal{O}(N^2)$. Durch die Anwendung der *multilevel multi-integration cluster*-Methode kann der Rechenaufwand auf $\mathcal{O}(N)$ reduziert werden. In dieser Arbeit wurden zwei Strömungslöser implementiert. Ein Strömungslöser basiert auf Greenschen Funktionen und eignet sich für Simulationen mittlerer Ordnung. Der zweite Strömungslöser basiert auf der Lösung der Poisson-Gleichung und eignet sich für Simulationen höherer Ordnung.

Die Strömungslöser wurden anhand von analytischen und numerischen Ergebnissen aus der Literatur validiert und der optimal mögliche Rechenaufwand von $\mathcal{O}(N)$ realisiert. Anhand der folgenden ausgewählten Beispiele wurde

der breite Anwendungsbereich der Methode demonstriert: Der Nachlauf eines elliptischen Tragflügelprofils, das Vier-Wirbel-System im Nachlauf eines Flugzeugs, das helikale Nachlaufsystem einer dreiblättrigen Windkraftanlage sowie die instabilen Moden eines helikalen Wirbelfadens.

Acknowledgement

The first person I have to thank is my father, Peter Saverin. If you hadn't have shared your enthusiasm for the natural world I never would have gone down the path of becoming an engineer. Whether it was helping me fix up the old bush car or staring up at the stars, your enthusiasm for understanding how things work trickled directly down to me.

I certainly wouldn't be where I am today without a long list of excellent teachers. Thank you Mr. Paymon for being my first motivated math teacher, you helped lay the foundations for what would later come. In undergrad one man greatly altered my academic trajectory. Thank you Prof. Mike Pemberton infinitely many times for showing me not only how beautiful mathematics truly can be, but also for illustrating the value of communicating your enthusiasm for a topic to a student. Thank you to Prof. Stefan Frank at the HTW for taking an aussie under your wing and also for trusting me to take over the teaching of your course. It helped me realise just how much I love fluid dynamics.

Thanks to Dr. Georgios Pechlivanoglou for believing in me and being a great role model. Thanks to Prof. Paschereit for taking me on as a PhD student at the TU Berlin and for the years of joy teaching Strömungslehre together. Many thanks to Dr.-Ing. Navid Nayeri for giving me the freedom to work on my own topics as I saw fit, I could not imagine a better supervisor. Thanks to everybody in the institute. I hit the jackpot having so many funny, smart and interesting colleagues. Finally, a million thanks to Arne van Garrel. You have helped motivate me throughout the last few years and could always answer any question I had. Your consistently high-quality and thorough work is a goal to aspire to.

On a personal note, thank you to Marta for putting up with my constant conversations about my PhD topic. Thanks to my boys Jamie and Basti, my brothers from another mother. Last but not least, thank you my darling Elli for supporting me in every possible way for the last two years and for your patience while I spent many evenings typing this manuscript.

Contents

Nomenclature

Dimensions

$[\,\mathrm{m}\,]$	meter	length
$[\,\mathrm{s}\,]$	second	time
$[\,\mathrm{rad}\,]$	radian	angle
$[\,-\,]$	dimensionless	
$[\,\cdots\,]$	dimension from context	

Acronyms

BC	Boundary condition
BS	Biot-Savart
CFD	Computational fluid dynamics
CPU	Central processing unit
FD	Finite difference
FFT	Fast Fourier transform
FP	Fast Poisson
GPU	Graphical processing unit
HOA	High-order algebraic
HV	Hyperviscosity
JL	James-Lackner
LOA	Low-order algebraic
LES	Large eddy simulation
MLMIC	Multi-Level Multi-Integration Cluster
NS	Navier-Stokes
PSE	Particle strength exchange
RANS	Reynolds-averaged Navier-Stokes
Re	Reynolds number
RPM	Revolutions per minute
RVM	Regularised variational multiscale
SF	Stream function
URANS	Unsteady Reynolds-averaged Navier-Stokes
VP	Vortex Particle
VPM	Vortex Particle Method
2D	Two-Dimensional
3D	Three-Dimensional

Roman symbols

$\mathcal{B}$	$[-]$	Box region
$\mathcal{G}$	$[\cdots]$	Green's function
H_b	$[\mathrm{m}]$	base box size
i, j, k	$[-]$	Cartesian indices
$\mathcal{I}$	$[\cdots]$	Influence
$\mathcal{K}$	$[\cdots]$	kernel function
N	$[-]$	Particle count
P	$[-]$	Polynomial order
Re	$[-]$	Reynolds number
r	$[\mathrm{m}]$	Euclidean distance
t	$[\mathrm{s}]$	time
u	$[\mathrm{ms}^{-1}]$	velocity
x, y, z	$[\mathrm{m}]$	Cartesian coordinates

Greek symbols

α	$[\mathrm{m}^2\mathrm{s}^{-1}]$	Circulation
δ	$[\cdots]$	Dirac delta function
Δ	$[-]$	Difference
ϵ	$[\cdots]$	Relative error
Γ	$[\mathrm{m}^2\mathrm{s}^{-1}]$	Vortex strength
κ	$[-]$	Magnitude filtering factor
ν	$[\mathrm{m}^2\mathrm{s}^{-1}]$	kinematic viscosity
ω	$[\mathrm{s}^{-1}]$	Vorticity
Ψ	$[\cdots]$	Stokes' stream function
ψ	$[\mathrm{m}^2\mathrm{s}^{-1}]$	Stream function
ρ	$[-]$	Normalised distance (r/σ)
σ	$[\mathrm{m}]$	Core size
τ	$[-]$	Normalised time
θ	$[\mathrm{rad}]$	Rotation angle

Vectors, matrices, and tensors

$\vec{\alpha}$	$[\,\mathrm{m^2 s^{-1}}\,]$	Circulation vector
$\vec{\gamma}$	$[\,\mathrm{m^2 s^{-1}}\,]$	Vortex strength
$\overline{\overline{I}}$	$[\,-\,]$	Interpolation matrix
$\vec{\Omega}$	$[\,\mathrm{rads^{-1}}\,]$	Angular velocity vector
$\vec{\omega}_v$	$[\,\mathrm{s^{-1}}\,]$	Vorticity
$\vec{\psi}$	$[\,\mathrm{m^2 s^{-1}}\,]$	Vector potential (stream function)
$\vec{r}$	$[\,\mathrm{m}\,]$	Displacement vector
$\vec{u}$	$[\,\mathrm{ms^{-1}}\,]$	Velocity vector
$\vec{x}$	$[\,\mathrm{m}\,]$	Probe point position
$\vec{y}$	$[\,\mathrm{m}\,]$	Source point position

Subscripts and superscripts

$(\,)_{\mathrm{char}}$	Characteristic value
$(\,)_l$	Box level
$(\,)_{\mathrm{ff}}$	Far field contribution
$(\,)_{\mathrm{nf}}$	Near field contribution
$(\,)_x$	component in x-direction
$(\,)_y$	component in y-direction
$(\,)_z$	component in z-direction

Chapter 1 Introduction

The field of fluid dynamics contains a plethora of fascinating and puzzling phenomena. The application of mathematical analysis, in particular the field of continuum mechanics, bestows upon the avid investigator a powerful tool which allows great insight into the underlying processes. In some cases it is possible to derive analytical or closed-form solutions to certain flow problems [1]. In the majority of cases, however, analytical solutions are difficult to obtain due to the complex nature of the differential equations to be solved. Since the advent of modern computers, the advancements made in the field of numerical analysis have allowed engineers to gain insight into such phenomena by making use of numerical approximations to the equations of fluid motion, a field referred to commonly as computational fluid dynamics (CFD) [2].

Figure 1.1: Left: The wake of a Boeing 747 [3]. Right: The wake behind a row of wind turbines in the Horns Rev offshore wind farm [4].

The main application case for the work presented here is the simulation of the wake generated by lifting bodies, such as the blades of a wind turbine or wings of an aircraft– see Fig. 1.1. Within a design environment, when selecting simulation tools one often resorts to the use of low-order models in order to achieve quick solution turnaround times. These models often neglect important physics of the problem under investigation. Alternatively, one may employ high-fidelity simulation. This accounts for higher order physics, however can often require unrealistic computational resources. The work in this thesis aims to close this gap by developing a calculation method which allows for the simulation of complex flows without the necessity to

resort to high performance computing (HPC). The motivation for the choice of modelling method is outlined in the proceeding sections, followed by the main application cases. Finally an outline of the thesis is provided.

1.1 Objectives and Modelling Choices

There exists numerous methodologies for simulating a fluid numerically depending on the available resources and desired application. It is hence instructive to first clearly define the objectives prior to reviewing different modelling options.

1.1.1 Objectives

The aim of the work presented here is to develop a flow solver which is capable of simulating unbounded incompressible flows of a Newtonian fluid in the wake a lifting body. The main application case is the wake of a horizontal axis wind turbine. This however can be considered a specialised case of the more general lifting body problem. To increase the range of applicability of the solver the more general problem should be handled.

Modelling objectives The solver should be capable of capturing dominant physical phenomena at numerous ranges of fidelity. This motivates the modelling choices in the next section. The following objectives are specified:

- Wake flows resolved both spatially and temporally
- Wall-bounded flows are **not** considered
- Influence of viscous diffusion captured
- Influence of turbulent diffusion captured
- Adaptability of solver depending on desired resolution
- Seamless extension to multiple bodies / turbines

Performance objectives The solver should be deployable in numerous environments.

- The expense should scale optimally for a given problem size
- Parallel processing on a central processing unit (CPU)
- Parallel processing on a graphical processing unit (GPU)

- Applicability and scalability for future applications with high performance computing (HPC)

The target user is the practicing engineer wishing to carry out simulations at a range of fidelities. As such, the solver should be optimised for a single CPU-GPU combination environment.

1.1.2 Modelling Choices

It is necessary at this stage to carry out a review of the range of modelling choices available. This allows for an informed decision as to which method is best applied to achieve the objectives set down in Section 1.1.

General problem statement The motion of an incompressible Newtonian fluid can be handled numerically by solving two differential equations for the flow field: the continuity equation and the Navier-Stokes (NS) equation (see Chapter 2). Treatment of these equations can be broadly classed into two categories: Eulerian methods or Lagrangian methods. For a given vector quantity of interest $\vec{a}$ in a flow field with a local velocity of $\vec{u}$, the substantial derivative of $\vec{a}$ is given by [5]:

$$\frac{d\vec{a}}{dt} = \underbrace{\frac{\partial \vec{a}}{\partial t}}_{\text{Local}} + \underbrace{(\vec{u}\cdot\nabla)\vec{a}}_{\text{Convective}}, \tag{1.1}$$

where the two terms on the right hand side are the **local** or **temporal** derivative, and the **convective** derivative. As these differ greatly in their implementation, they will be discussed separately.

Eulerian (grid-based) methods The majority of current CFD solvers make use of an Eulerian method, where the flow quantities are resolved on a fixed grid. As these coordinates do not translate with the local flow, the convective component must be explicitly accounted for in the equations of motion. A range of structured or unstructured grid options are possible. The differential equations describing the fluid motion are discretely approximated using finite differences (FD) for flow quantities (mass, momentum etc.) either between grid nodes in the finite-difference method (FDM) or as grid volume or *cell* fluxes in the finite-volume method (FVM) [6]. The use of basis functions over these elements to represent solutions gives rise to other approaches, such as the discontinuous Galerkin method [7] and spectral methods [8]. In regions where strong gradients must

be resolved (e.g. near a lifting body) the grid resolution must necessarily be increased to resolve gradients. The presence of moving bodies such as lifting surfaces furthermore requires the use of overset or sliding grids, increasing complexity. If resolution of the flow near walls does not need to be resolved (for example when investigating the wake), computational savings can be made with immersed source treatments of lifting bodies such as the well-established actuator disc or actuator line methods [9, 10].

As per the modelling objectives, focus is now restricted to simulating unbounded flows. This requires the choice of a computational domain which is sufficiently large for the problem. The extent of the domain has to be chosen large enough so that the representative boundary conditions (BC) on the domain can be specified and do not interfere with the developing flow field. This BC unfortunately often necessitates very large grid domains to ensure that the far field BCs are met. Grid coarsening near boundaries is often applied in order to reduce computational expense. Despite this, large regions of the domain must be meshed, despite the fact that the solution in these regions is not sought.

The numerical approximation of the flow quantities is a Taylor series usually carried out to a given (often 2^{nd}) order, the effect of neglecting higher order terms introduces numerical diffusion into the solution, this plays a significant role in wake simulations [11]. Velocity gradients alone are

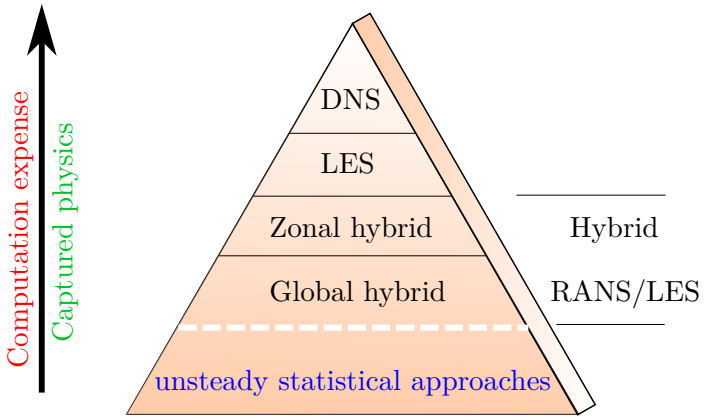

Figure 1.2: Classification of unsteady approaches to turbulence modelling [12].

sufficient for resolving viscous diffusion. Resolution of turbulent diffusion requires a suitable turbulence model, a broad field in which great progress has been made [13]. To limit the scope of the review and align it with the objectives stated above, attention is now restricted to unsteady models. An overview of modelling approaches is given in Fig. 1.2. Direct Numerical

Simulation (DNS) resolves the relative time and length scales in a turbulent flow [14]. For a flow with Reynolds number Re the computational complexity scales as Re^3 [15]. As wake flows generally have a high Re, this option is quickly deemed to be unfeasible.

Numerous methods exists for truncating the time and length scales which are resolved in order to reduce computational expense. A reduction in computational expense is achieved with large-eddy simulation (LES) where larger 3D unsteady turbulent motions are resolved and smaller-scale, more universal fluctuations are modelled [15, 16]. This utilises an appropriate spatial filter for small-scale motions [17] which are generally modelled with a sub-grid scale (SGS) model, the simplest being the one proposed by Smagorinski [18]. The least expensive turbulence modelling can be achieved by applying the Reynolds-Averaged Navier-Stokes (RANS) equations, where an averaging of the flow field is applied and the turbulent Reynolds' stress is modelled [15]. The target applications considered in this dissertation require temporal resolution of the flow field, making the unsteady analog URANS more favorable. In this case unsteadiness of the mean flow field is accounted for. Although additional turbulence closure models are required, grid and temporal resolutions can be significantly lower in URANS than LES, leading to computational savings. These models however are generally less accurate for unsteady flows due to the underlying averaging operations [19]. Numerous combinations of these two approaches are available: global hybrid models such the partially averaged Navier-Stokes equations [20] or zonal hybrid models, such as detached eddy simulation [19].

Lagrangian (grid-free) methods These methods do not require specification of a grid and the flow quantities are specified with discrete *elements* which are advected with the local fluid velocity, and hence within a moving reference frame [21]. The convective term therefore does not need to be explicitly calculated. Theoretically, these methods have zero numerical diffusion. Elements are created or destroyed as required to accurately describe the flow field, implying a spatial adaptivity of the method. There are two main types of Lagrangian flow modelling applied to fluid simulations, vortex methods and smoothed particle hydrodynamics.

Vortex methods (VM) express the vorticity field at a set of discrete points, each with a given position and orientation. These can effectively be considered as quadrature nodes of the field. Applying the Helmholtz decomposition, the velocity can be extracted directly from the vorticity field using the Poisson equation [22]. Every time step, the properties of the vortex elements are updated as follows: positions are updated via convection with the local

velocity and element strengths are updated with the curl of the NS equations- the vorticity transport equation (VTE) [5]. For incompressible fluids being acted upon by a conservative body force, the pressure term vanishes from the equation, simplifying modelling. Each vortex element influences the entire flow domain of interest, as such the solution of the Poisson equation, achieved by performing a convolution of the vorticity field, can lead to high computational costs.

In order to avoid singular solutions to the field equations, the vorticity description is supplemented with a regularisation or *smoothing* in order to make the model numerically applicable [23]. Regularisations are generally radially symmetric, implying that they are not suitable for applications with a strong preferential gradient. In fact, for wall-bounded flows the ratio between the number of grid nodes for a non-conforming and body-fitted mesh scales as $\mathrm{Re}^{3/2}$ for a 3D problem [24]. This makes this method impractical for resolution of e.g. lifting surfaces, particularly in high Re 3D scenarios. This however is inconsequential here as wall-bounded flows are not within the objectives set out. Numerous works have demonstrated the advantages of hybrid Eulerian-Lagrangian solvers which use an Eulerian approach for body-fitted meshes and Lagrangian solvers elsewhere [25, 26, 11]. The effect of viscous diffusion appears in the VTE as the Laplacian of the vorticity field. This is calculated either with an integral operator– the particle strength exchange (PSE) scheme [27]– or with FD on a local grid [28]. The application of LES to VM has been successful with numerous publications demonstrating the efficacy of this approach [29, 30, 25, 31, 32]. Here the filtering procedure is applied to the vorticity field, otherwise the approaches are similar to those used for Eulerian solution methods.

Within smoothed particle hydrodynamics (SPH), a discrete representation is again used, however the field is represented with pressure terms. This allows the introduction of an equation of state if desired [33]. The pressure field is updated by solving an appropriate pressure Poisson equation [34]. Unlike VM, each element has a finite influence radius and hence a limited interaction distance. This makes the method ideal for calculation on a GPU [35]. Viscous effects are captured with a direct SPH discretisation of the Laplacian of the velocity field, as with Eulerian methods. Although there are studies which suggest the applicability of SPH to turbulence modelling using LES [36], thus far few practical applications have been demonstrated.

Overview and summary A summary has been provided here of the advantages and disadvantages of both models, with consideration of the objectives outlined in Section 1.1. Based on these points, it can been seen

that Lagrangian vortex methods are the superior choice given the modelling objectives. The following sections expound in greater detail the relative literature and modelling choices relevant to this method.

Eulerian Methods	Lagrangian Methods
(+) Extensive literature	(+) VM extensive literature
(+) Comprehensive options for turbulence modelling	(+) VM & SPH: gridding unnecessary
(+) Variable fidelity possible	(+) Variable fidelity possible
(-) Numerical diffusion	(+) No numerical diffusion
(-) Complex gridding required	(+) Multiple bodies straightforward
(-) Large domain required: Very high expense	(-) SPH: Turbulence excluded
(-) Mesh complexity increase with multiple bodies	(-) VM: Convolutions expensive
	(-) Unsuitable for lifting bodies

Flow element choice For 2D problems there exist numerous practical methods of analysis. Discrete particles can be used to represent the vorticity field [37, 38, 39] and allow for the treatment of viscous effects [40]. The method of contour dynamics makes use of the material contours to follow regions or *patches* of circulation [41, 42, 43]. A third approach is to use particles of vorticity gradient as an alternative to contour dynamics [22]. As the focus here however is on fully 3D flows, the remaining options are either vortex filaments or 3D particles. Filaments by construction ensure that the vorticity field remains divergence free [22], however curvature of the filament plays a significant role in self-induction [44] and hence the filaments must be regularly updated to ensure that curvature is properly represented. Furthermore, the treatment of viscous diffusion can only be approximately modelled with simplified regularisations which approximately treat viscous core growth models [45, 46]. As stated in Winckelmans [22], these are best suited to inviscid flow problems and are hence unsuitable here. The 3D vortex particle method (VPM) however allows for the treatment of viscous interaction, adequate spatial representation of flow elements and furthermore enables the treatment of turbulent flows. The work here shall therefore focus on the use of 3D vortex particles.

Particle-particle and Particle-mesh methods The time-evolution of the particle set within VPM requires field data directly at the Lagrangian marker positions. Two approaches are commonly taken to resolve these field quantities. The first approach considers particle-particle interactions. In this case the influence of a particle on a given position is calculated using closed-form expressions for the velocity and shear fields. These expressions depend upon the choice of particle regularisation scheme. Three schemes can be applied for the calculation of the shear terms: the classic, transpose and mixed schemes [23]. The availability of closed-form expression facilitate simple implementation however choice of regularisation and gradient scheme can potentially greatly influence the evolution of the particle distribution on vorticity distribution, particle spacing and flow topology [47, 48]. This approach can be extended for viscous effects with PSE, provided suitable stability criteria are considered [40].

Particle-particle Methods	Particle-mesh Methods
(+) Direct expressions for influence	(+) Application of efficient Fast-Poisson solvers
(+) Viscous stresses easily resolved with PSE	(+) Simple calculation of high order gradients
(+) Field resolved exactly where required	(+) Grid-interactions easily expressed as template
(-) Restrictive stability constraints	(-) Large memory footprint
(-) High order gradients difficult to extract	(-) Mapping to/from particles required
	(-) Gradient calc. require grid *overlap* regions

The second approach applies particle-mesh interactions. In this case the source distribution of the Lagrangian particles is mapped to an underlying regular grid. The Poisson equation is solved on this grid and field quantities are extracted by applying a FD scheme. These are interpolated back to the particle position to calculate particle set evolution. Although this technically can be considered a grid-based method, the grids are regular and adaptive and therefore require no mesh preparation *per se*. In addition, particles are still convected in a Lagrangian sense. This is conceptually simpler than the particle-particle methods as complicated interaction expressions are avoided and near-field interactions are automatically handled by the Poisson solver. Gradients of any order of the flow field are accessible simply through

application of FD stencils, enabling for example straight-forward application of turbulence models. A simple overview of advantages and disadvantages of both approaches are summarised in the table above. Both of these methods present attractive features. Which approach is most suitable certainly depends on both flow properties and desired outcomes. Both methods will be explored in the work carried out in this thesis.

Optimisation of problem scaling Considering now the calculation of a given problem with VPM, the velocity field is necessary for specification of the advection velocity of the Lagrangian elements. This can be calculated by solving a Poisson equation (see Chapter 2) which leads to an expression for the velocity induced by each vortex or **source** particle. It shall be assumed that there are N_s **source** particles and it is desired to evaluate the velocity at N_p **probe** points. Without loss of generality, it shall be assumed the $N_s = N_p$ and the computational complexity is seen to scale as $\mathcal{O}(N^2)$: N evaluation points each being influenced by N source points. As the particle set grows in size, as e.g. with an evolving wake, this scaling becomes impractical. In order for the method to be applicable without the necessity to resort to HPC, methods of optimisation must be sought out.

For direct particle interactions there exists a range of techniques which optimise computational complexity. These all involve some form of spatial coarsening of the source distribution. For the proceeding discussion the word **kernel** refers conceptually to the function which describes the influence a unit source has on a given probe position. The properties of the kernel applied for the solution of the Poisson equation generally admit approximation for large separation distances. This was first carried out in the work of Barnes & Hut by applying it to the gravity kernel for astrophysical N-body problems [49]. Their method used an Octtree-style approach to describe the source distribution hierarchically, where source regions are subdivided until each source is effectively enclosed within an isolated region or cell.

The source points at each box level are approximated by an equivalent pseudo-source placed at the centroid of the particle in that box, this is visualised for an exemplary 2D case in Fig. 1.3. For the calculation of the influence at any probe position as required an integration over the whole field (all boxes) is carried out. If a given source box obeys a specified distance threshold, the influence of the pseudo-particle is calculated rather than the contribution of each particle contained within. The error of the approximation decreases (and computational expense increases) by increasing the distance threshold. This algorithm greatly reduces the cost of the influence of distance particles and the computation expense is reduced to

$\mathcal{O}(N \log N)$. For N very large $\log N$ behaves almost as a constant and this performance is in practice almost indiscernible from $\mathcal{O}(N)$. Conceptually, **some** form of influence calculation must occur for each source particle N, one can hence heuristically argue that the most optimum possible calculation has order $\mathcal{O}(N)$. The approximation of the source particles using a single

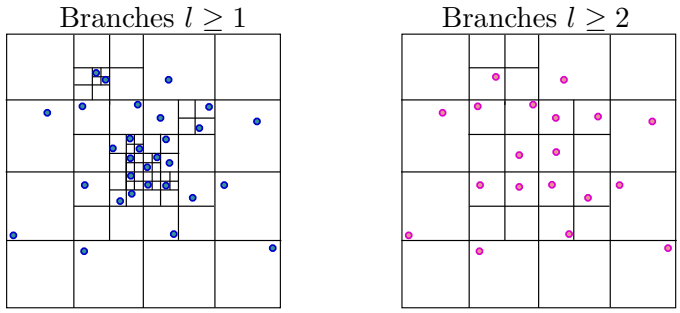

Figure 1.3: Octtree concept of Barnes-Hut algorithm. Left: Source distribution. Right: Source centroids of higher branch levels. Adapted from [49].

lumped mass treatment can be greatly improved by applying the concept of a multipole expansion. This is defined as a polynomial expansion for functions which depend only on two angular arguments, generally expressed in terms of spherical harmonics. This was carried out in the seminal work by Greengard [50], where the so-called Fast-Multipole Method (FMM) was first described. As with a Taylor series, the infinite expansion involves higher order derivatives and the error is controlled by specifying the order of the multipole expansion. It was shown that this method, when applied to 3D problems also reduces problem complexity to $\mathcal{O}(N \log N)$ [51]. This method has been found to be well suited to VPM [28, 37], however the application to the higher-order VPM interactions such as vortex stretching is not intuitive and presents challenges for implementation [52]. Other approaches to this problem have been suggested to simplify this approach [47].

In cases where the particle set can be mapped or approximated on a regular grid (this topic will be handled later), there exists numerous methods to optimise efficiency depending on the differential equation being investigated, for example the elliptic Poisson equation in this work. Multigrid methods [53] apply a range of grid resolutions to iteratively reduce the error of the solution by using restriction and interpolation operations between grid levels. High-frequency errors decay efficiently on finer grids and low-frequency errors on coarser grids, with the residual acting as a communication variable between grid levels [54]. This approach can reduce the computation complexity to $\mathcal{O}(N \log N)$, where in this case N refers to number of grid points. This has been implemented for general problems in the open-source library MudPack

[55]. For application to the solution of the Poisson problem specifically, the linear system on the grid can be expressed as an expansion of Fourier coefficients and solved using the Fast Fourier Transform (FFT), which has computational complexity $\mathcal{O}(N \log N)$. This class of solvers are referred to as Fast Poisson solvers (FP) and also have numerous implementations [56, 57].

A method recently developed by van Garrel [58], the multilevel multi-integration cluster (MLMIC) scheme, applies an adaptation of the Multi-Level Multi-Integration (MLMI) scheme of Brandt, Lubrecht & Venner [54, 59] to particle-like problems. In this scheme both the kernel function (source distribution) and influence are approximated using pseudo-source and probe nodes with an interpolating polynomial. Although, for a given accuracy, fewer expansion terms are required by FMM, the MLMIC scheme appears to be much more intuitive and amenable to optimisation. In [58] it was shown that this method reduces the computational complexity to the optimal $\mathcal{O}(N)$. The method furthermore is greatly generalisable, and can be applied to numerous physical problems involving fluid dynamics, astrophysics and electrostatics.

For the work applied here two solvers have been developed depending on the desired application case, both make use of the MLMIC scheme and the second makes use of the FP solver.

1.2 VPM: Wind Turbine Aerodynamics

The evolution and stability of the wake of a wind turbine is a complex and fascinating process [60, 61]. The clustering of turbines in a wind farms inevitably gives rise to wake interactions- i.e. wind turbines operating in either partial or full wake shadowing of upstream turbines- increasing unsteady and fatigue loads [62], which impacts the operational life of turbine components. It has also been shown to reduce farm energy yield by $5-10\%$ [63]. These factors are directly coupled to the Levelized Cost of Energy (LCOE) of operating a farm [64] and therefore influence investment in wind energy. There has been growing interest in floating offshore wind energy in the last decade due to the abundant wind energy resources in offshore farms. Here the aforementioned wake interaction effects have been shown to be even more prominent than in onshore scenarios as the turbines are exposed to stronger, steadier winds and stabler atmospheric stratification than comparable onshore environments. Under these conditions turbine wakes can remain stable and coherent for much greater distances (up to tens of kilometres) downstream of the turbine [65].

It is important here to distinguish between two wind turbine architectures: the much more commonly applied horizontal axis wind turbine (HAWT) and the vertical axis wind turbine (VAWT). The blade and wake aerodynamics in both cases are vastly different and VPM has been applied extensively to the simulation of both architectures.

VAWT wake aerodynamics This case poses additional modelling challenges as the blade interacts directly with the wake in the downstream rotation sector. As early as 1981, a 2D vortex model was applied to investigate the aerodynamics of VAWTs by Strickland [66]. This can also be applied to the unsteady wake behind an oscillating airfoil for unsteady 2D cases [67, 68]. More recently a much higher fidelity LES simulation was carried by Chatelain et al. [32] and was shown to capture in detail wake patterns and aerodynamics at multiple tip speed ratios λ– see Fig. 1.4. The case of offshore VAWT wakes was inspected in the work of Balty et al. [69], where simulations demonstrated that wake interaction leads to larger oscillations of downstream turbines and accelerated wake breakdown.

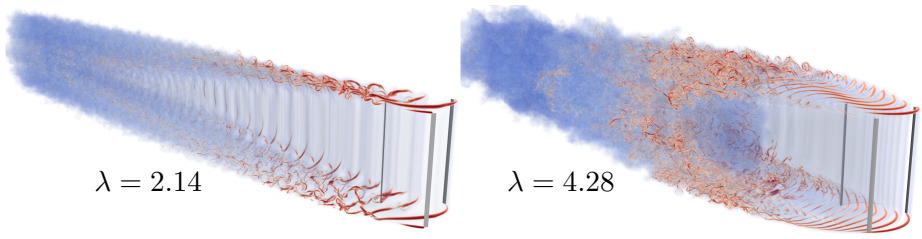

Figure 1.4: VPM-LES simulations of the wake of a VAWT, taken from [32].

HAWT wake aerodynamics Here it is observed that generally two approaches have been taken. In the first approach, the vortex filament is not spatially resolved. This filament-style treatment greatly reduces computational expense and is adequate for near-field analysis, e.g. single turbine performance as shown in the work by the author [70]. In the second approach the filament (and generally the vorticity distribution shed from the blade) is spatially resolved. This is commonly carried out with a hybrid Langrangian-Eulerian approach making use of a FP solver. Interaction between multiple turbine was simulated along with the effect of turbulent inflow description in the work of Chatelain et al. [31]. The method was applied to aeroelastic simulations with a VPM representation of turbulent inflow in the paper of Branlard et al. [71]. A similar approach was carried out for a hybrid filament-particle solver within the MIRAS solver in Ramos-

Garcia et al. [72]. The same model was applied to investigate the influence of the atmospheric boundary layer and ground effect in a follow-up paper [73]. An analysis of the underlying modes of symmetry of a helical wake, described analytically in Widnall [74], was investigated with VPM in Walther [75].

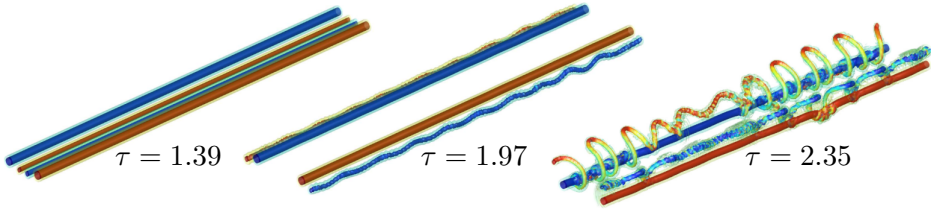

Figure 1.5: VPM simulation of medium-wavelength instability in a four-vortex aircraft wake system, taken from [76]. Normalised time shown.

1.3 VPM: Aircraft Wake Aerodynamics

As stated in the objectives, it is desired that the solver should generally be applicable to unbounded aerodynamic problems. For this reason the applicability of VPM to aircraft wake problems is briefly reviewed. The geometry of the aircraft wake problem is in general much simpler than that of a wind turbine. Persistent aircraft wakes give rise to problematic take-off and landing conditions for a trailing aircraft, motivating investigation into their evolution and excitation [77]. Here, two common wake geometries are usually investigated: the **two**-vortex system, where only tip vortex filaments (or more generally the merged tip-trailing vortex system) is inspected. Much as with the helical case, there exist well explored analytical results for the stabilities which arise in the two vortex system, namely the Crow (long-wavelength) [78] and the Widnall (short-wavelength) [79, 80] instabilities. These generally make use of an underlying assumption of periodicity in flight direction. The second configuration is a four-vortex system where a secondary inner vortex set is generated by inboard flap edges [81]. VPM was applied successfully to analyse these instabilities, determine optimum excitation parameters and investigate nonlinear behaviour in Winckelmans et al. [82]. Very high resolution investigations were carried out in Chatelain et al. [83] where a hitherto unseen resolution of the connection process was captured. Medium wavelength instabilities were also captured in the works of Cocle et al. [28], see Fig. 1.5.

1.4 Thesis Outline

The thesis has been structured as follows.

Chapter 2: The vortex particle method The fundamental assumptions of the flow field and the corresponding equations of motion are presented. A description of VPM is given. The solvers implemented in the work here are described: i) the particle-particle scheme based on Green's functions and ii) the particle-mesh scheme based on the solution of Poisson's equation. A description of both solution methods is given along with important modelling considerations when applying VPM.

Chapter 3: The multilevel integration method The scheme utilised here to achieve optimal problem scaling is described along with the application to VPM. The method is applied to a representative flow geometry and the ability of the scheme to accurately integrate the particle influence for both kernels of interest is validated.

Chapter 4: Flow solver implementation A description of the applied numerical scheme is given. This includes particle description, grid definition and parameters, and time integration schemes. The methodology for handling problems with symmetry and periodicity are described, along with individual parameters for both solvers.

Chapter 5: Flow solver validation and performance evaluation The flow solvers are verified against analytical and numerical results from the literature for a set of cases involving vortex rings. The ability of both solvers to capture a range of physical phenomena is demonstrated. The ability to simulate turbulent flow is demonstrated. The performance of the method is analysed and compared to expected behaviour.

Chapter 6: Flow solver application A set of flow cases suitable for application of the method are demonstrated. These include: i) a time and space evolving wake behind an elliptical airfoil, ii) a demonstration of instability growth in an infinite aircraft wake , iii) the wake of a wind turbine, and iv) the modes of instability of a helical wake.

Chapter 7: Conclusions and outlook A summary of the results is given along with an overview of the planned future work using the imple-

mented solver. Other applications of practical interest are also described.

Appendices Numerous validation cases are provided to supplement the results in Chapter 5, in additional to a 2D validation of the solver. The theory of the vortex ring is described as an addendum to the cases in the validation. The expressions for the Green's functions for a regularised vortex particle are given.

Chapter 2 The Vortex Particle Method

This chapter exposes the theory underlying the vortex particle method (VPM). Two methods are commonly applied when resolving the desired field quantities with the VPM.

- Green's method: This makes use of Green's functions to calculate direct *particle-particle* interactions- described in Section 2.3.
- Poisson method: This makes use of an intermediate grid to calculate *particle-mesh* interactions- described in Section 2.4.

Although the two methods resolve the same field quantities, their methodologies differs significantly, hence motivating separate exposition. The field equations underlying both methods are initially described in Section 2.1 along with relevant nomenclature. The discretisation of the vorticity field along with time evolution of the discrete particle set is detailed in Section 2.2. Following this important numerical considerations for application of the VPM along with criteria for well-resolved simulations are described in Section 2.5. Finally methods for the resolution of turbulent shear stresses within the VPM are described.

2.1 Field Equations

It assumed that the flow is incompressible and consisting entirely of a Newtonian fluid. Position is denoted with the vector $\vec{x}$ and time with the symbol t. An implicit assumption is hereafter made that all variables are a function of both space and time.

Velocity formulation The flow is described in terms of the velocity field $\vec{u}(\vec{x},t)$. Enforcing the conservation of mass leads to the continuity equation:

$$\nabla \cdot \vec{u} = 0\,. \tag{2.1}$$

Conservation of momentum for an incompressible Newtonian fluid leads to the Navier-Stokes (NS) equation:

$$\frac{d\vec{u}}{dt} = \frac{\partial \vec{u}}{\partial t} + (\vec{u}\cdot\nabla)\vec{u} = \vec{f} - \frac{1}{\rho}\nabla p + \nu\,\nabla^2\vec{u}\,, \tag{2.2}$$

where ν is the kinematic viscosity of the fluid, $\vec{f}$ is the body force and p is the fluid pressure. Although of great theoretical and practical interest, these forms of the field equations are not directly applied in the work here.

Velocity-vorticity formulation An alternative representation is achieved by taking the curl of the velocity field, the vorticity $\vec{\omega}(\vec{x},t) = \nabla \times \vec{u}$. By nature of its definition, it follows that the vorticity is divergence-free:

$$\nabla \cdot \vec{\omega} = 0\,. \tag{2.3}$$

Assuming that the body force is irrotational, taking the curl of the NS equations gives the vorticity transport equation (VTE):

$$\frac{d\vec{\omega}}{dt} = \frac{\partial \vec{\omega}}{\partial t} + (\vec{u}\cdot\nabla)\vec{\omega} = (\vec{\omega}\cdot\nabla)\vec{u} + \nu\,\nabla^2\vec{\omega}\,. \tag{2.4}$$

Helmholtz decomposition Helmholtz [84] demonstrated that a twice continuously differentiable vector field can be expressed as the sum of an irrotational scalar potential Φ and a divergence-free vector potential $\vec{\psi}$, here expressed for the velocity field:

$$\vec{u} = -\nabla\Phi + \nabla\times\vec{\psi} = \vec{U}_\infty + \nabla\times\vec{\psi}\,. \tag{2.5}$$

The vector potential $\vec{\psi}$ is generally referred to as the stream function. $\vec{U}_\infty$ represents the ambient freestream flow. Taking the curl of the above equation results in the Poisson equation:

$$\nabla^2\vec{\psi} = -\vec{\omega}\,, \tag{2.6}$$

which directly links vorticity to the stream function of the flow.

2.2 Vortex Particle Representation

In the vortex particle method, the spatial distribution of vorticity is defined as the sum of a set of vortex particles, each having vorticity $\vec{\omega}_p$ and occupying a volume dv_p, representing an incremental circulation $\vec{\alpha_p} = \vec{\omega}_p dv_p$:

$$\vec{\omega}(\vec{x}) \overset{\text{disc.}}{=} \sum_p \vec{\omega}_p\,\delta(\vec{x}-\vec{x}_p)\,, \tag{2.7}$$

where δ is the 3D Dirac delta function. The evolution of the particle set is treated in a Lagrangian sense, implying the particle positions and strengths are tracked, rather than being defined within a spatially constant grid.

Time evolution of particle position Recalling that the evolution equation for a material line element $\delta\vec{I}$ is given by [85]: $\frac{d}{dt}\delta\vec{I} = (\delta\vec{I}\cdot\nabla)\vec{u}$ and comparing with Eq. (2.4), it follows that vortex filaments move as material lines for inviscid flows. This is precisely the statement of the second theorem of Helmholtz [84]. For viscous flows however, vortex tubes do not necessarily retain their identity due to merging and breakdown. For a particle treatment, particle motion is specified purely with the kinematic definition:

$$\frac{d\vec{x}_p}{dt} = \vec{u}(\vec{x}_p)\,. \tag{2.8}$$

Time evolution of particle strength Inspecting the VTE in Eq. (2.4), the evolution of particle strength is seen to be composed of two contributions:

$$\frac{d\vec{\omega}_p}{dt} = \underbrace{(\vec{\omega}\cdot\nabla)\vec{u}}_{\text{Stretching}} + \underbrace{\nu\nabla^2\vec{\omega}}_{\text{Viscous diffusion}}\,. \tag{2.9}$$

The first term is a consequence of the divergence-free nature of the vorticity field and represents the change in vorticity which occurs when the vortex is sheared or strained. The second term is the change due to viscous diffusion. There are numerous approaches to calculating these terms. The two approaches used in the work here are described in the following sections.

2.3 VPM: Green's Method

When using the Green's function approach, one makes direct use of unbounded Green's functions [86] to calculate the velocity field. Closed-form expressions for the velocity and stretching fields induced by a particle are derived and these terms are used to calculate direct particle-particle interactions, this is visualised in Fig. 2.1.

2.3.1 Resolution of Field Quantities

The application of Green's function begins by initially describing the method used to retrieve the velocity field from the vorticity field. These quantities are then used to calculate higher order terms. These are described individually for clarity:

Velocity field The stream function can be calculated from the vorticity field using Eq. (2.6) by applying the Green's function for the Laplacian in

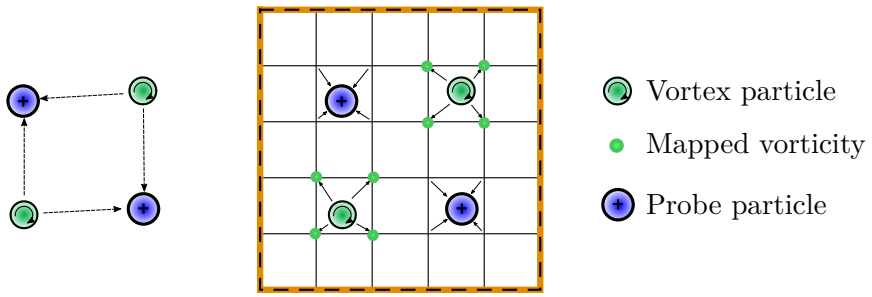

Figure 2.1: Left: Green's method (particle-particle). Right: Poisson method (particle-mesh).

an unbounded domain $\mathcal{G}(\vec{r}) = (4\pi\|\vec{r}\|)^{-1}$ to perform a spatial convolution:

$$\vec{\psi}(\vec{x}) = \mathcal{G} \circledast \vec{\omega} \overset{\text{disc.}}{=} \sum_p \mathcal{G}(\vec{x} - \vec{x}_p)\, \vec{\alpha}_p = \frac{1}{4\pi} \sum_p \frac{\vec{\alpha}_p}{\|\vec{x} - \vec{x}_p\|}\,. \tag{2.10}$$

It should be noted here that the discrete particle representation of $\vec{\omega}$ and $\vec{\psi}$ are in general **not** divergence free, and care must be taken in their application. The velocity field corresponding to the particles is then extracted from the stream function field by applying Eq. (2.5):

$$\vec{u}(\vec{x}) = \nabla \times \vec{\psi} \overset{\text{disc.}}{=} \frac{1}{4\pi} \sum_p \nabla\mathcal{G}(\vec{x} - \vec{x}_p) \times \vec{\alpha}_p(\vec{x}_p) = \sum_p K(\vec{x} - \vec{x}_p) \times \vec{\alpha}_p\,, \tag{2.11}$$

where K represents the Biot-Savart (BS) kernel for a singular vortex element:

$$K(\vec{r}) = -\frac{1}{4\pi} \frac{\vec{r}}{\|\vec{r}\|^3}\,. \tag{2.12}$$

The velocity field is hence calculated through a convolution of the vorticity with the Biot-Savart kernel. In practice the singular representation of the vorticity field of Eq. (2.7) generally gives rise to singular velocity fields, which are of limited practical application. This is overcome by applying a regularisation to the vorticity field.

Regularised vorticity field A more realistic and numerically applicable representation of the vorticity field $\vec{\omega}_\sigma$ is achieved by introducing a smoothing function ζ:

$$\vec{\omega}_\sigma(\vec{x}) = \zeta(\vec{x}) \circledast \vec{\omega} \overset{\text{disc.}}{=} \sum_p \vec{\omega}_p\, dv_p\, \zeta(\vec{x} - \vec{x}_p)\,. \tag{2.13}$$

The smoothing function is generally taken as being a radially symmetric function of ρ, the distance normalised with respect to a characteristic length

σ:

$$\rho = \frac{\|\vec{x} - \vec{x}_p\|}{\sigma}\,, \tag{2.14}$$

where σ effectively specifies the spatial *spreading* of the vorticity, analogous to the standard deviation of a Gaussian distribution. This parameter is generally associated with the inner core region a vortex element, hence the common name for σ: the *core radius*. The smoothing function ζ is chosen such as to ensure that the circulation is conserved: $4\pi \int_0^\infty \zeta(\rho)\rho^2\, d\rho = 1$. A similar procedure with somewhat more careful consideration is carried out as with singular particles in order to extract the stream function [23]. This is again expressed in terms of the corresponding regularised Green's function $\mathcal{G}_\sigma$:

$$\vec{\psi}(\vec{x}) \overset{\text{disc.}}{=} \sum_p \mathcal{G}_\sigma(\vec{x} - \vec{x}_p)\,\vec{\alpha}_p\,. \tag{2.15}$$

This is then used to calculate the regularised velocity field:

$$\vec{u}_\sigma(\vec{x}) = \nabla \times \vec{\psi}_\sigma \overset{\text{disc.}}{=} \sum_p K_\sigma(\vec{x} - \vec{x}_p) \times \vec{\alpha}_p\,. \tag{2.16}$$

Numerous choices of ζ are available, a summary of these is provided in Appendix F. The influence of the choice ζ on the velocity field is shown in Fig. 2.2. For a comprehensive overview the reader is referred to Winckelmans & Leonard [23]. For the work shown here, unless otherwise stated, it should be assumed that a Gaussian regularisation function has been used.

Vortex stretching field It is seen in Eq. (2.9) that the stretching operator $(\vec{\omega}\cdot\nabla)$ must be calculated for resolution of the stretching terms. Alternative forms of these can be written [23]:

$$\underbrace{(\vec{\omega}\cdot\nabla)\vec{u}}_{\text{Classic}} = \underbrace{(\vec{\omega}\cdot\nabla^T)\vec{u}}_{\text{Transpose}} = \underbrace{\frac{1}{2}(\vec{\omega}\cdot(\nabla+\nabla^T))\vec{u}}_{\text{Mixed}}\,. \tag{2.17}$$

The three methods are equivalent if it is assumed that vorticity exactly satisfies $\vec{\omega} = \nabla \times \vec{u}$, however as mentioned above the discrete VPM does not guarantee a divergence-free vorticity field. The three methods hence slightly deviate when applied to the discretised particle set. As described in [23], the transpose scheme alone appears to lead to the exact conservation of total vorticity [87] and to a weak solution of Eq. (2.4) [88]. This certainly appears to be the better scheme for singular particles, for regularised particles the advantages of the transpose formulation are less pronounced. Despite this, the transpose method has been used for essentially all simulations carried

out in this work. Calculating the divergence of the velocity field gives rise to a stretching kernel of the form:

$$K_{\nabla} = \frac{3}{4\pi} \frac{\vec{r}}{\|\vec{r}\|^5} \,. \tag{2.18}$$

This appears in combination with the BS kernel in Eq. (2.11) for the closed-form solutions of the stretching, which are described for available schemes in Appendix F. For brevity these are not expressed here.

Viscous diffusion There are a number of approaches applied to calculate viscous diffusion with the VP method. An example is the random walk method, whereby a random velocity components is added to the particle velocities with certain statistical normalisations [89]. A further example is the core-spreading method [90], a much more intuitive approach where the smoothing parameter σ varies in time to account for diffusion. Certainly the most successfully applied method however is the particle strength exchange (PSE) scheme introduced by Degond & Mas-Gallic [27], whereby the diffusion operator ∇^2 is approximated by an integral operator:

$$\nabla^2 \vec{\omega}(\vec{x}) \simeq \frac{2}{\sigma^2} \int (\vec{\omega}(\vec{y}) - \vec{\omega}(\vec{x}))\, \eta_\sigma(\vec{x} - \vec{y})\, d\vec{y} \,, \tag{2.19}$$

where the diffusive function η_σ follows directly from the choice of smoothing function: $\eta(\rho) = -(d\zeta/d\rho)/\rho$. The choice of Gaussian smoothing in this case has the computational advantage that $\eta(\rho) = \zeta(\rho)$. Although not strictly a Green's function, this approach allows the diffusive nature of the vorticity field to again be represented with a simple particle-particle representation. The PSE scheme, however, is also subject to stability constraints, which depend on the time integration scheme used. As described in [40], this constraint specifies the timestep must be such that $\nu \Delta t\, \sigma^{-2} \leq \phi_s$ where for the Euler explicit scheme $\phi_s = 0.595$ and for the Adams-Bashforth 2^{nd} order scheme $\phi_s = 0.297$.

2.3.2 Far Field Behaviour

The interaction between two particles is completely described with closed-form expressions as described above. An important observation can be made here: As the distance between evaluation point and source particle becomes very large ($\rho \to \infty$), the influence asymptotes to that of a singular particle. This region is referred to as the *far field* of the particle, as opposed to the *near field*, where the smoothing function ζ has a significant impact on the

particle's influence. This is demonstrated in Fig. 2.2 for a range of standard regularisation functions: Singular, low-order algebraic (LOA), high-order algebraic (HOA) [22], Gaussian, and Hejlesen [91]. It is observed that they asymptote to the singular kernel in the far field. Equivalent behaviour is

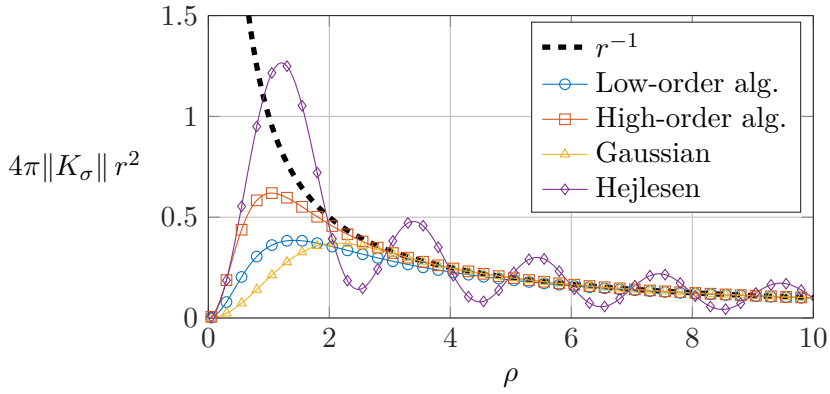

Figure 2.2: Biot-Savart kernel for a range of regularisation functions.

observed for the stretching kernel, however this has not been exploited in this work. The PSE kernels described above vanish very quickly, generally within $5\,\sigma$, implying physically that the action of viscosity is dominated by near field interactions. This is an intuitive result as the friction forces scale with the velocity gradient, which is greatest for small ρ. In the far-field the tendency towards the singular BS kernel implies the tendency towards an irrotational far field influence, where viscous effects play no role. This shall be demonstrated in Chapter 3.

2.4 VPM: Poisson Method

When using the Poisson method, a particle-mesh approach is applied. Here the volume solution is attained on a volume mesh, from which all desired flow quantities are extracted. This process substantially is composed of three steps:

i) Vorticity defined at source particle positions is mapped to the regular grid with chosen mapping function- Section 2.4.1.

ii) Solve Eq. (2.6) on the grid for resolution of stream function $\vec{\psi}$ using Poisson solver- Section 2.4.2.

iii) Extract flow from the grid using FD and map desired quantities back to particles with chosen mapping function.

These steps are illustrated in Fig. 2.1.

2.4.1 Mapping Functions

Within VPM the vorticity field is defined at a set of particle positions (Eq. (2.7)), which are in general not regularly distributed in space. The Poisson solver however requires specification of vorticity on a regular grid. This implies that the known vorticity field of each particle must be transferred to the surrounding regular grid points of the Poisson solver. This can be accomplished with a mapping function $M(\vec{x})$. These are formulated such that the spatial moments of vorticity are preserved up to order m: $\vec{\Omega} = \vec{\omega}\, x^m$. A set of common mapping functions are shown in Fig. 2.3: The

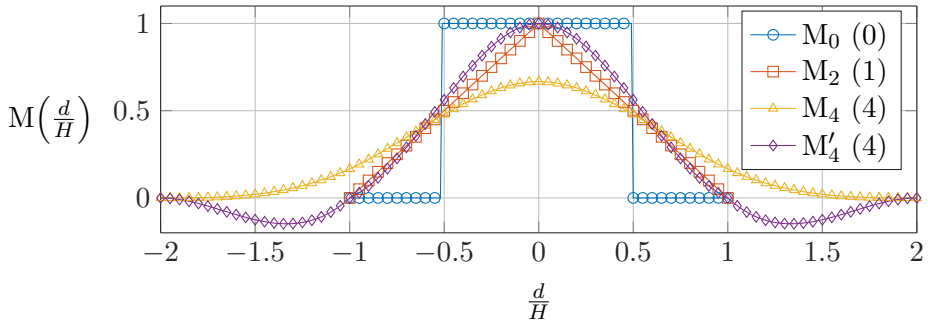

Figure 2.3: Mapping functions to transfer quantities from Lagrangian particle positions to a regular grid of cell width H: d is the distance to the neighboring grid point. The order of the mapping is shown in parenthesis.

mapping function determines, based on the relative position within the grid, the portion of vorticity mapped to each of the surrounding grid nodes. The stencil width refers to the number of surrounding points to which the particle vorticity is mapped, this increases with desired mapping order m. The mapping functions are generally described as one-dimensional, and produce a mapping in a single dimension $M(x)$. For mapping in higher dimensions the tensor product is taken $M_{x,y,z} = M(x)\, M(y)\, M(z)$. These mapping functions have the additional advantage that they can be used not only for mapping *to* the grid, but also for interpolating results from the grid, which is necessary as the field variables are generally solved on the grid using FD. A visualisation of the mapped field is given in Fig. 2.4.

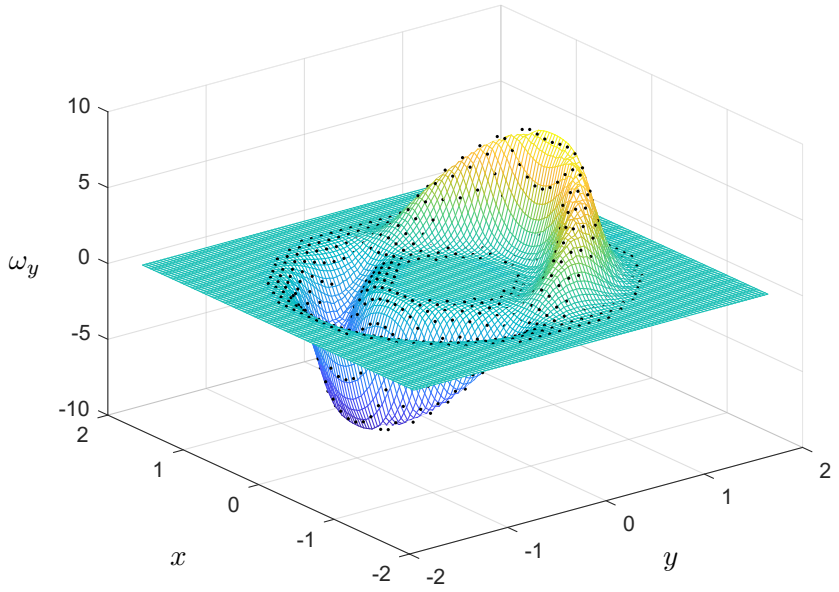

Figure 2.4: The x-y plane of a dense vortex ring- investigated in Chapter 5. The ω_y field is shown corresponding to the mapped vorticity field. The particle positions and magnitudes are shown with points and the resultant mapped vorticity field is shown with the contour.

2.4.2 Poisson Solver

The solution to Eq. (2.6) on a regular grid can be achieved numerically through the application of FD methods. The Laplacian of the stream function at a given node is expressed in terms of the surrounding vorticity values by means of an FD stencil. The unknowns values of ψ_i over the N nodes can be expressed as a block-tridiagonal linear equation system which can be solved with numerous approaches. For a comprehensive overview, the reader is referred to Hockney & Eastwood [92], here common approaches are described:

- **Direct Evaluation:** By directly using Gaussian elimination the system has computational complexity $\mathcal{O}(N^3)$.
- **Generalised Thomas Algorithm:** An optimised Gaussian elimination procedure with theoretical complexity $\mathcal{O}(N^{3/2})$ [93].
- **Spectral Solver:** Representing the solution in terms of a finite Fourier series allows the linear system to be reformulated in terms of

Fourier coefficients. Application of the Fast Fourier Transform (FFT) allows theoretical complexity reduction to $\mathcal{O}(N\log(N))$ [94].

- **Multigrid Method:** The solution is found iteratively between numerous grids of varying resolution by using the residual at one grid level to find the solution vector at a coarser grid level to remove lower frequency errors. This method has a theoretical complexity of $\mathcal{O}(N)$ [95].

Clearly the multigrid method has the optimal theoretical complexity, however in practice $\log N$ behaves essentially as a constant for large N and significant advances made in the field of FFT algorithms (e.g. for image compression) imply that the spectral solver generally leads to the best performance.

Fast Poisson solvers An algorithm which applies the FFT to the solution of the Poisson equation is generally referred to as a *Fast Poisson* (FP) solver. These are commonly formulated to be applicable to the solution of the Helmholtz equation, of which the Poisson equation is a special case:

$$\underbrace{-\nabla^2\vec{\psi}+q\vec{u}=\vec{f}}_{\text{Helmholtz Eq.}} \overset{q=0}{\longrightarrow} \underbrace{-\nabla^2\vec{\psi}=\vec{f}}_{\text{Poisson Eq.}} \overset{f=0}{\longrightarrow} \underbrace{\nabla^2\vec{\psi}=\vec{0}}_{\text{Laplace Eq.}}. \tag{2.20}$$

This greatly expands the scope of applicability of the solvers to problems of astrophysics, acoustics, electromagnetics and potential flow, amongst others. The fast Poisson method is based upon applying a discrete Fourier transform to the variables, and then solving the corresponding continuous problem from the discrete formulation given above. It can then be shown that the Fourier modes are the eigenvectors of the system matrix, and furthermore are eigenmodes of the continuous solution [94]. The coefficients of the Fourier modes can be computed by applying an FFT. The solution of the problem is sought on a domain $\mathcal{D}$ with boundary $\partial\mathcal{D}$. The boundary condition, the value of the stream function on $\partial\mathcal{D}$ must be known as this allows for a unique solution. This requirement is important to note at this point, as this influences greatly the method of solution. There are numerous approaches to address this:

- **Lattice Green's Functions** The Fourier-space Green's function on the grid are calculated such that they satisfy the desired boundary conditions on the given domain. The vorticity field $\vec{f}$ (here $\vec{\omega}$) is first Fourier-transformed, the solution found in Fourier space. The backward transform of the solution is then calculated [96]. This includes the **Hockney-Eastwood** method where an additional *padding* zone

appended to the domain allows for unique specification of the Fourier coefficients for the solution of the unbounded problem [92].

- **James-Lackner Algorithm** This involves separating the solution into two contributions: a homogeneous solution (with zero boundary condition) and a boundary forcing contribution- chosen to satisfy continuity of the full solution at the boundary [97, 98].
- **Method of Local Corrections** A multigrid-style approach where the solution is found on progressively coarser grids and the result is interpolated or *contracted* to finer grids [11, 99].

The method of local corrections is ideal for implementation purposes as only the Poisson solver is required for calculating the solution. The interpolation process however is relatively complex and great care needs to be taken to ensure influences from nested grids are not counted multiple times [99]. In terms of computational expense the Hockney-Eastwood method is most optimal. However, the method requires a rectangular grid, the vorticity field is therefore not tightly contained which can lead to unnecessary memory overhead. The padding regions additionally greatly increase memory overhead. Despite this, this method has enjoyed great success and is commonly applied to VPM simulations [100, 32]. The application of the James-Lackner (JL) algorithm however ensures tight domain constraints and is amenable to application of the multilevel method, which shall be described in Chapter 3. A more detailed description of the JL algorithm is now given.

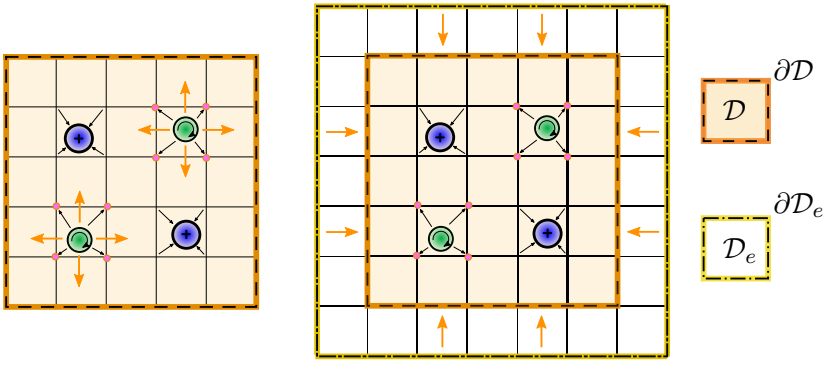

Figure 2.5: The solution to $\vec{\psi}$ is sought within $\mathcal{D}$. This requires specification of $\vec{\psi}$ on the boundary $\partial\mathcal{D}$. Left: Direct specification with Green's function. Right: Use of the James-Lackner algorithm.

James-Lackner algorithm Rather than selecting the solution domain $\mathcal{D}$ (here the Poisson grid) large enough so that the stream function on the

boundary $\partial\mathcal{D}$ vanishes, the unbounded problem can be treated on a compact grid with an equivalent *boundary forcing* term, as described in James [97] and Lackner [98]. A necessary condition for the JL method is that the vorticity has *compact support*, implying that a slightly larger domain $\mathcal{D}_e$ must be constructed such that the region of interest $\mathcal{D}$ is padded with zero vorticity, as is seen in Fig. 2.5. The solution is attained by treating the problem as the superposition of two potentials:

i) **The homogeneous potential** $\vec{\psi}_0$: The Poisson equation is solved with zero boundary forcing (boundary potential zero):

$$\nabla^2\vec{\psi}_0(\vec{x}) = \vec{\omega}(\vec{x}) \;,\; \vec{x} \in \mathcal{D}_e \,. \quad \vec{\psi}_0(\vec{x}) = \vec{0} \;,\; \vec{x} \in \partial\mathcal{D}_e \,. \tag{2.21}$$

ii) **The single-layer potential** $\vec{\psi}_1$: The Poisson equation is solved with zero volume forcing:

$$\nabla^2\vec{\psi}_1(\vec{x}) = \vec{0} \;,\; \vec{x} \in \mathcal{D} \,. \quad \vec{\psi}_1(\vec{x}) = \vec{\omega}_\partial \;,\; \vec{x} \in \partial\mathcal{D} \,. \tag{2.22}$$

The full solution is then attained by superposing the two contributions: $\vec{\psi} = \vec{\psi}_0 + \vec{\psi}_1$. The boundary forcing term $\vec{\omega}_\partial$ for the single-layer step is the external forcing which occurs due to the truncation from an unbounded problem to a bounded one. This is treated as a source influence distributed along the boundary $\partial\mathcal{D}_e$, and can be calculated with the following convolution:

$$\vec{\psi}_1(\vec{x}) = \int_{\partial\mathcal{D}_e} \vec{\gamma}(\vec{x}_\partial)\, \mathcal{G}(\vec{x} - \vec{x}_\partial) d\vec{x}_\partial \,, \tag{2.23}$$

where the force density on the boundary $\vec{\gamma}$ represents the jump in the normal derivative on $\partial\mathcal{D}_e$: $\frac{\partial\psi_1}{\partial n}$. This influence is valid both interior and exterior to $\mathcal{D}_e$. The source density $\vec{\gamma}$ can be determined simply by requiring continuity of the solution $\psi = \psi_0 + \psi_1$ across the boundary:

$$\frac{\partial\psi}{\partial n} = \frac{\partial\psi_0}{\partial n} + \frac{\partial\psi_1}{\partial n} = 0 \quad \longrightarrow \quad \vec{\gamma}(\vec{x}_\partial) = \frac{\partial\psi_1}{\partial n} = -\frac{\partial\psi_0}{\partial n} \,. \tag{2.24}$$

In summary, the total solution is found by calculating first the homogeneous potential $\vec{\psi}_0$. The normal gradient of $\vec{\psi}_0$ on $\partial\mathcal{D}$ is then calculated with FD to specify boundary forcing $\vec{\gamma}$. The boundary forcing allows calculation of $\vec{\psi}_1$ within the domain $\mathcal{D}_e$. An efficient approach for the total solution here is to calculate $\vec{\psi}_1$ only over $\partial\mathcal{D}$. The homogeneous solution $\vec{\psi}_0$ is extracted at this boundary from the previous step and added to $\vec{\psi}_1$. The total boundary condition on the boundary $\partial\mathcal{D}$ being known, the FP solver can be executed again for the full solution. This implies that for the full solution the JL algorithm requires two FP evaluations plus a boundary gradient calculation.

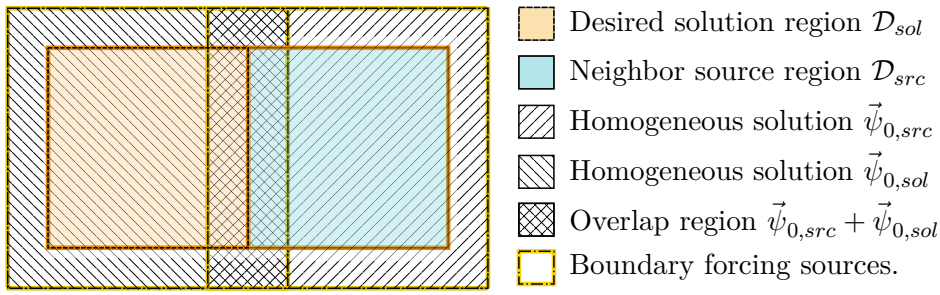

Figure 2.6: The James-Lackner algorithm applied to neighboring source regions.

Multiple Domains The problem described above is perfectly applicable to multiple regions of vorticity, allowing a compact treatment of vortical domains. As described previously, an enlarged region must be taken to ensure that the vorticity on the grid has compact support. For neighboring domains this leads to overlaps between the regions as visualised in Fig. 2.6. Care must simply be taken to ensure that the solution on the desired boundary ($\partial\mathcal{D}_{sol}$ in Fig. 2.6) includes overlapping homogeneous solutions from neighboring region $\vec{\psi}_{0,src}$, as well as the boundary forcing contributed by neighboring regions $\vec{\gamma}_{src}$. Although illustrated in Fig. 2.6 only for a single neighbor position, the extension to any neighboring source region and in fact to 3D is straightforward.

2.4.3 Resolution of Field Quantities

After the solution with the FP solver, the extraction of desired field quantities from the $\vec{\psi}$ grid is achieved by using FD calculations. This has the computational advantage that the grids are uniform, the FD stencils are hence specified at the beginning of the simulation and thereafter remain unchanged. The quantities at the desired evaluation points are then specified using the mapping procedure described above.

Velocity field When the solution to the stream function $\vec{\psi}$ is known on the grid, the velocity field can be extracted by applying Eq. (2.5). The partial derivatives of velocity $\partial u_i/\partial x_j$ can be determined by applying a FD scheme of desired order to calculate the components of $\nabla \times \vec{\psi}$. This is visualised in Fig. 2.7.

Stretching field FD can again be applied upon the calculated velocity field in order to calculate $\nabla\vec{u}$ (a second order tensor). Vorticity has been

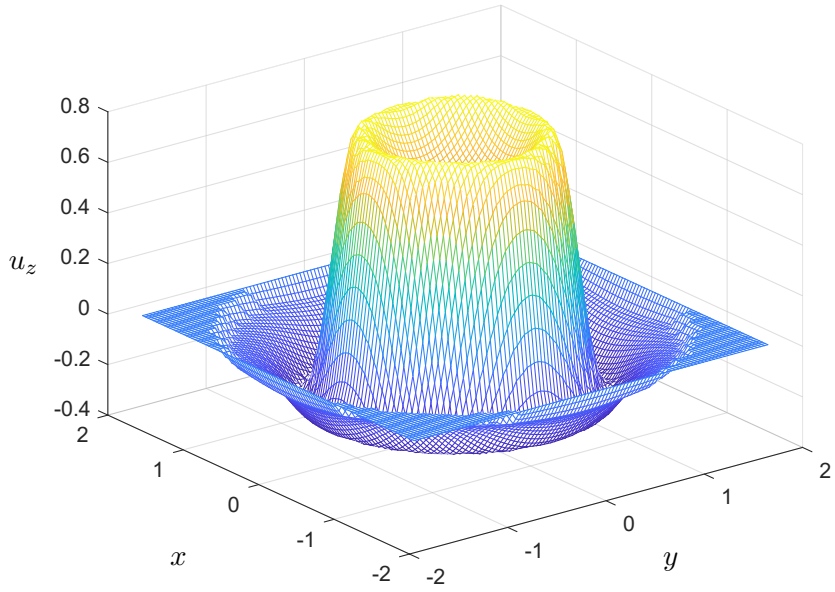

Figure 2.7: Outputs of step iii) of the Poisson method. FD stencils have been used to extract the velocity field. The contour shown corresponds to u_z. The field here corresponds to the flow problem in Fig. 2.4.

previously stored on the grid for the Poisson solver, the procedure to calculate $(\omega \cdot \nabla)\vec{u}$ is hence straightforward.

Calculation of viscous diffusion With the value of vorticity being known on the grid, a Laplacian FD stencil can be used to extract $\nabla^2\vec{\omega}$. As with velocity and stretching, this is mapped back to the particles as desired.

2.5 Numerical Considerations

The numerical treatment of the flow field with Lagrangian elements introduces modelling challenges. If unaccounted for, these can lead to erroneous results and badly conditioned particle sets. The following sections detail important modelling procedures which must be applied.

Remeshing The regularised vortex particle method can be proven to converge to smooth solutions of the Euler equations with grid-free formulations

[101, 102]. The solution converges, however only under the assumption that the number of particles is increased, and that the particles overlap: $\rho \leq 1$. The combined effects however of Lagrangian convection and vortex stretching often imply that neighboring particles bunch together or are stretched apart and the aforementioned requirement is no longer fulfilled. If left unchecked, this generally leads very quickly to a badly conditioned particle set with discontinuous field quantities. This can be overcome by regularly *remeshing* the particle set. This is carried out by mapping the particles onto a regular grid with a desired choice of mapping function (see Section 2.4.1). This is illustrated in Fig. 2.8. This furthermore introduces a spatial expansion which artificially accounts for spatial diffusion. Remeshing introduces a constraint between vortex core size σ and characteristic grid size, which can be used to ensure particle overlapping. The frequency of

Figure 2.8: Remeshing applied to the neutral plane of a vortex ring. Left: Azimuthal description, particle separation is seen to be inconsistent. Right: After remeshing (M'_4 scheme) the vorticity distribution is seen to be conserved, however the particles are now regularly distributed.

remeshing is user-defined and optimum specification depends on the flow scenario and numerous spatial grid and time integration parameters. It is furthermore not strictly necessary to remesh the entire particle set, but rather to restrict remeshing to regions where e.g. increased particle stretching is seen. The remeshing procedure produces regularly spaced particles, each with the characteristic volume V_{char} of the grid. This shall be detailed in Chapter 4.

Magnitude filtering Remeshing in general maps each particle onto numerous surrounding particles (see Fig. 2.8). Without some form of filtering process, one can imagine how the outer boundary of a domain continuously expands with each remeshing procedure and the number of particles in a simulation grows without bound. One practical way of overcoming this

issue is to perform magnitude filtering, as described in Cocle et al. [28]. The particle set is first scanned such that the maximum vorticity ω_{max} is found. Following this, the particle set is again scanned and all particles for which $\|\vec{\omega}\| < \kappa\,\omega_{max}$ are discarded. Very low vorticity particles contribute negligibly to the surrounding flow field. A further method is to perform *spatial* particle filtering. Here particle are removed in regions where the flow is known to be restricted or where the relative energy of the particles contribute negligibly to the global solution.

Vorticity field divergence correction The discretised treatment of the flow field generally leads to a vorticity field which is not divergence-free. For a particle set which is being time-integrated, the particle set after updating should continue to satisfy the relation of Eq. (2.13). These effects amplify for either long-time or under-resolved simulations. Numerous approaches have been suggested to remedy this. Pedrizzetti [103] suggests using Eq. (2.13) to apply a relaxation to the time integration of the particle set. A further method suggested by Winckelmans & Leonard [23] reconstructs the divergence-free field with an expression from Novikov [104] and then expresses this as a linear system to be solved. These are both procedures of complexity $\mathcal{O}(N^2)$. A much more efficient method, formulated within the framework of the Poisson solver, is to re-project the vorticity onto a divergence-free basis [28, 105]:

$$\vec{\omega}_{new} = \vec{\omega}_{old} - \nabla F \quad \text{, where:} \quad \nabla^2 F = \nabla \cdot \vec{\omega}\,. \tag{2.25}$$

The divergence of the vorticity field is hence used to solve for the intermediate variable F, which acts to decrease divergence of the vorticity field. The fast Poisson solver can again be used, here with zero Dirichlet BC. This allows a quick, non-diffusive correction to the vorticity field. This is applied at regular time steps, again user-defined. The calculation overhead is insignificant as only a single call to the fast Poisson solver is required.

Criteria of quality for a VP simulation Two criteria guarantee that the field is adequately resolved temporally and spatially. The first parameter is the *mesh* Reynolds number, which guarantees that the computational grid is sufficiently fine to resolve viscous scales present in the flow:

$$Re_{\text{mesh}} = \frac{\omega_{max} H}{\nu}\,, \tag{2.26}$$

where H is the characteristic grid size. For a well-resolved simulation $Re_{\text{mesh}} \approx \mathcal{O}(1)$. Furthermore, the effects due to diffusion must also be

correctly captured by ensuring the diffusion time scales are correctly resolved:

$$D = \frac{\nu \Delta t}{H^2} \,. \tag{2.27}$$

For a well-resolved simulation $D \approx \mathcal{O}(1)$. Combining these two one finds that $\omega_{max}\Delta t \approx \mathcal{O}(1)$ should be satisfied. As described in Ploumhans [40] this parameter essentially limits the relative rotation of the particles.

Turbulent shear stresses The effects of turbulence can be included with the use of a large-eddy simulation (LES) approach. In the LES approach the velocity field is decomposed using an appropriate convolution into a filtered (or resolved) field and a residual (or sub-grid scale (SGS)) field [15]. The effect of the residual field is included in the momentum Eq. (2.4) in the form of an additional shear stress term τ_t:

$$\frac{d\vec{\omega}}{dt} = \frac{\partial \vec{\omega}}{\partial t} + (\vec{u}\cdot\nabla)\vec{\omega} = (\vec{\omega}\cdot\nabla)\vec{u} + \nu\nabla^2\vec{\omega} - \nabla\cdot\tau_t \,. \tag{2.28}$$

Closure of the system is obtained by specifying τ_t with a model of the SGS stress tensor. Care must be taken to ensure sufficient grid resolution for the SGS modelling. A common measure for this is the Pope criterion: $M = k_r/(K + k_r)$ [106], where k_r and K are the turbulent kinetic energies of the residual and resolved motions, respectively. For a well-resolved LES simulation generally $M < 0.2$ is suggested. Two SGS models have been applied in the work here, substantially as described in Cocle et al. [28]. The implementation of both methods is described in Chapter 4 and validation has been carried out in Chapter 5. The two methods are briefly described here:

HV Model: The first approach makes use of a hyper-viscosity operator as described in Jeanmart et al. [30], written here in general form for an order-n operator:

$$\tau_t = -2C(-1)^n\Delta^{2(n+1)}\|S_{ij}\|\,\nabla^{2n}S_{ij} \,, \tag{2.29}$$

where Δ is the length scale, here taken to be the grid size H and S_{ij} is the rate of strain tensor of the velocity field. This reduces to the classical Smagorinsky eddy viscosity model [18] for the case $n = 0$. The use of higher order derivatives extends the range of the inviscid inertial cascade by pushing dissipation towards the smaller scales in the flow [107]. In the work here the case $n = 1$ was applied, in which case application to the vorticity field is expressed as:

$$\nabla\cdot\tau_t = \frac{C}{T_0}(H^2\nabla^2)^2\vec{\omega} \,, \tag{2.30}$$

where C is a Smagorinski constant and T_0 a global time constant. This approach carries with it a very low computational cost, however the necessary specification of a global time constant makes the application less general.

RVM Model: The second approach makes use of a regularised variational multiscale approach. Here the diffusion operates solely on the high-frequency or *small-scale* vorticity field: $\vec{\omega}_s$, which is found by applying a spatial filter to the velocity field n times to achieve an n^{th} order filter as described in Jeanmart et al. [30]:

$$\tau_t = \nu_{sgs}[\nabla\vec{\omega}_s + (\nabla\vec{\omega}_s)^T] \quad \text{where:} \quad \nu_{sgs} = C_r^n \Delta^2 \sqrt{2S_{ij}S_{ij}}\,. \tag{2.31}$$

The eddy viscosity ν_{sgs} is seen to take the form of the standard Smagorisnki expression with C_r dependent upon filter order n, these are given in Cocle [76]. Although more computationally demanding, this method does not require global flow constants and is therewith more general.

Chapter 3 Multilevel Integration

The main contribution of this work is the application of the multilevel multi-integration cluster (**MLMIC**) scheme as described in van Garrel [108] to the vortex particle method (VPM). When evaluated directly, a VPM calculation with N particles has a computational complexity of the order $\mathcal{O}(N^2)$. This computational scaling is unrealistic for large N as calculations are prohibitively slow. Application of the MLMIC method to spatially integrate the particle set allows the complexity to be reduced to $\mathcal{O}(N)$, massively increasing efficiency for large particle sets. The general problem description and nomenclature shall first be described in Section 3.1 followed by a description of the method of spatial discretisation in Section 3.2. The underlying concepts and calculation steps of the MLMIC method are then described in Section 3.3. The MLMIC method has been thoroughly validated for the integration kernels of interest in this work by applying the method to a representative geometry in Section 3.5.

For clarity some points of nomenclature shall be addressed here: A **source** refers to an object which influences the surrounding flow field, this can be either distributed continuously in space or specified discretely at points. A **probe** refers to a discrete point where flow quantities are to be calculated. For the VPM it is often necessary to evaluate the field quantities directly at particle positions, in which case sources and probes coincide.

3.1 Description of the Problem

The spatial density of the source distribution is given by $\alpha(\vec{y})$. In general α can be scalar, vector or tensor. The influence of this source field on a position $\vec{x}$ is given by convoluting over the entire source field with the kernel $\mathcal{K}$. The canonical problem is to find the total influence $\mathcal{I}$ due to the source field at a point $\vec{x}$:

$$\mathcal{I}(\vec{x}) = \int_{\mathcal{R}} \mathcal{K}(\vec{x}, \vec{y}) \alpha(\vec{y}) \, d\vec{y} \,. \tag{3.1}$$

The source representation α can be continuous or discrete. In the case of VPM, source positions $\vec{y}_p$ are discrete and each represent a discrete volume dV_p. This allows the above equation to be expressed as a discrete sum over

source particle positions:

$$\mathcal{I}(\vec{x}) = \sum_{p \in \mathcal{R}} \mathcal{K}(\vec{x}, \vec{y}_p)\alpha(\vec{y}_p)\, dV_p\,. \tag{3.2}$$

Evaluating the terms in this equation by making use of the convolution kernels (App. F) shall hereafter be referred to as **direct evaluation**. This is illustrated visually for a simple case in Fig. 3.1. For the analysis of a particle set of size N, the particles themselves are probes, one hence has to carry out the above evaluation for N probe positions. The summation displays immediately how the problem scales as $\mathcal{O}(N^2)$ as for each probe evaluation $\mathcal{I}(\vec{x}_i)|_{i=1:N}$, N calculations are required.

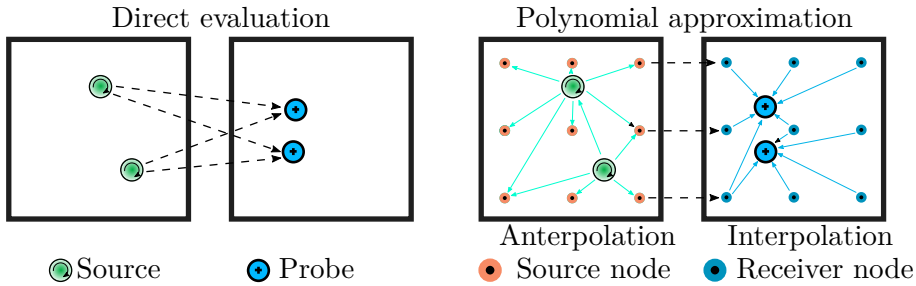

Figure 3.1: A demonstration of the underlying concept of the MLMIC method. Left: Particle influence calculated directly. Right: Influence of source nodes on receiver nodes is calculated.

3.2 Spatial Discretisation

The region $\mathcal{R}$ is uniformly discretised into non-overlapping regions $\mathcal{B}_i$, $\mathcal{R} = \cup\mathcal{B}_i$. These regions are cubic in form. This is not a requirement, but is done rather for implementation purposes– see Chapter 4. These regions are hereafter referred to as **boxes**. This allows expression of the above equation as:

$$\mathcal{I}(x) = \sum_{\mathcal{B}_i \in \mathcal{R}} \mathcal{I}(\mathcal{B}_i) = \sum_{\mathcal{B}_i \in \mathcal{R}} \left\{ \sum_{p \in \mathcal{B}_i} \mathcal{K}(\vec{x}, \vec{y}_p)\alpha(\vec{y}_p)\, dV_p \right\}, \tag{3.3}$$

where $\mathcal{I}(\mathcal{B}_i)$ is the influence due to particles within box $\mathcal{B}_i$. The influence is therewith discretised to contributions from individual source boxes. The spatial discretisation allows for a simplified coarsening of the domain by using *nested* box regions. The smallest boxes have a characteristic side length H_b and are hereafter referred to as **base boxes** and are denoted with subscript 0: $\mathcal{B}_{0,i}$. These are geometrically contained within coarser

boxes, denoted as $\mathcal{B}_{l,i}$ where l refers to the box **level**. This is illustrated for three grid levels in Fig. 3.2. The representation with coarser box regions

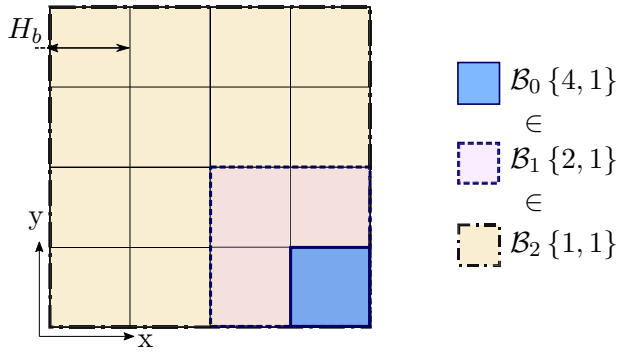

Figure 3.2: The nested box geometry illustrated for a 2D case (quadrants). Here each box $\mathcal{B}_1$ contains four boxes $\mathcal{B}_0$. In 3D (octants) each box $\mathcal{B}_1$ contains eight boxes $\mathcal{B}_0$. Box Cartesian IDs are given in parenthesis (see Chapter 4).

allows the discretised integral above to be expressed in terms of coarser box regions:

$$\mathcal{I}(\vec{x}) = \sum_{\mathcal{B}_{0,i}} \mathcal{I}(\mathcal{B}_{0,i}) = \sum_{\mathcal{B}_{1,i}} \mathcal{I}(\mathcal{B}_{1,i}) = \cdots = \sum_{\mathcal{B}_{l,i}} \mathcal{I}(\mathcal{B}_{l,i})\,. \tag{3.4}$$

This representation will be exploited frequently in the application of the multilevel method.

3.3 Multilevel Multi-Integration Cluster

The MLMIC method reduces the complexity of the calculation of Eq. (3.2) by spatially approximating the kernel influence $\mathcal{K}$. This is carried out in such a way that the order of approximation decreases with increasing distance from the evaluation point by using geometric coarsening. As opposed to direct evaluation, where the exact source and probe positions are used to calculate the influence $\mathcal{I}(\vec{x}, \vec{y}_p)$, in the MLMIC method the source distribution within a box is mapped to **source nodes** at predefined positions within the box. The influence on **receiver nodes** within the probe-containing box is then calculated, these act as pseudo-probes. The influence is then interpolated to the actual probe positions, this is illustrated in Fig. 3.1.

The method is applicable to d-dimensional space $\mathcal{R}^d$, however in the work here the method has been applied only to 2D ($\mathcal{R}^2$) and 3D ($\mathcal{R}^3$) problems. For a full mathematical description the reader is referred to the comprehensive description given in van Garrel [108]. The MLMIC method is composed of three primary steps:

1. **Anterpolation**: The source distribution is approximated at source nodes with polynomial interpolation;
2. **Interaction**: Influence of source nodes on receiver nodes is calculated.
3. **Interpolation**: A polynomial approximation is used to interpolate influence from receiver nodes to probes.

The general concept is visualised in Fig. 3.1. These three processes shall be described in the following sections. For simple reference in that which follows, the naming convention **parent** and **child box** will be taken to refer to a box pair where the child box is **within** the parent box.

3.3.1 Anterpolation

Boxes containing sources are referred to as **source boxes**. Within each source box the kernel function is approximated at a set of **source nodes** within the box. The approximation and node position is specified by a suitable polynomial, described in Section 3.3.3. This results in a mapping matrix I_{mn} which has dimension $[n_{int}, n_p]$, where n_{int} is the number of source nodes and n_p is the number of sources. The inverse process: mapping to the source nodes is achieved by using the adjoint-interpolation (hence **an**terpolation), which for real-valued sources is simply the transpose matrix I^T. In this sense, anterpolation can be considered to be the process by which the spatial distribution of the sources is represented at the source nodes of each box. The anterpolation process is illustrated graphically in Fig. 3.3. This source description can take on arbitrary forms: scalar, vector (used here) or tensor. The location of the sources nodes is a function of the interpolation scheme applied and is described in Section 3.3.3.

The number of source nodes in each spatial dimension is defined by the order P of the approximating polynomial function, so that the total number of source nodes $n_{int} = P^d$. The greater P, the better the spatial approximation of the kernel function. This will be demonstrated in Section 3.5. By making use of the nested geometry approach, the source distribution can be spatially coarsened by anterpolating to the parent box, as illustrated in Fig. 3.3.

This provides a method to approximate the source distribution at higher box levels within Eq. (3.4). If the order P between parent and child is equivalent, an advantage is gained in implementation as the anterpolation matrix I^T between between both levels is equivalent and hence need only be calculated once. This naturally also incurs a penalty as the spatial approximation becomes less accurate with higher levels. This is used in the work here, however custom P_l at different grid levels can be applied if desired.

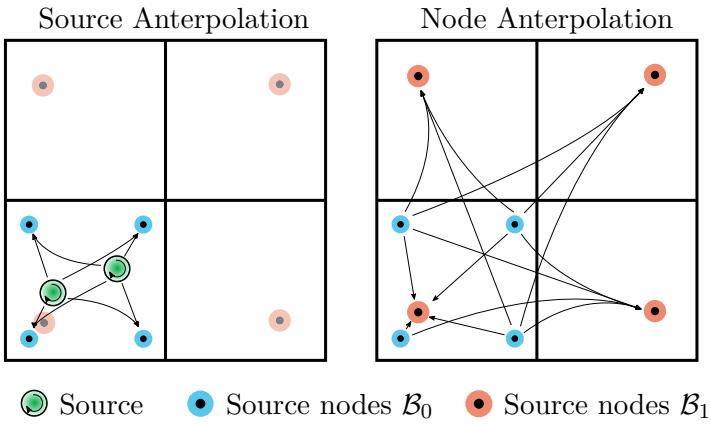

Figure 3.3: Anterpolation process. Particle source strength is mapped to source nodes using anterpolation. The same process is carried out for higher box levels in order to coarsen the spatial source representation. For illustration purposes here, the order of anterpolation $P_{int} = 2$.

3.3.2 Interaction

With direct evaluation, interaction is calculated between sources and probes. In the MLMIC method, however, interaction is calculated between source nodes and receiver nodes. Much as with the source nodes, receiver nodes are nodes within boxes containing probes, or **receiver boxes**. The advantage of this approach is that it allows interaction templates to be identified, making use of source-receiver box symmetries. This is illustrated in Fig. 3.4. The width of this template is specified by the parameter NF_{int}: the number of surrounding source boxes (in each spatial direction) which contribute to the desired receiver box. This region shall hereafter be referred to as the **near field** of the receiver box. Everything outside of this is referred to as the **far field**. For many kernels of interest (most importantly for the work here the Biot-Savart kernel and the stream function kernels), this interaction is easily expressed as a matrix multiplication. The anterpolated source node strengths are simply multiplied by the interaction template to get the corresponding influence at the desired receiver nodes. This simplifies the influence $\mathcal{I}(\mathcal{B}_i)$ to a matrix multiplication, this matrix is hereafter referred to as an **interaction template**. The interaction template need only be calculated once at the beginning of the simulation from the relative positions between source and receiver nodes of the given template. This appears at first to account **only** for the influence on a reciever box due to the near field. The influence of the far field can however also be taken into account by using the concept of *nested interaction templates*.

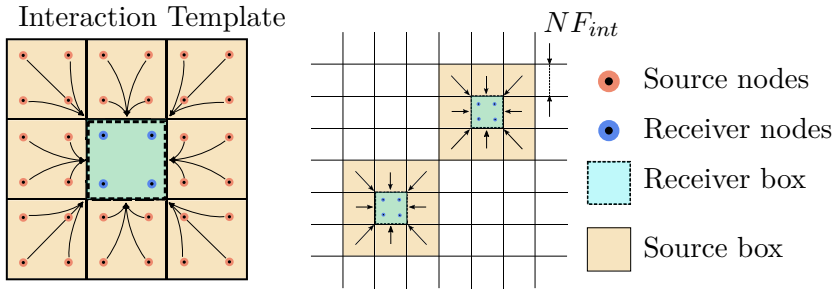

Figure 3.4: Left: Box interaction template: The influence of source nodes on receiver nodes is calculated, arrows indicate influence. Right: Demonstration of repetition of the interaction template for a region containing multiple receiver boxes.

Nested interaction templates The use of box regions $\mathcal{B}_i$ to enable a nested geometrical description can be exploited to represent interactions at different box levels. Comparing the near field template of a child box to the near field template of a parent box in Fig. 3.5, one sees immediately that the templates are geometrically equivalent, however scaled. Assuming an equivalent P_l at every box level, it becomes clear that the relative positions between source nodes and receiver nodes at higher levels are also simply scaled. The interaction templates are therefore also geometrically scaled.

> **Note:** The geometric scaling of the interaction template **cannot** generally be assumed and this assumption will be investigated later for kernels of interest.

Influence is always calculated between sources and receivers nodes at the same box level. This implies two important consequences:

1. The influence calculated within parent boxes must be interpolated to children boxes, this shall be described in the following section.
2. The near-field influence of the child box must be somehow be excluded from the interaction template, otherwise the influence will erroneously be counted twice. This is achieved simply by ensuring that the corresponding portions of the interaction template are nullified.

This principal is applied recursively to higher levels boxes (aka parents of parents) until the entire source field contributes to the solution. This demonstrates how higher level regions influence a receiver box in such a way that increasing distance from the receiver box implies coarser approximation

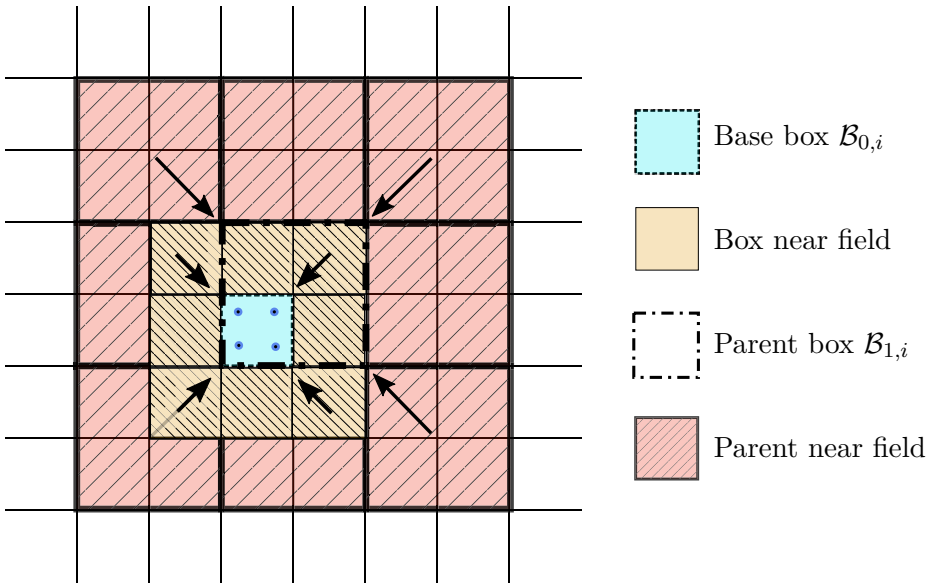

Figure 3.5: The interaction template for the base box (downward hatching) is equivalent to that of the parent box (upward hatching), however with geometric scaling. The influence of the near field of the parent can hence also be accounted for with the given interaction template. This motivates anterpolation to parent boxes. This concept is applied to calculate the source influence of progressively larger surrounding regions.

of the source field. The interaction template itself does not need to be modified in order to account for interaction at higher grid levels, all that is necessary is that the anterpolated strengths be multiplied by an appropriate scaling factor which represents the geometric scaling of the kernel. This is usually simply a factor of 2^{-l}. By correctly concatenating the source node strength matrix to be multiplied with the interaction template, this implies that the entire far field calculation carried out with the MLMIC method of an interaction template is practically represented as a single matrix multiplication.

3.3.3 Interpolation

In the interaction step the influence the surrounding source field was calculated at the receiver nodes of a given box and the receiver nodes of its parent box (and so on for higher levels). The influence at the receiver nodes of a given parent box can be interpolated to the receiver nodes of the child box using the inverse process of parent anterpolation described in Section 3.3.1. This procedure is carried out for every box level until the base box level is

reached. Within the base boxes the interpolation matrix I_{mn} interpolates the influence at the probes from the receiver nodes. The calculated influence at the base boxes is hence composed of the superposition of the interpolation of the influences calculated at the base box level plus all higher levels and therewith the entire source field. It follows logically from Section 3.3.1 that increasing P improves the spatial approximation of the influence and therewith the accuracy of the MLMIC approximation. At the base box level, the desired field quantities are then calculated at the probe positions and stored for output.

Polynomial interpolation Interpolating polynomials and node positions must be chosen to ensure that spurious boundary values which generally occur during polynomial interpolation of a function on a closed domain are avoided. Following van Garrel [108], barycentric Lagrangian interpolation has been chosen due to its stability and ease of calculation [109]. This makes use of the following formula for an interpolating polynomial p at the position x between $n+1$ **known** values f_j at positions x_j:

$$p(x) = \sum_{j=0}^{n} \frac{w_j}{x - x_j} f_j \left(\sum_{j=0}^{n} \frac{w_j}{x - x_j} \right)^{-1} . \tag{3.5}$$

The barycentric weights w_j of the interpolation are defined by:

$$w_j = \frac{1}{\prod_{k \neq j} (x_j - x_k)} . \tag{3.6}$$

For certain choices of x_j, there exist explicit formulas for the weights w_j. The choice of equidistant nodes leads to large oscillations in the interpolant at the edge of the interpolation interval and should hence be avoided. For well-behaved interpolation one can choose *Chebyshev* nodes, chosen by projecting equally spaced points on the unit circle down to the unit interval $[-1, 1]$. Chebyshev points of the first kind and their corresponding barycentric weights are given by:

$$x_j = \cos \frac{(2j+1)\pi}{2n+2}, \quad j = 0, \ldots, n \quad \longrightarrow \quad w_j = (-1)^j \sin \frac{(2j+1)\pi}{2n+2} . \tag{3.7}$$

In the work here these are applied for the Green's method solver as the distribution of the influence is interpolated throughout the volume of a given receiver box– see Chapter 4. Chebyshev points of the **second** kind and their corresponding barycentric weights are given by:

$$x_j = \cos \frac{j\pi}{n}, \quad j = 0, \ldots, n \quad \longrightarrow \quad w_j = \begin{cases} 1/2(-1)^j & j = 0 \text{ or } n \\ (-1)^j & \text{otherwise} \end{cases} \tag{3.8}$$

In this case, interpolation nodes are also located on the boundary of a given domain. This is beneficial for the Poisson method solver as the distribution of the influence must be calculated on the boundary of the box (for specification of the stream function BC). The interpolating polynomials are illustrated in Fig. 3.6.

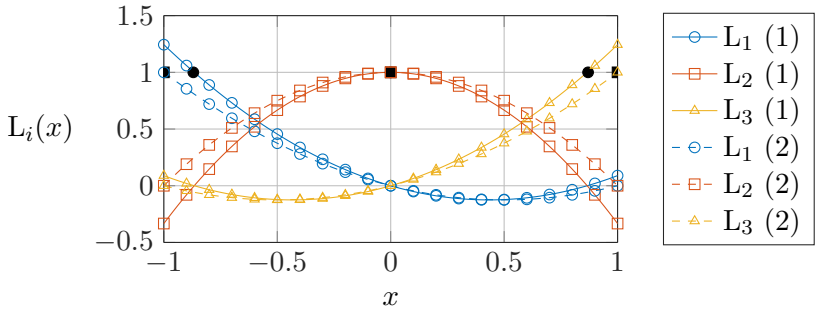

Figure 3.6: Basis polynomials for Lagrangian interpolation through Chebyshev nodes for interpolation order $P = 3$. The Chebyshev node type is in parenthesis in the legend. The intercepts of the polynomial are shown with circular (type 1) and square (type 2) nodes. It can be seen that type 2 interpolation extends to the boundary edges.

The choice of the P hence has a twofold effect on the accuracy solution by specifying:

1. The order of the spatial anterpolation of the source distribution (source boxes); and
2. The order of the spatial interpolation of the influence (receiver boxes)

It is shown in van Garrel [108] that the maximum error of this interpolation for smooth kernels is given by $|\epsilon_{\max}| = \mathcal{O}(H^P)$. It furthermore directly influences the computational expense of the far field integration, as the number of source/receiver nodes per box scales as P^3. The influence this has on the computational expense will be investigated in Chapter 5. The interpolation functions above are stated for a single dimension. For two or three dimensional interpolations, the tensor product is simply used: $p(x, y, z) = p(x)\, p(y)\, p(z)$.

The choice of barycentric Lagrangian interpolation has the advantage that the gradient of the interpolating function can also be pre-calculated based on the gradients of the interpolating polynomials. This procedure carries minimimal overhead and is carried out in the pre-processing step. In this case the maximum error is given by $|\epsilon_{\max}| = \mathcal{O}(H^{P-1})$. This is extensively described in Appendix F of van Garrel [108] and in Berrut & Trefethen

[109]. This has been used to calculate $\nabla\vec{u}$ in the Green's method solver described in Chapter 4.

Separation of near field and far field influence A distinction must be made at this stage for the behaviour of the kernel $\mathcal{K}$. In the case that the kernel does not become singular and has bounded higher-order derivatives, the approach described above is sufficient to calculate the interaction at every box level [108]. This however is often not the case. For the cases of interest here for example, the Biot-Savart and stream function kernels both behave singularly as the evaluation distance $\|\vec{x} - \vec{y}\|$ approaches zero, similarly to the velocity field induced by a potential flow vortex [1]. This implies that the maximum error of the multilevel approximation, given by $\mathcal{O}(\mathcal{K}^P H_b^P)$ [108], scales unfavorably. Below a certain cutoff distance, the approximation carried out in the MLMIC method is unsuitable and the influence should be calculated with direct evaluation. This region is controlled by specifying the size of H_b and the parameter NF_{int} to ensure that any source elements outside of this region are automatically captured by the multilevel scheme.

3.3.4 Summary of the MLMIC Procedure

Within the MLMIC routine the spatial integration of Eq. (3.2) is broken into two calculations steps. The entire particle set (containing sources and probes) is discretised into base boxes $\mathcal{B}_i$. At the beginning of the simulation, a template for the source and receiver node distributions is calculated, with the appropriate barycentric lagrangian positions and weights for the far field calculation. An interaction template is calculated based upon the kernel being investigated. The specific kernels applied here will be described in more detail in Chapter 4.

Far field influence The far field calculation is composed of three individual steps which must be chronologically executed:

1. **Anterpolation** A polynomial approximation of the source distribution at the base box level is carried out for all boxes containing source nodes. This source distribution is mapped to parent boxes. This mapping is repeated for higher levels until the maximum box level is reached (this is a modelling parameter and will be described in more detail in Chapter 4.
2. **Interaction** The interaction template is used to calculate the influence

at receiver nodes due to the anterpolated source strengths carried out in the previous step. Using appropriate scaling of the source strengths, this procedure can be reduced for each interaction template to a single, albeit large, matrix multiplication.

3. **Interpolation** The influence at parents box receiver nodes is interpolated to children boxes. This interpolation is repeated for lower levels until the base box level is reached. For all boxes containing probe nodes, an interpolation maps the far field influence to the probe positions.

Two modelling parameters influence the accuracy and computational complexity of the MLMIC routine: i) the order of the polynomial approximation P and ii) the near field parameter NF_{int}, which influences the size of the interaction template.

Near field influence Within an isolated region surrounding each base box, termed the **near field**, the polynomial approximation carried out in the far field step leads to unfavorable integration errors. This is avoided by directly evaluating the influence of sources within this region with the direct expressions, given in Appendix F. The size of the near field is specified by two modelling parameters: i) the base box side length H_b and ii) the near field parameter NF_{int}.

3.4 Application to the Vortex Particle Method

The VPM is an ideal application case of the MLMIC method, as both sources and probes both are treated as discrete points and the far field behaviour of the kernels can be well predicted. In the following sections the fundamentals for applying the MLMIC to the VPM will be described. A full description of the implementation of the MLMIC method and the VPM is given in Chapter 4. The interpolation quantities will be described initially, before the inspection of the two kernels of interest in this work are investigated. The first is the Biot-Savart (BS) kernel, which shall be resolved when using a Green's type solution method to determine the velocity field around a probe point. The second is the stream function (SF) kernel, this shall be used to calculate the value of the stream function on the boundary of a regular grid.

Source treatment The anterpolation as described requires a source strength to be mapped to the source nodes. As described in Section 3.3.1,

this can take arbitrary forms. For the treatment of a potential source for example, this is represented simply as a scalar- the source strength. A quadrupole for application to e.g. aeroacoustic problems [110] would require a tensor description. It is seen in the expressions for the velocity or stream function that the source distribution is represented by the circulation vector of each particle $\vec{\alpha}_p = \vec{\omega}_p \, dV_p$. This term is hence calculated for each source particle and this is mapped to the source nodes. This implies that particles volumes do not need not be equivalent, and it is not a requirement that the particle are regularly distributed in space, this can be advantageous for certain geometries [23]. The source vector is described as follows for 2D and 3D problems, respectively:

$$\vec{\alpha} = \begin{cases} \begin{bmatrix} 0 & 0 & \omega_z \end{bmatrix} \cdot dV_p & \text{if 2D} \\ \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix} \cdot dV_p & \text{if 3D} \end{cases} . \tag{3.9}$$

Probe treatment In the case that the problem is 2D, the outputs of the MLMIC integration are given by:

$$\mathcal{I}_{2D} = \begin{cases} \begin{bmatrix} u_x & u_y & 0 \end{bmatrix} & \text{Biot-Savart kernel} \\ \begin{bmatrix} 0 & 0 & \psi_z \end{bmatrix} & \text{Stream function kernel} \end{cases} \tag{3.10}$$

In the case that the problem is 3D, the outputs of the MLMIC integration are given by:

$$\mathcal{I}_{3D} = \begin{cases} \begin{bmatrix} u_x & u_y & u_z \end{bmatrix} & \text{Biot-Savart kernel} \\ \begin{bmatrix} \psi_x & \psi_y & \psi_z \end{bmatrix} & \text{Stream function kernel} \end{cases} \tag{3.11}$$

Regardless of the problem type or solution method, the application here has been formulated such that the influences calculated are of matrix data type for generality.

3.4.1 Behaviour of the Biot-Savart Kernel

For the application of the Green's method of solution the BS kernel is calculated with interaction templates as described earlier. This requires an analysis of the behaviour in the near and far field to ensure geometric scaling of the kernel. Inspecting Eq. (2.16), the velocity induced by a vortex particle can be expressed as:

$$\vec{u}_{\sigma,p} = K_\sigma(\vec{r}) \times \vec{\alpha}_p = -\frac{1}{4\pi r^3} q(\rho)(\vec{r} \times \vec{\alpha}_p) \, , \tag{3.12}$$

where $\vec{r} = \vec{y} - \vec{x}_p$ is the distance between source point $\vec{x}_p$ and evaluation (probe) point $\vec{y}$ and r is the norm of this distance. The function $q(\rho)$ is derived from the chosen regularisation $\zeta(\rho)$. The behaviour of this function for a range of common smoothing functions is shown in Fig. 3.7. It is seen

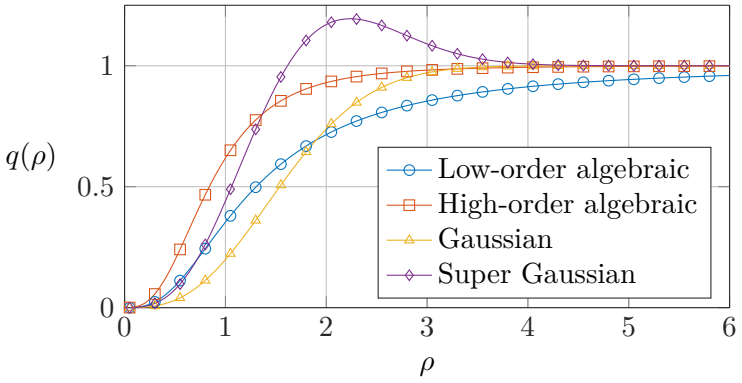

Figure 3.7: Biot-Savart smoothing for a range of common smoothing types.

that the smoothing factor $q(\rho)$ tends to unity for large ρ. This implies that for large ρ the influence behaves as the singular BS kernel:

$$\lim_{\rho\to\infty} \vec{u}_{\sigma,p} = -\frac{1}{4\pi r^3}(\vec{r} \times \vec{\alpha}_p)\,. \tag{3.13}$$

This is advantageous as the interaction $\vec{r}/r^3$ is easily implemented as an interaction template within the MLMIC method. This furthermore motivates specifying the minimum box size H_b as a multiple of the characteristic core size σ of the problem being investigated. This will be investigated further in the following section. It shall be demonstrated in Chapter 4 that this requires the calculation of three interaction templates, corresponding to $[r_x, r_y, r_z]\, r^{-3}$.

3.4.2 Behaviour of the Stream Function Kernel

An analogous inspection is carried out for the SF kernel. In this case the discrete contribution to the stream function due to a source particle is given by inspecting Eq. (2.15):

$$\vec{\Psi}_{\sigma,p} = \mathcal{G}_\sigma(\vec{x} - \vec{x}_p)\,\vec{\alpha}_p = \frac{1}{4\pi r} g(\rho)\,. \tag{3.14}$$

As with for the BS kernel, the function $g(\rho)$ is derived from the chosen smoothing function $\zeta(\rho)$. The behaviour of g for a range of common

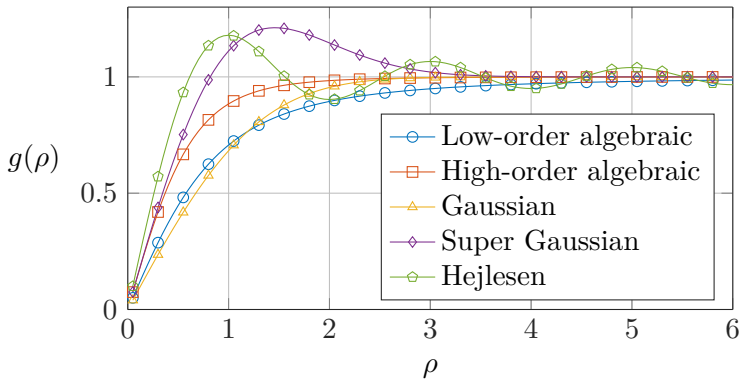

Figure 3.8: Stream function smoothing for a range of common smoothing types.

smoothing functions is shown in Fig. 3.8. Analogous behaviour to the BS kernels is seen. For large arguments the g function tends to unity:

$$\lim_{\rho\to\infty} \vec{\psi}_{\sigma,p} = \frac{1}{4\pi r}\vec{\alpha}_p \,. \tag{3.15}$$

This again is simple to express as an interaction template, however here only a single interaction is necessary: r^{-1}.

3.5 MLMIC Verification

The existence of explicit expressions for induced velocity, stretching and stream function of a source particle allow investigation into the performance of the MLMIC method. This is achieved by calculating the influence of a particle set with direct evaluation and comparing this to the result of integration with the MLMIC method. The two far field quantities which are resolved by the MLMIC method are the velocity $\vec{u}$ and the stream function $\vec{\psi}$. In addition, if using Green's method the calculation of the vorticity stretching terms $(\vec{\omega}\cdot\nabla)\vec{u}$ are required for the accurate prediction of vorticity evolution. This however is a secondary output of the MLMIC method and is hence handled subsequently to the verification of the velocity field. The ability of the MLMIC method to predict the near field influence is unnecessary, as these are directly evaluated within the solvers using the explicit expressions described in Chapter 2 and Appendix F.

3.5.1 Test Vorticity Distribution

A suitable flow field must be chosen to test the MLMIC method on. Cases of dense and more complicated vorticity distributions shall be handled in the solver validation presented in Chapter 5. Focus is made on a simple flow field demonstrating features representative of wake flows. A suitable flow field here is a helical vortex. The helical vortex filament has the following spatial parametrisation:

$$R\cos(2\pi\,t)\,\vec{e}_y + R\sin(2\pi\,t)\,\vec{e}_z + k\,t\,\vec{e}_x + R \quad \text{for:} \quad t \in [0,1]\,, \tag{3.16}$$

where R and k are the radius and length of the helix respectively. This describes a helix of pitch $2\pi\,k$. This is shown for a triple root-tip vortex filament set in Figure 3.9. For the triple-helix cases visualised here and investigated in the proceeding cases, $k = 4$ and $R = 1$. The vorticity field around the filament is described with a Gaussian distribution given by:

$$\vec{\omega}(r) = \frac{\Gamma}{2\pi a^2}\exp\left\{-\frac{r^2}{2a^2}\right\}\vec{e}_t\,, \tag{3.17}$$

where r is the radial distance to the vortex filament, a is the vortex core size and Γ is taken as unity. The unit vector $\vec{e}_t$ is tangent to the vortex filament. In the proceeding cases $a = 0.05\,\text{m}$ was chosen to represent a *thick* vortex core. This choice geometry includes a number of important features which are inherent to the vorticity distribution in the wake of a wind turbine. These include:

- **Geometry ratios** For this field the global characteristic length parameters (length-radius-core) mimic those of a realistic application;
- **Small scale interactions** The vortex core roughly mimics that of an actual tip vortex shed from a sharp-edged blade. Accuracy of the predictions must therefore also be modelled for local interactions at the scale of a;
- **Large scale interactions** The actions at larger separation distances $r > 5a$ must also be captured to ensure accurate global behaviour.

Generation of particle grid The vorticity distribution was generated on a regular grid with spacing H. The strength of the vortex particle was specified by initially generating a single vortex filament along the curve given by Eq. (3.16). It is observed in Fig. 3.10 that for $r > 5a$ the vorticity practically vanishes, so coordinates which satisfy this equality are essentially ignored and no vortex particles are generated. For a cubic region with

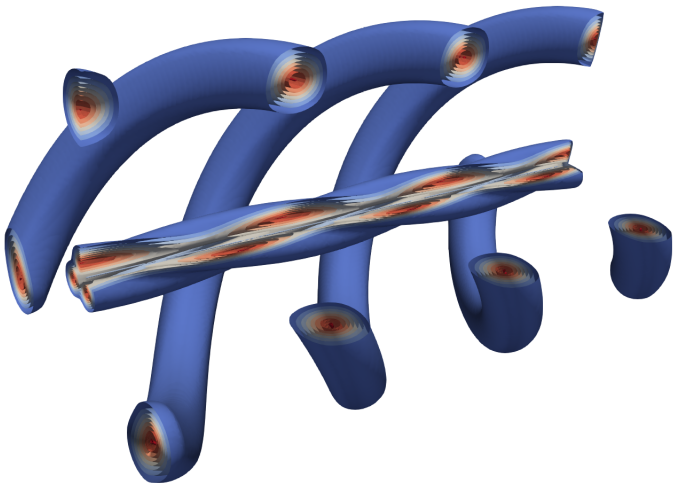

Figure 3.9: The helical vortex arrangement for a 3-bladed helical wake. This represents approximately the tip and root vortex of a wind turbine or propellor.

dimension $[5a, 5a, 5a]$ around each filament point, the convolution of the filament strength is carried out with Eq. (3.17). For overlapping convolution regions and multiple helices, the vortex strengths are simply superimposed. The field visualised in Fig. 3.9 was generated by creating this field and then

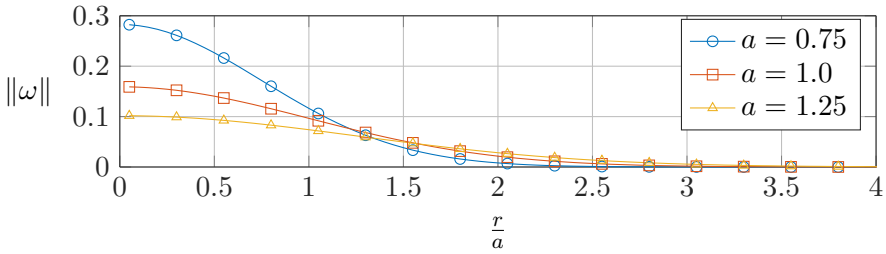

Figure 3.10: The decay of the Gaussian distribution given by Eq. (3.17) for a range of core sizes a.

extracting vorticity contour plots from the volume vorticity distribution.

3.5.2 Error Metric

The influence of the far field using direct evaluation with the explicit particle expressions shall be referred to as $\vec{I}_{\text{dir}}$. This allows the error of the predicted field with the MLMIC method $\vec{I}_{\text{ml}}$ to be quantified. The errors in these

tests are based on the continuous L_m-norm of a variable x over a region $\mathcal{R}$ which is defined as:

$$L_m(x) = \left(\frac{1}{V} \int_{\mathcal{R}} |x|^m \, dV \right)^{\frac{1}{m}} \quad \overset{\text{disc.}}{=} \quad \left(\frac{1}{n} \sum_{n \in \mathcal{R}} |x|^m \, dv_n \right)^{\frac{1}{m}}, \tag{3.18}$$

where $|\cdot|$ represents the absolute value for scalar functions and norm in the case of vectors. A normalized error metric is achieved by dividing the error term with $\mathrm{L}_2(\vec{I}_{\mathrm{dir}})$, resulting in the relative L_2-norm error ϵ:

$$\epsilon(x) = \frac{L_2(\vec{I}_{\mathrm{ml}}(x) - \vec{I}_{\mathrm{dir}}(x))}{L_2(\vec{I}_{\mathrm{dir}}(x))} \, . \tag{3.19}$$

The MLMIC solver has been configured in such a way that the near and far field influences– $\mathcal{I}_{\mathrm{nf}}$ and $\mathcal{I}_{\mathrm{ff}}$, respectively– are stored separately for testing purposes. This is necessary here as $\mathcal{I}_{\mathrm{nf}}$ generally dominates the global influence at a position in space. Including $\mathcal{I}_{\mathrm{nf}}$ in the calculation would hence bias the results in a positive sense, as the relative error would appear smaller.

3.5.3 Biot-Savart Kernel

The convolution with this kernel is being calculated by the MLMIC method when using Green's method in order to calculate the velocity field $\vec{u}$. It should be noted that a far field influence term is also calculated for the stretching term $d\vec{\omega}/dt$. This however is derived from the velocity field and is therewith a secondary output. This shall be described in the following sections.

Specification of minimum box size $\mathbf{H_b}$ The choice of minimum box size H_b must ensure that the far field influence is sufficiently approximated by the singular particle approximation. The distance at which this approximation is valid shall be hereafter referred to as the **cut-off** distance r_c, and is normalized with the core size σ of the source particle to give $\rho_c = r_c/\sigma$ as described previously. It is instructive to inspect the behaviour of the kernels of interest for different regularisations to provide a metric for a suitable cutoff distance. The higher the order of the regularisation function, the lower the error of the particle treatment. This naturally motivates the use of the highest-possible order regularisation. These however incur additional calculation expense and make expression of the stretching functions more complicated and error-prone. For this reason, three regularisations were investigated here. These are shown in Figure 3.7 for the BS kernel $q(\rho)$.

As can be seen the choice of regularisation function influences the cut-off distance. In order to quantify the error incurred by making this approximation for a given choice of r_c, a range of relative accuracies of the approximation: $\epsilon_\rho = 1 - g_\rho/g_{\text{sing}}$ is given in Table 3.1. It can be seen that the Gaussian

Table 3.1: Cut-off distance for a range of desired accuracies ϵ_ρ.

	ρ_c - Biot Savart			ρ_c - Stream function		
ϵ_ρ	LOA	HOA	Gaussian	LOA	HOA	Gaussian
10^{-1}	2.07	1.07	1.65	3.71	1.77	2.50
10^{-2}	7.02	2.30	2.58	12.20	3.54	3.37
10^{-3}	22.24	4.30	3.29	38.71	6.49	4.03
10^{-4}	70.71	7.77	3.89	122.47	11.65	4.59
10^{-5}	223.61	13.86	4.42	387.30	20.78	5.09
10^{-6}	707.11	24.73	4.89	1224.7	36.99	5.54

kernel requires the smallest cut-off distance. For this reason the Gaussian regularisation has been used in the following work, as this allows the near field region to be chosen smaller, therewith reducing the expense of the near field calculation. Furthermore, in the expansive literature on the VPM, the Gaussian kernel appears to be the most common form of smoothing and has an intuitive connection to realistic flow in that the vorticity distribution is that of a Lamb-Oseen vortex at a given time value [1]. The LOA and HOA regularisations are practical for reduction of computational overhead and for comparison with analytical results [22]. This is however irrelevant here as the choice of regularisation only affects the near field, in the far field all interactions are treated with the singular particle approximation, any results shown here are hence also valid for the LOA and HOA regularisations.

Relative contribution of far field influence In order to quantify from $\mathcal{I}$ the relative proportions of near field influence $\mathcal{I}_{\text{nf}}$ and far field influence $\mathcal{I}_{\text{ff}}$, these quantities have been inspected here. The portion $\chi_{\text{ff}} = \mathcal{I}_{\text{ff}}/\mathcal{I}$ is shown in Fig. 3.11 for the velocity and stretching fields, respectively. The results are shown for a range of particle spacing H and base box sizes H_b, normalized with respect to the characteristic core size σ. This was chosen so as to ensure particle overlap $\sigma_{char} = 1.1\,H$.

The first point to be observed is that, as expected, the contribution of the far-field decreases as H_b increases. This is because the portion of the total source field which is in the geometric far field (dictated by H_b) is decreased. The second noteworthy point is that in general the stretching term is dominated by the near field interaction. This is well explained with

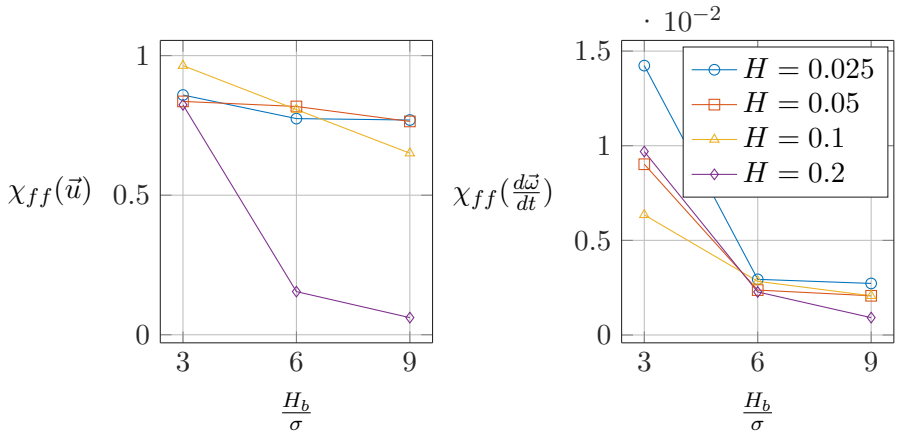

Figure 3.11: Far field contribution to velocity (left) and stretching term (right) for a range of grid parameters.

inspection of the BS kernel, which scales in the far field as r^{-3}, as opposed to the stretching kernel, which scales as r^{-5}– see Section 2.3.1. The far field contribution to the velocity field for the case $H = 0.2$ is seen to rapidly decrease for larger box sizes, this is because the far field is in this case geometrically smaller than the near field.

Accuracy of far field velocity influence In order to assess the accuracy of the MLMIC method in the calculation of the induced velocity from regularized vortex particles, the geometry described above has been used with the grid sizes $H = 0.2, 0.1, 0.05$ and $0.025\,\mathrm{m}$, these have particle counts N between approximately 10^3 ($0.2\,\mathrm{m}$) and 10^6 particles ($0.025\,\mathrm{m}$). Three choices of minimum box size H_b have been inspected. The minimum box size ensures the minimum distance where the singular particle approximation is employed by the MLMIC solver, and hence it is expected that the accuracy decreases as this is reduced. For the coarsest case ($H = 0.2\,\mathrm{m}$, $H_b = 9\,\sigma$) the particle distribution is sufficiently dense that the MLMIC routine only extends up to grid level $l = 1$. In comparison to this, for the finest case ($H = 0.025\,\mathrm{m}$, $H_b = 3\,\sigma$) the MLMIC routine must extend up to box level $l = 5$ in order to capture the entire source field. The results are shown in Fig. 3.12. For the simulations carried out here a Gaussian regularisation has been used with a characteristic particle core size which ensures particle overlap $\sigma_{char} = 1.1\,H$.

It is observed in Fig. 3.7 and Table 3.1 that, in general, for $\rho < 5$ the particle influence is not well predicted by the singular particle approximation, this is observed in Fig. 3.12 as for all cases where $H_b = 3$, the error asymptotes

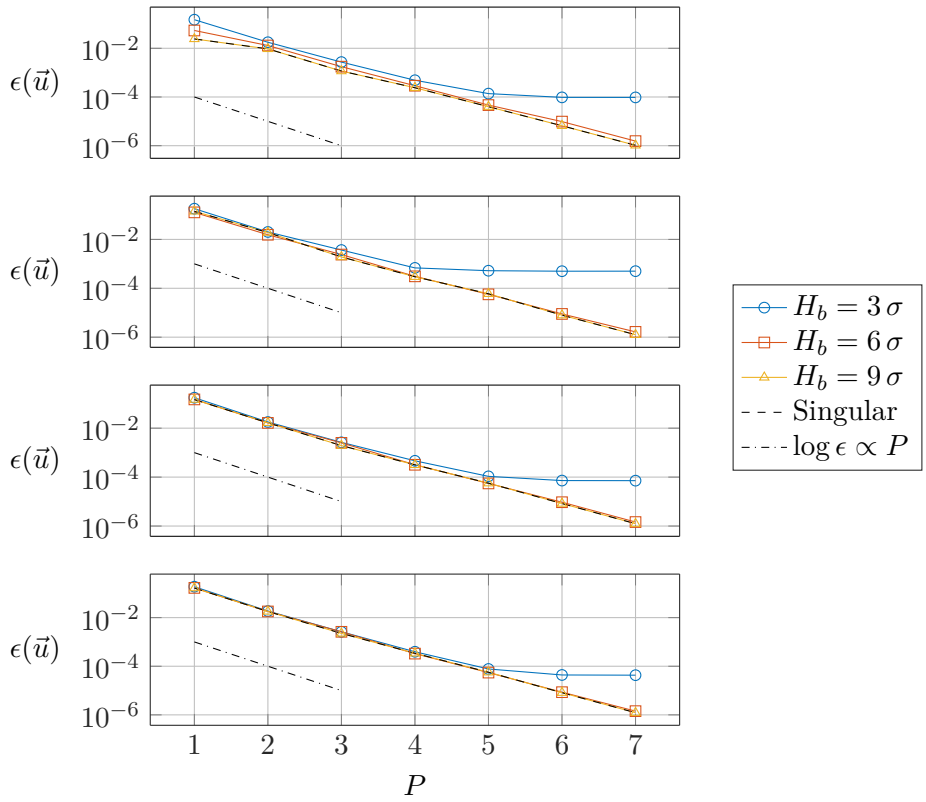

Figure 3.12: L_2-norm error induced by using the multilevel approximation of the velocity field compared to direct evaluation. From top to bottom the grid sizes are H = 0.2, 0.1, 0.05 and 0.025 m.

beyond a certain polynomial approximation P. Van Garrel describes in [108] that the maximum error $\epsilon_{\max}$ for interpolation of kernels with bounded higher order derivatives is proportional to $\mathcal{O}(H_b^P)$. This implies that for a fixed box size, it is expected that $\log(\epsilon_{\max}) \propto P$. This behaviour is marked with a dashed line in the plots shown in Fig. 3.12 for reference and it can be seen that this indeed is seen to be the case. The results for all grid sizes demonstrate that, provided the minimum box size is chosen adequately, the scaling of the error can be well controlled by specifying P as desired. Furthermore it can be seen that provided $H_b \geq 6\sigma$, the error can be reduced to an arbitrarily low value by specifying higher values of P. Similar results are observed for other regularisation functions such as LOA or HOA, however the minimum box size H_b must then be adapted as per Table 3.1.

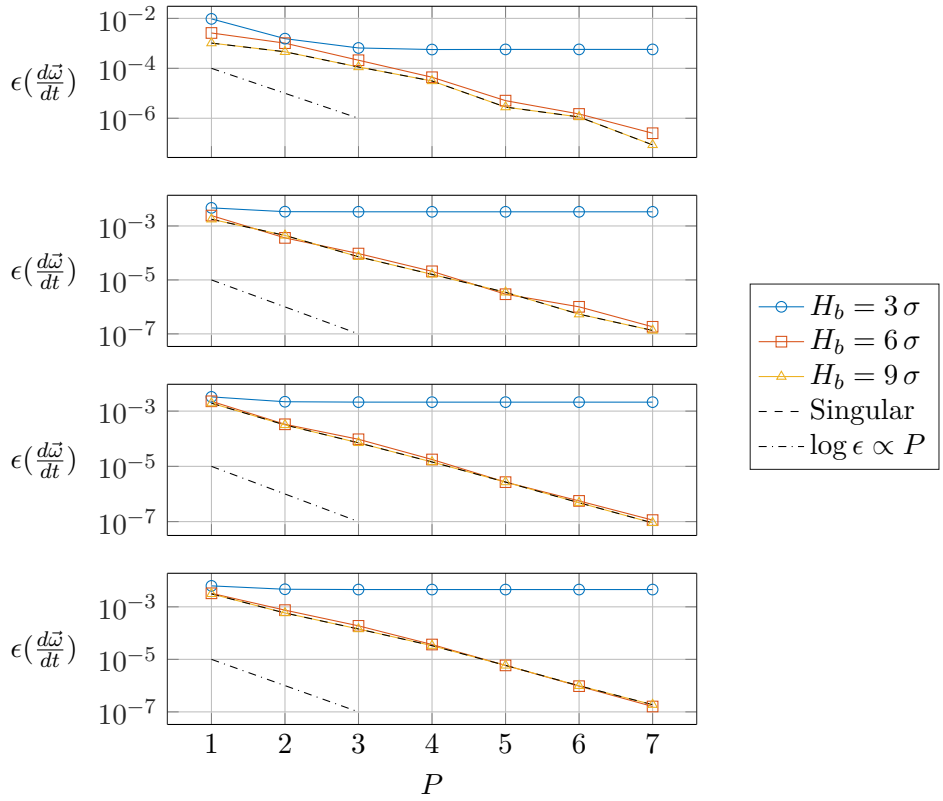

Figure 3.13: L_2-norm error induced by using the multilevel approximation of the stretching field compared to direct evaluation. From top to bottom the grid sizes are H = 0.2, 0.1, 0.05 and 0.025 m. The error for the case of no regularisation (singular particles) are also shown, and are seen to be practically equivalent to regularised particles.

Accuracy of far field vortex stretching An equivalent calculation is carried out to determine the error incurred in the stretching term when using the Green's method solver. A full description of the implementation is given in Chapter 4, a brief description is nonetheless given here in order to facilitate analysis. Although it is possible to formulate the far field stretching term as an interaction template, this greatly increases the number of templates required and the computational complexity increases becomes impractical. It is seen in Fig. 3.11 that the far field contribution to the stretching term is in general not dominant and hence a further approximation may be suitable. The stretching term requires the calculation of the velocity gradient tensor $\nabla\vec{u}$, calculated here using the polynomial approximation inherent to barycentric Lagrangian interpolation as described in Section 3.3.3. The relative L_2-norm error $\epsilon(d\vec{\omega}/dt)$ for a range of grid sizes H

and base box sizes H_b is given in Fig. 3.13. For these tests the transpose stretching scheme has been used.

Similar results are observed as with the BS kernel. The errors incurred by choosing an insufficient minimum box size H_b are however more marked. This is to be expected as the near field contribution dominates. As described previously, an additional error is incurred due to the polynomial interpolation of the gradient of the velocity field. The maximum error should here scale as $\log(\epsilon_{\max}) \propto (P-1) \propto P$, this has been shown as a dashed line in Fig. 3.13 and appears to approximately describe the accuracy. These results demonstrate again that, provided a suitable choice of H_b is made, the error can be arbitrarily specified with P. Equivalent behaviour was been observed for both classic and mixed stretching schemes. Similar behaviour was also observed for other regularisation schemes.

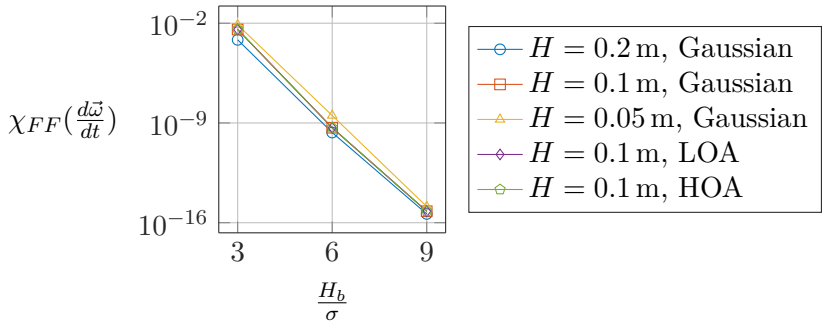

Figure 3.14: The relative contribution due to the far field influence on the viscous diffusion term for a range of regularisations and grid sizes. The results demonstrate that for large H_b, the far field contributes negligibly. The error for the case of no regularisation (singular particles) are also shown, and are seen to be practically equivalent to regularised particles.

Near field viscous diffusion When solving with Green's method, viscous diffusion is treated with the PSE scheme as described in Chapter 2. This makes use of a diffusion function η_σ which decays rapidly with ρ. For e.g. the Gaussian regularisation the function decays as $\exp\{-\rho^2\}$. This implies that, the contribution of the far field (under the assumption of a somewhat uniform source distribution) is negligible. This has been investigated by evaluating the far field contribution of the viscous diffusion term, this is defined analogously to χ_{ff} above. The results are displayed in Fig. 3.14 for a range of grid resolutions and regularisation types.

It can be seen how the influence greatly decreases with increasing H_b due to the behaviour of the PSE kernel. This implies that, provided H_b is chosen

large enough, the far field contribution of the viscous diffusion can effectively be neglected with negligible consequences to the accuracy of the solution. In parallel with the results shown above, it is again clear that the error of ignoring the far field contribution to diffusion can be reduced arbitrarily with the parameter H_b.

3.5.4 Stream Function Kernel

When using the Poisson method solver it is necessary to resolve the stream function of a particle set in order to specify boundary conditions on the Eulerian grid. This calculation proceeds exactly as with the BS kernel, however the SF kernel is somewhat simpler.

Specification of minimum box size $\mathbf{H_b}$ Analog to the procedure carried out for the BS kernel, the box size must be specified such that any particles in the far field are well approximated by the singular particle representation. This is dictated by the $g(\rho)$ function as illustrated in Fig. 3.8. The accuracy of the approximation is summarised in Table 3.1. Here the behaviour can be seen to be seen to be quite comparable to the BS kernel, however in general a slightly larger cutoff distance is required for a given accuracy when approximating with the singular kernel. In practice the core size is not chosen by this constraint, but rather by specifying that the calculated stream function from the particle representation agrees with that predicted by the fast Poisson (FP) solver. This is done by comparing the stream function for a unit strength particle between the two approaches, and specifying the core size σ such that they coincide. This allows specification of the characteristic coresize σ_{char} as a function of grid resolution H and is detailed further in Chapter 4.

Relative error limits The relative contribution due to the far field of the stream function is again inspected here and summarized with the parameter χ_{ff}, defined as above. The results for a range of grid sizes H and base box sizes H_b are shown in Fig. 3.15. As opposed to the case of the BS kernel, for the Poisson solver generally it is desired to increase the FP domain size (base box size H_b– see Chapter 4) in order to improve efficiency. For this reason, values of H_b up to $40\,\sigma$ have been investigated here. For the BS kernel this would be unnecessarily large and greatly reduce performance.

It is seen also here that increasing H_b in general decreases the contribution to the far field, again completely explained in terms of the geometric size of the near field. The relationship is seen to be well approximated as a linear

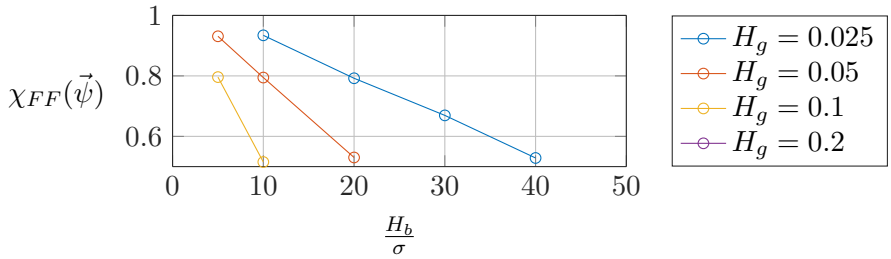

Figure 3.15: Far field contribution to the stream function. The error for the case of no regularisation (singular particles) are also shown, and are seen to be practically equivalent to regularised particles.

function of H_b. In comparison to Fig. 3.11, it can generally be observed that the far field contributes slightly more to the SF kernel than to the BS kernel. This is an intuitive result as the BS kernel scales as r^{-3} whereas the SF kernel scales as r^{-1}, implying that for a continuous source distribution the far field contribution dominates.

Far field stream function An equivalent procedure was used to verify the far field influence of the SF kernel as carried out above for the BS kernel. The relative error of using the MLMIC method to calculate the SF kernel for grid sizes $H = 0.2$, 0.1, 0.05 and 0.025 m and three base box sizes H_b are given in Fig. 3.16.

Essentially equivalent behaviour is seen here as with the BS kernel, however the results appear to be less sensitive to the choice of H_b. In fact, for the highly resolved case $H = 0.025\,\text{m}$, it appears that the error can be completely controlled by P. Again, the error can be controlled for a given box size H_b by specifying the polynomial order P. A choice of $H_b \geq 6\,\sigma$ appears to produce perfectly satisfactory results.

3.5.5 Summary

It has been shown that if applying the MLMIC method, the integration error for both kernels of interest in this work:

1. The BS kernel for application with the Green's method solver, and;
2. The SF kernel for the Poisson method solver,

can be reduced to an arbitrarily small value by appropriately specifying the polynomial order of interpolation P. Provided the minimum box size H_b

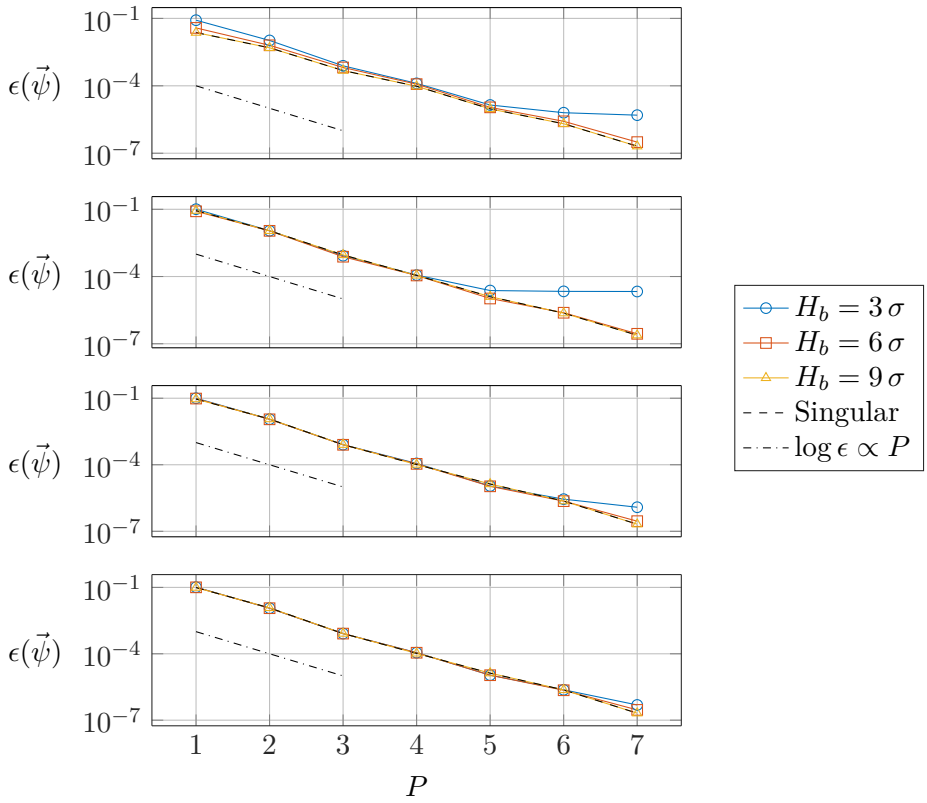

Figure 3.16: L_2-norm error induced by using the multilevel approximation of the stream function field compared to direct evaluation. From top to bottom the grid sizes are H = 0.2, 0.1, 0.05 and 0.025 m.

is specified to be large enough that the singular particle representation of the far field is satisfied (see Table 3.1), the maximum relative error $\epsilon_{\max}$ scales as $\log(\epsilon_{\max}) \propto P$. Higher values of P lead to higher computational complexity of the MLMIC method due to:

1. Increased order of the polynomial approximation of the source distribution within source boxes,
2. Increased order of the polynomial approximation of the influence within receiver boxes, and
3. Increased size of the interaction templates,

it will however be shown in Chapter 5 that this expense is warranted, as the computational complexity is indeed reduced from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$.

Chapter 4 Flow Solver Implementation

Chapters 2 and 3 described the underlying theories of the VPM the MLMIC methods. This chapter presents the implementation of these concepts in the solver developed in this work. The software shall hereafter be referred to as the **v**ortex **p**article **m**ulti**l**evel library or `VPML`. Within the `VPML` library two solvers have been implemented:

- **`GML` Solver** A *particle-particle* solver making use of Green's functions to resolve the field quantities.
- **`PML` Solver** A *particle-mesh* solver making use of the FP solver and FD to resolve field quantities.

Although the two approaches to solving the field equations are conceptually quite different, the implementation has been carried out such that both solvers make use of common data types and have a common library architecture. The library is written in the object-oriented language `C++` and makes extensive use of the concepts of polymorphism and inheritance, as shall be later described. The solver makes frequent use of matrix and vector operations, for this the linear algebra library `Eigen` [111] has been extensively employed. All visualisations have been provided for 2D systems for clear representation of the concepts, however in practice the application has been predominantly applied to 3D problems.

Optimisations For cases where large matrix multiplications are necessary such as template interactions (Section 3.3.2) or for the James-Lackner algorithm (Section 4.4), the procedure has been carried out on the graphical processing unit (GPU). This task is much more amenable to GPU calculation and furthermore there already exists optimized linear algebra libraries. This has been accomplished with both the open-source library `OpenCLBlast` [112] and the proprietary library `cuBLAS` from NVIDIA [113]. Another process which can be highly parallelised are near field calculations for the Greens solver, in this case the `OpenCL` framework [114] has been used for preparing kernels for parallelised execution on the GPU. Finally the shared-memory multi-processing platform `OpenMP` [115] has been employed to carry out parallel processing operations on the CPU, such a discrete box operations (see Section 4.1.2).

Visualisations Within the simple user-interface of `VPML` for purposes of code checking a simple particle-grid visualisation has been prepared with the `OpenGL` library. For detailed visualisations presented in this thesis an export functionality was prepared which generates a node-cell representation with cell connectivity in `.vtk` file format for use in the visualisation tool `ParaView` [116].

Particle Treatment The field at any time step is described entirely in terms of a set of discrete vortex particles. These are represented most conveniently as an `Eigen` vector type, $\vec{P}$:

$$\vec{P} = \Big[\underbrace{x \quad y \quad z}_{Position} \quad \underbrace{\omega_x \quad \omega_y \quad \omega_z}_{Vorticity} \quad \underbrace{\sigma}_{Core\,size} \quad \underbrace{dV}_{Volume} \Big] . \tag{4.1}$$

Particles are stored in a dynamic array which is updated (modified, appended to or truncated) during a simulation. This representation has numerous benefits. Export of a particle set is achieved simply by storing the data in row-format in a standard `.dat` file. For cases where matrix operations are to be performed from the particle data, the necessary elements can easily be extracted from the vector, or for multiple particles concatenated to a matrix object.

4.1 Grid Definition

For both solvers, an underlying grid is required for both remeshing and divergence filtering of the particle set. This grid makes use of a nested volume (box $\mathcal{B}$) description to account for particle- and probe-containing regions.

Three grid parameters are introduced here:

- **Characteristic Grid Size** H The spatial resolution which dictates characteristic particle spacing.
- **Base box sidelength** H_b The flow domain is discretised into boxes $\mathcal{B}_i$ with side length specified as a multiple of H: $H_b = n\,H$.
- **Cartesian ID** CID Specifies the grid index locations with bracket notation $\{i_x, i_y, i_z\}$. This is used to describe both box and cell positions.

These parameters are visualised in Fig. 4.1. The hierarchical description of the volume implies that larger boxes contain smaller boxes, this allows

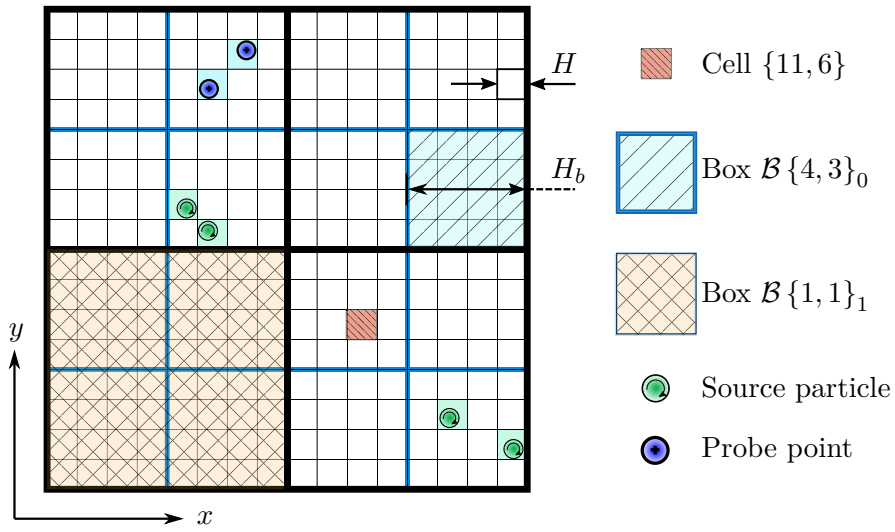

Figure 4.1: Characteristic objects and indices used to describe the grid in the `VPML` library. The volumes are described in a hierarchical sense: Cell $\{11,6\} \in$ Box $\mathcal{B}\{3,1\}_0 \in$ Box $\mathcal{B}\{2,1\}_1$ etc. For the case here, $H_b = 4\,H$, so that each box is discretised into four cells in each spatial direction.

simple application of the MLMIC concept described in Chapter 3. For box descriptions the box level is denoted with the subscript l. Boxes at level 0 are denoted as *base boxes* $\mathcal{B}\{\cdot,\cdot,\cdot\}_0$. These grid definitions allow the specification of two further grid-dependent parameters.

- **Characteristic Volume** dV Specifies the volume occupied by the particle within the mesh and is equivalent to the characteristic cell volume $dV_{2d} = H^2$, $dV_{3d} = H^3$.
- **Characteristic Core Size** σ Specifies the core size parameter for particle regularisation: Eq. (2.14). The specification of this parameter depends on the solver type and is described later.

4.1.1 Box Octtree Data Structure

Particle storage and data access with box objects implies the necessity of a framework to create, access and edit the box objects. There exists numerous methodologies which could be employed to this end. One option is to generate a rectangular box region to describe the entire domain. This is the most straightforward approach in terms of creation and access, however it generally leads to creation of boxes which for the duration of a simulation remain unused, hence consuming unnecessary runtime memory. Furthermore,

as resolution increases, this method of storage leads to a larger memory overhead due to the scaling (cubic in 3D or quadratic in 2D) of the domain volume. This option is activated within the solver with the `BLOCK` grid option.

A second approach has been taken here which generates only boxes which are required, and which allows optimised access and generation speed. This has been accomplished with an octtree data structure [117] and is activated with the `TREE` grid option. Within the octtree data structure space is divided into octants, and the octant index refers to a given region in space. Any volume region is then conveniently described simply with a list of octant IDs (O_l), called a *tree id*: $T_{\mathrm{ID}} = \{O_{l1}, O_{l2}...\}$. This is illustrated for the 2D analog (quadtree) in Fig. 4.2.

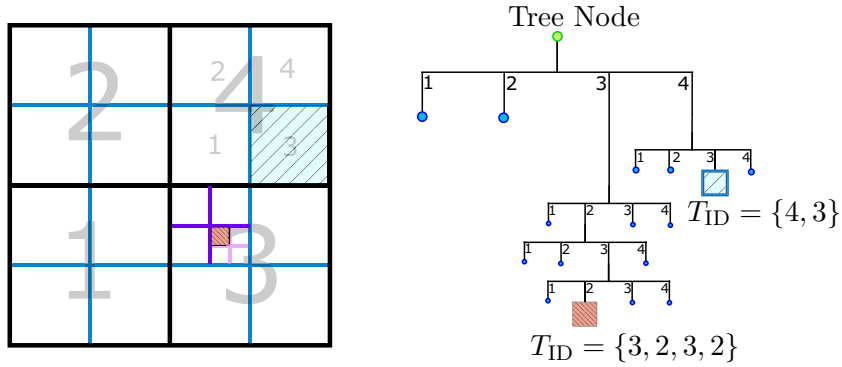

Figure 4.2: Left: A visualisation of spatial discretisation with a quadtree. Right: The heirarchical data structure along with the tree IDs for the two boxes of interest.

Analog to the grid box hierarchy described previously (Fig. 4.1), the volume regions are nested within this data structure and access to the next higher/lower grid object is achieved by a simple translation of the tree id. The functionality of the tree index is encapsulated within the `Leaf_Node` class, a template class which accepts any object for storage in a given leaf position. The relative position and indexing of the objects is inherently guaranteed by the leaf-tree storage structure, illustrated in Fig. 4.2. If a box contains smaller box octants, it is referred to as a *parent* box. Equivalently, boxes stored within a parent box are naturally referred to as *children* boxes. The leaf objects of the tree themselves consume minimal memory overhead, and ensure that access to the objects is easily achieved through the tree node (see Fig. 4.2) and the desired tree ID. The octtree is not thread-safe, implying parallel access is not possible without risking a data race, where multiple processors attempt to access the same data object concurrently.

The overheads of tree access are however negligible in comparison to other calculation steps, this is further detailed in Chapter 5.

4.1.2 Grid Activity & Proactivity

A distinction is made here between vortex particles which have a nonzero vorticity and therewith have an influence on the flow field (**source** points), and points at which the field quantitities are desired (**probe** points). For the majority of cases investigated in the work here, the field quantities are desired directly at the source particle positions, in this case source points and probe points coincide. This however is not the most general case, and for generality it is much more desirable to differentiate the two. This also has important consequences for the application and efficiency of the MLMIC method. The following box descriptors are defined:

Active boxes Boxes which contain source points are flagged as being **active**. The source density is anterpolated onto the source nodes and the boxes actively contribute to the multilevel expansion. For example, in Fig. 4.1 base boxes $\mathcal{B}\{4,1\}_0$ and $\mathcal{B}\{2,3\}_0$ are active. Corresponding to Fig. 4.2 these boxes have T_{ID} $\{3,3\}$ and $\{2,3\}$, respectively.

Proactive boxes Boxes which contain probe points are flagged as being **proactive** (probe-active). The field quantities are calculated **only** within proactive boxes. For example, in Fig. 4.1 only base box $\mathcal{B}\{2,4\}_0$ is proactive. Corresponding to Fig. 4.2 this box has T_{ID} $\{2,4\}$.

4.1.3 Binning

At the beginning of the calculation for a particle set, the source points are *binned* into their respective base boxes, which are marked as **active**. The active base boxes are then stored in a temporary array. An equivalent procedure is carried out for the probe nodes with proactive boxes. By doing this, the entire particle interaction problem is transformed from a particle-particle interaction problem to a box-box interaction problem. This has the enormous computational advantage that box-local procedures (e.g. for the `PML` solver grid mapping or for the `GML` solver influence interpolation) can be carried out in parallel. This will be detailed in Chapter 5.

4.1.4 MLMIC Tree Structure

After binning has been completed, the box tree structure for use in the MLMIC method must be created. For each active box, the parent box is found, which is marked as `ML_Active` and added to an MLMIC branch list. The same procedure is carried out for proactive boxes, which are marked as `ML_Proactive`. This process is repeated for higher box levels by identifying parents boxes with the octtree structure. The progression to higher box levels terminates based on a user-specified option:

- **MaxBranch Option** A specified maximum tree level l is considered in the ML algorithm, specified as `MaxBranch`. This essentially restricts the radius around a given proactive box which contributes to the solution. This may be practical in cases where the influence of sources beyond a given Euclidean distance can be assumed to negligibly contribute to the solution:
- **Automatic Option** The entire active field is accounted for in the MLMIC algorithm This option ensures that the influence of all sources in the field are accounted for. This has been used for all simulations in this work.

Upon construction of the MLMIC tree, the interaction templates are used to calculate the influences between `ML_Active` boxes and receiver nodes in `ML_Proactive` boxes. For all `ML_Proactive` boxes the influence at higher grid levels are interpolated down the tree to lower grid levels and added with local MLMIC interactions. This is repeated at each grid level until the base box level is reached, where the far field influences are added to the near field influences for the full influence. This approach ensures that only active (source) boxes are anterpolated and the influence is calculated only where it is required in proactive (probe) boxes automatically. This reduces computational expense and ensures generality for both solver configurations.

4.1.5 Eulerian Grid Template

Each base box is meshed with a regular Eulerian grid. In the `GML` solver this is used exclusively for remeshing and divergence filtering, within the `PML` solver however the grid contributes actively to the solution, as this provides the data for the FP solver. The mapping procedure distributes the vorticity over a certain grid stencil width, this implies that box grids overlap. Without accounting for this within the mapping, large discontinuities at box boundaries would be observed and spurious errors in the solution would

arise. This is overcome by constructing the Eulerian grid within each box with an overlap factor N_+, which describes the number of cells which extend beyond the box domain– see Fig. 4.3.

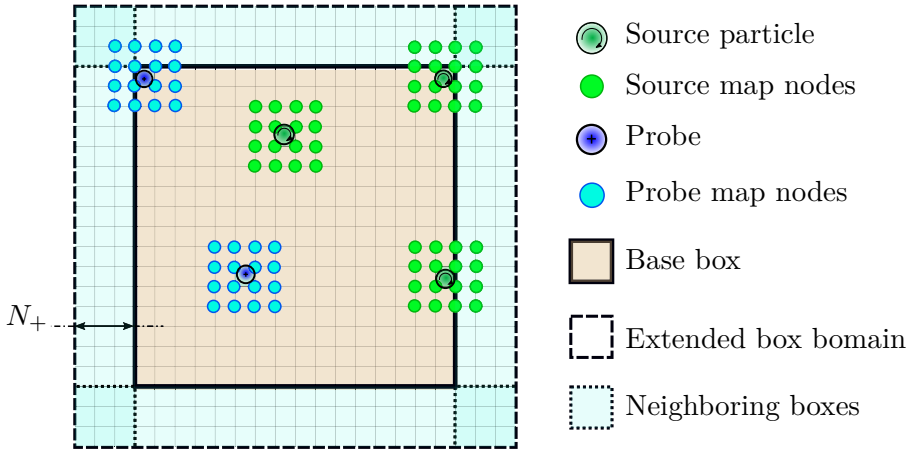

Figure 4.3: Overlapping Eulerian grids ensure that mapping occurs consistently with neighboring domains. The mapping stencil here corresponds to the M'_4 mapping routine (see Section 2.4.1). For the example here the box side length $H_b = 16\,H$.

This principle applies not only for vorticity mapping to the grid, but also for mapping field quantities from the grid back to probe nodes. Although this increases the memory footprint of the base boxes, it is necessary to ensure continuity between domains. Care must be taken to ensure that the local grid `CID` is correctly calculated for transfers to and from Eulerian grid objects. Upon completion of the vorticity mapping within each base box, an additional global routine ensures that grid values are continuous by superimposing overlap regions.

4.1.6 Problems Exhibiting Symmetry

There are many cases which allow symmetries of the flow field to be exploited. Such cases are easily handled within the solver by specifying a plane of symmetry, this is illustrated for the case of an airfoil in Fig. 4.4, where the plane $y = 0$ acts as the symmetry plane. After particle binning, all active boxes are duplicated such that the mirrored box $\mathcal{B}_M$ is created at the mirror position about the symmetry plane. An example is given here for a case with y-symmetry:

$$\mathcal{B}\{i,j,k\}_0 \overset{\text{y symm.}}{\longrightarrow} \mathcal{B}\{i,-j,k\}_{0,M}\ . \tag{4.2}$$

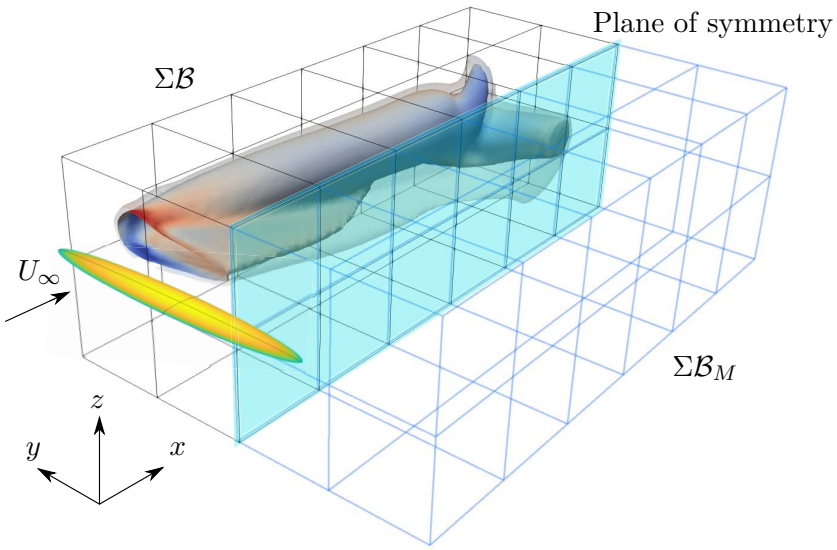

Figure 4.4: The wake of a symmetric airfoil modelled using a symmetry condition at the plane $y = 0$. Blue boxes represents the position of the reflected boxes $\mathcal{B}_M$.

To ensure that the influence is correctly mirrored, all source particles within the mirror box are also mirrored- both position and vorticity vector:

$$\vec{P}(x, y, z, \omega_x, \omega_y, \omega_z) \overset{\text{y symm.}}{\longrightarrow} \vec{P}_M(x, -y, z, -\omega_x, \omega_y, -\omega_z)\,. \tag{4.3}$$

The mirrored boxes for the aforementioned case are visualised in Fig. 4.4. This approach has the computational advantage that field quantities only need to be calculated on one side of the symmetry plane due to the known symmetry of the solution. The number of probe points and therewith computational expense is hence halved for each symmetry plane. Care is taken after particle evolution to ensure that any particles which translate through the symmetry plane are immediately replaced with their mirrored counterpart to ensure that no erroneous reflection errors occur.

4.1.7 Problems Exhibiting Periodicity

The method applied here to treat periodic problems is not significantly different to that applied for symmetry planes. Reference is made here to Fig. 4.5. Let $\mathcal{D}$ be the domain to be repeated in the periodic direction. One observes that the influence of source particles in domain $\mathcal{D}$ on domain $\mathcal{D}_{+1}$ is equivalent to the influence of source particles in domain $\mathcal{D}_{-1}$ on domain $\mathcal{D}$. The same is true for $\mathcal{D}$ on $\mathcal{D}_{+2}$ and $\mathcal{D}_{-2}$ on $\mathcal{D}$. This implies that only the source box $\mathcal{D}$ need be treated as **active**, neighboring regions however

are treated as **proactive**. The influence on domains $\mathcal{D}_{\pm}$ are simply added to $\mathcal{D}$ in order to calculate the periodic influence. Provided that the periodic length L_{per} is chosen such that it is an integer multiple of the box side length H_b, this is also perfectly amenable to analysis with box-box interactions.

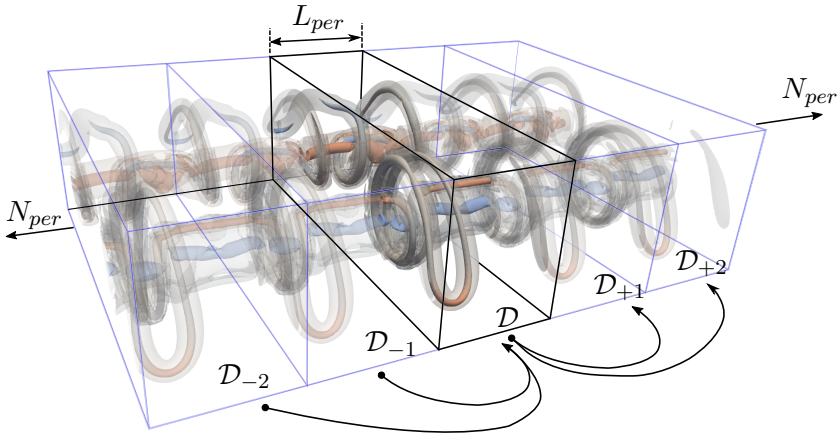

Figure 4.5: Method of treatment for periodic problems. The source domain is repeated in the periodic direction and neighboring sections are proactive.

The number of repeated domains on each side N_{per} is user-defined and chosen to ensure that the true periodic flow is captured. It is suggested in Cocle et al [28] that $N_{per} = 50$ should guarantee practically periodic solutions, however in the work here $N_{per} = 30$ appeared to be completely sufficient. As with the case of symmetric conditions, care must be taken to ensure that any particles which convect out of $\mathcal{D}$ are replaced by their equivalent particles within $\mathcal{D}$. It is observed here that only the single domain $\mathcal{D}$ is **active**, this significantly decreases the calculation overhead, as the majority of interactions calculated are in the far field and are hence captured by the MLMIC calculation.

4.2 Time Evolution of Particle Set

`VPML` is formulated such that regardless of the choice of solver, the spatial quantities of interest are resolved at the given particle/probe positions. These are then used to specify particle evolution with a suitable time integration scheme. The reader is referred to Kreyszig [86] for an overview of the schemes described here. For purposes of simple integration testing a first-order *Eulerian-forward* (`EF`) scheme has been implemented. For

function $\vec{f}$ at timestep n this is expressed as:

$$\vec{f}\,\Big|_{n+1} = \vec{f}\,\Big|_{n} + \Delta t \left.\frac{d\vec{f}}{dt}\right|_{n} . \tag{4.4}$$

For higher-order time integration the second and fourth-order *Runge-Kutta* schemes have been implemented (`RK2`,`RK4`). The second-order scheme is given here:

$$\vec{f}\,\Big|_{n+1} = \vec{f}\,\Big|_{n} + \frac{1}{2}\Delta t \left\{ \left.\frac{d\vec{f}}{dt}\right|_{n} + \left.\frac{d\vec{f}}{dt}\right|_{n^+} \right\} , \tag{4.5}$$

where the n^+ is a corrector step taken at $t = t^n + 0.5\,\Delta t$. As with spatial integration a balance must always be struck between accuracy and computational speed. Despite the stability of the `RK` methods, they require the calculation of predictor steps to achieve the higher order accuracy. In the case of `RK4` for example three additional intermediate steps are required for a single time step. For this reason the second-order multi-step Adams-Bashforth method (`AB2`) has also been implemented. This allows the gradients from previous timesteps to be utilised, incurring minimal memory overhead while recovering higher order accuracy. The Adams-Bashforth second order method is given by:

$$\vec{f}\,\Big|_{n+1} = \vec{f}\,\Big|_{n} + \Delta t \left\{ \frac{3}{2} \left.\frac{d\vec{f}}{dt}\right|_{n} - \frac{1}{2} \left.\frac{d\vec{f}}{dt}\right|_{n-1} \right\} . \tag{4.6}$$

It has been shown furthermore that leapfrog methods (`LF`), which makes use of staggered position-velocity data to perform timestepping are well suited to particle convection problems [28]. The second-order leap frog method is given by:

$$\vec{f}\,\Big|_{n+1} = \vec{f}\,\Big|_{n-1} + 2\Delta t \left.\frac{d\vec{f}}{dt}\right|_{n} . \tag{4.7}$$

The method most frequently applied to the simulations carried out in this work combines `LF` for particle position update with `AB2` for particle strength update. This combination is named `AB2LF`.

4.3 `GML` Solver

The `GML` solver calculates particle-particle interaction using the theory outlined in Chapter 2, Section 2.3. The resolution of field quantities due

to source contributions in both far field and near field shall be separately outlined here for the sake of clarity. Upon their calculation the two terms are added to yield the full influence on each probe.

Far Field

The simulation flag `SPAT_INT` specifies which method is used to spatially integrate the particle set. Three options are available within the `VPML` library:

- `DIRECT` The direct expressions (Appendix F) are used and evaluated in parallel with `OpenMP` on the CPU,
- `DIRECT_OCL` The direct expressions are used and evaluated in parallel with `OpenCL` on the GPU,
- `MULTILEVEL` Interactions are calculated with the MLMIC routine as described in Chapter 3.

The following sections describe how individual terms are calculated when the spatial integration option `MULTILEVEL` has been activated.

Velocity Inspection of the BS kernel (Eq. (2.11)) allows the velocity terms to be deconstructed:

$$\begin{aligned} 4\pi\,\vec{u} = \frac{\vec{r}}{r^3} \times \vec{\alpha} = &\left\{ \left[\frac{r_y}{r^3}\right] \alpha_z - \left[\frac{r_z}{r^3}\right] \alpha_y \right\} \vec{e}_x \\ &+ \left\{ \left[\frac{r_z}{r^3}\right] \alpha_x - \left[\frac{r_x}{r^3}\right] \alpha_z \right\} \vec{e}_y \\ &+ \left\{ \left[\frac{r_x}{r^3}\right] \alpha_y - \left[\frac{r_y}{r^3}\right] \alpha_x \right\} \vec{e}_z \,. \end{aligned} \tag{4.8}$$

The influence coefficients given by the bracketed terms $[\,\cdot\,]$ are calculated here using a multilevel expansion calculated with an interaction template at the beginning of each simulation. There are three individual interaction templates as seen above. At each calculation the vorticity terms α_i of all active boxes are anterpolated. The influence on all proactive boxes are then calculated with the multilevel expansions and the products are added together to get the desired far field velocity influence. In total, this operation requires six matrix multiplications, followed by three matrix additions.

Stretching It is possible to formulate the stretching equations (Eq. (2.17)) as is done above for the velocity field. Due to additional products here however this requires an additional 18 interaction templates, compared to the 3 required for the BS velocity interaction. Inspection of Eq. (2.18) reveals that the influence decreases as r^{-5}. This has the effect that the far field contributes minimally to the stretching terms. This is demonstrated in Fig. 3.11 where the contribution is seen to not exceed 0.5% ($H_b = 8\,H$). It is questionable whether this increased expense is warranted considering the great increase in computational expense.

A second method to calculate the far field stretching terms is by resorting back to the expression for the stretching term: $d\vec{\omega}_p/dt = (\vec{\omega}\cdot\nabla)\vec{u}$. Using the known vorticity of the probe particle, it remains simply to calculate the velocity gradient tensor $\nabla\,\vec{u}$. This is conveniently achieved using the properties of barycentric Lagrangian interpolation (Section 3.3.3), which allow for the calculation of the gradient based on the pre-calculated interpolation weights [109]. The velocity gradient due to the far field contributions are then known, and are then interpolated to the probe position. The use of the classic, transpose and mixed schemes here is easily facilitated by simply using $\nabla\,\vec{u}$, $(\nabla\,\vec{u})^T$ and $1/2(\nabla\,\vec{u} + (\nabla\,\vec{u})^T)$, respectively.

Although this method greatly reduces computational expense as compared to the aforementioned method, it should be noted that the approximation of the gradient on the interpolation nodes is essentially a weighted FD approximation, which incurs error. The error incurred by using this calculation was already investigated in Chapter 3. There it was seen that the error can be completely controlled with adequate specification of H_b and P. This approach is validated for actual flow cases in the following chapter.

PSE The viscous diffusion kernel decays very quickly with ρ. Testing has shown that regardless of choice of regularisation, the contribution due to particles at a distance $\rho > 5$ is essentially negligible. For this reason the far field viscous contribution is ignored here.

Turbulent shear stresses The approaches described in Chapter 2 require knowledge of higher order derivatives of the flow field. The HV method requires knowledge of the bilaplacian of the vorticity field $\nabla^4\vec{\omega}$, and the RVM approach requires knowledge of the small-scale vorticity $\vec{\omega}_s$, which implies the use of an appropriate spatial filtering on a regular grid. It would be possible to map the vorticity field to a local grid and extract this information, however this constitutes a large portion of the solution procedure of the `PML` solver, and hence the overhead incurred for this component alone within the

GML solver would make simply using the PML solver more practical. For this reason, no turbulence modelling has been incorporated into the GML solver.

Near Field

Unlike the far field influence, where the MLMIC method is used to accelerate the calculation, the near field influence must be directly calculated (see Section 3.3.4). The near field of any given box (and the probes within) is specified as the region which contains NF_{int} boxes in all spatial directions. If, for example, $NF_{int} = 1$, each box has 27 near field boxes (in 3D) which contribute to its near field solution. The influence on the probes of the sources within these boxes is directly calculating. The expressions for velocity, stretching and PSE are implemented directly from the closed-form expressions (Appendix F). The user is able to specify these with two solver options:

- REG specifies the type of regularisation desired for the particle smoothing. This choice influences all field parameters. The options are: LOA, HOA, GAUSS, SUP_GAUSS, HEJ, representing low-order algebraic, high-order algebraic [23], Gaussian [118], higher order Gaussian polynomial and spectrally convergent kernel [91], respectively. Unless otherwise stated the GAUSS regularisation has been used in the work here. The options above are applicable to 3D problems, however 2D particle regularisations were also implemented.
- STRETCH specifies which type of stretching scheme is to be used: The options are CLASSIC, TRANSPOSE and MIXED.

Specification of characteristic core size The expressions for the velocity, stretching and diffusion (PSE) terms are functions of the regularised radius ρ and therefore the core size σ of each source particle. The GML solver can accept particle sets with non-uniform core size specification for purposes of comparison and to ensure particle overlap for non-uniform problems, this however is generally not the case as uniform particle sets have been investigated. In the work here, the core size is specified such as to ensure that the particle overlap requirement is met after remeshing: $\sigma = 1.1\,H$. It was observed that, provided remeshing was regularly carried out, the results were not noticeably impacted for $1.1\,H \leq \sigma \leq 1.5\,H$, beyond this range however an overly large damping effect was observed in the velocity fields. It should be noted furthermore that it has been observed that for under-resolved simulations (large relative grid size), the stretching terms

would cause the particle set to diverge. This was difficult to avoid without prescribing an overly large σ to dampen the equation system. This could be somewhat delayed by the application of strong divergence filtering, however this did not overcome the underlying stability issue.

Specification of stretching scheme All three forms of stretching have been investigated. In Chapter 2 it was described that for numerous reasons the transpose method appears to be the most suitable. It was found through preliminary investigation that this scheme performed most satisfactorily in terms of stability, robustness, and conservation of flow quantities. For this reason, for all simulations in this work the `TRANSPOSE` scheme has been applied.

Optimisations The *box-box* style implementation implies that sets of particle interact, rather than single particles. This approach is amenable to calculation on the GPU, as the particle sets can be tiled into blocks of a given size, specified with the `TILESIZE` variable, and their interaction calculated in parallel. Depending on the hardware available the choice of tilesize can have a drastic influence on performance. Implementation has been done here with the `OpenCL` framework and the modelling options specified above are automatically accounted for when compiling the `OpenCL` kernels. This greatly increases speed as the data is passed in a single block to and from the GPU for each convolution, reducing communication overheads between the CPU and GPU. When using the `MULTILEVEL` solver option, the near-field interaction also makes use of these optimisations.

4.4 `PML` Solver

The field quantities in this method are extracted using FD on the Eulerian grids within each base box. The solution of the Poisson equation on each of these grids is achieved using the FP solver which requires two inputs:

- **Grid Vorticity** The values of vorticity on each grid are calculated by using mapping functions as described in Chapter 2. An additional global superposition check ensures that the boundary vorticity between neighboring regions is continuous as described in Section 4.1.5.
- **Boundary Condition** This is necessary to ensure that the stream function on the domain is correctly specified– see Chapter 2.

The distinction between near field and far field influence in the `PML` solver

hence refers not to the influence on individual probes as with the GML solver, but rather to the value of the stream function for the BC coordinates of a given Eulerian grid. Upon their calculation, the far field and near field terms are added to yield the full influence on each BC coordinate. Multiple approaches have been taken in the work here to calculate the near field contribution of neighboring regions, these are individually described below. For all methods however the far field methodology of the PML solver is equivalent, as described below.

Far Field

From Eq. (2.10), it is seen that the far field approximation of the stream function kernel is in fact even simpler than that of the BS kernel:

$$4\pi\,\vec{\psi} = \frac{\vec{\alpha}}{r} = \left\{ \left[\frac{1}{r}\right] \alpha_x \vec{e}_x + \left[\frac{1}{r}\right] \alpha_y \vec{e}_y + \left[\frac{1}{r}\right] \alpha_z \vec{e}_z \right\} . \tag{4.9}$$

This requires only a single interaction template $[r^{-1}]$ and three matrix multiplications. This can be further reduced by concatenating the three matrices α_i to require only one single matrix multiplication. The procedure follows otherwise exactly as for the BS kernel. The far field influence is then interpolated down the box levels to the proactive base boxes, where it is mapped from the interpolation grid to the boundary nodes of the Eulerian grid. This is again an interpolation template which is equivalent for all base boxes and can be calculated at the beginning of the simulation. This is illustrated in Fig. 4.6.

Near Field

Two methods have been implemented within the VPML solver for calculation of the near field influence when using the PML solver.

Near field direct evaluation

The first method implemented is activated by specifying the POISSON_DIR solver option. This method makes direct use of the Green's function for the SF kernel, see Eq. (2.15). Although conceptually the simplest of all approaches, this method is only really practical for relatively sparse particle distributions. This can be observed by carrying out an order of magnitude analysis of the number of evaluations required: the number of sources within a densely distributed volume N is proportional to the cube of the box

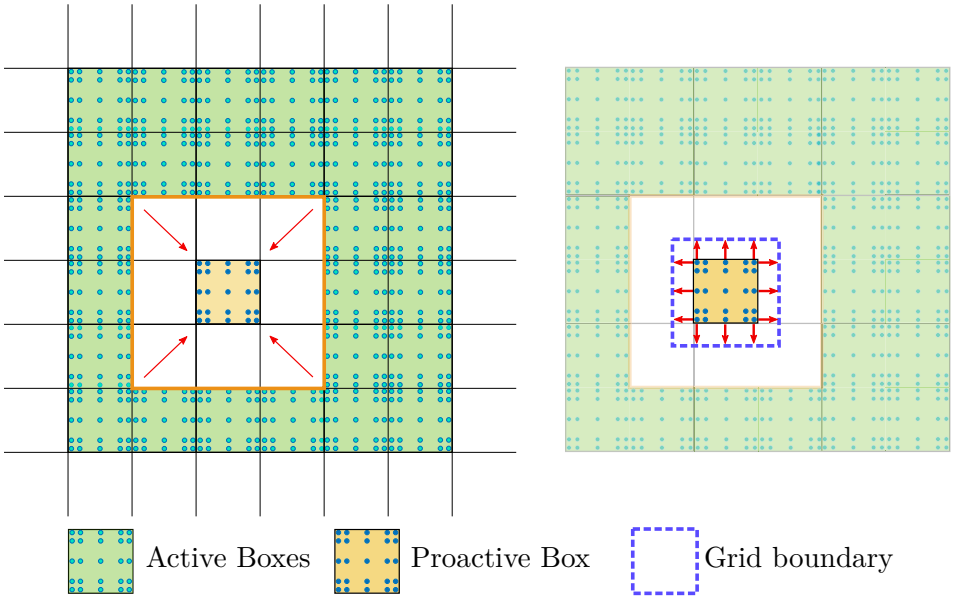

Figure 4.6: Far field specification of the stream function in the PML solver. Left: The far field influences on the receiver nodes of the proactive box is calculated with the multilevel expansion of $\vec{\psi}$. Right: The far field influence on the receiver nodes is interpolated to the boundary nodes of the Eulerian grid for specification of the solver. The boundary dilation is here exaggerated for illustrative purposes.

sidelength: $H_b = n\,H \longrightarrow N \sim n^3$, the number of boundary nodes is proportional to $n^2 \sim N^{2/3}$, the expense hence scales as $\mathcal{O}(N^{5/3})$ and has therewith unfavourable scaling. Despite this, the approach is practical as a means to validate the James-Lackner (JL) method described in the proceeding section. As with the BS kernel, the SF kernel must be regularised. This requires the specification of a particle core size. For the JL method described in the proceeding section, the singular particle treatment is used for boundary interactions and the specification of particle core size is unnecessary.

Specification of characteristic core size As opposed to the GML solver where the core size σ is specified by the overlap requirement, this is not the case for the PML solver as the particle-particle interactions are inherently handled by the grid solver. For consistency however, it is observed that when solving for an impulse node vorticity strength with the FP solver, the stream function distribution appears to behave as a regularised particle with a given core size. This is a function of the order of approximation used in the FD formulation of the FP solver stencil. For the Gaussian regularisation this was found to be $\sigma = 0.251\,H$. Interestingly, this value is

in exact agreement with Cocle et al., where a different FP solver was used ([28], Fig. 1).

James-Lackner method for near field calculations

The JL method is denoted within the solver as `POISSON_JL`, and makes use of the theory described in Section 2.4.2. This method requires additional solver steps, along with additional templates to treat boundary forcing from neighboring domains. In this method the stream function BC of each Eulerian grid it is composed of two contributions: the homogeneous solution and the single-layer solution, both from box self-influence and neighboring boxes. For simplified implementation each base box is filled with **two** Eulerian grids. The first, the omega grid $\mathcal{D}_\Omega$ is used for the homogeneous solve step to calculate $\vec{\psi}_0$. The second, the psi grid $\mathcal{D}_\Psi$) is used for the final solve incorporating the full BC. It was found in implementation that generating two distinct grids enabled simpler implementation as the grids have different overlap factors N_+.

Homogeneous solution The homogeneous solution $\vec{\psi}_0$ on the omega grid is calculated by executing the FP solver with the stream function BC set to zero. This step must occur **prior** to the superposition of the vorticity field from neighboring domains as described in Section 4.1.5, otherwise neighboring sources may be counted multiple times. The requirement that the vorticity field has compact support is conveniently achieved here by specifying the overlap factor N_+ of the omega grid to be larger than otherwise necessary, automatically padding the domain with a region of zero vorticity. As with the mapped vorticity, the homogeneous solution domain overlaps into neighboring domains, and an equivalent function as that for the vorticity is applied to ensure that the solutions of $\vec{\psi}_0$ are superimposed. It has been observed in practice that the larger the choice of N_+, the more continuous the solution for $\vec{\psi}_0$ over the boundary of neighboring domains is found. The choice of $N_+ = 5$ was found to be completely sufficient without incurring significant computational overhead. This motivates a larger choice of H_b for the `PML` solver, as a significant portion of the omega domain overlaps, increasing computational overhead and reduntant memory consumption.

Boundary forcing density The homogeneous solution is then used to specify the boundary forcing density $\vec{\gamma}$ over the boundary of the omega grid $\partial\mathcal{D}_\Omega$. This is calculated by determining the normal derivative over the surface by application of an appropriate FD stencil. The order of accuracy

of the FP solver dictates the order of the FD stencil used. In the work applied here the FP solver uses a five-point FD stencil and hence has 4^{th} order accuracy. For this reason the boundary forcing density is calculated with a 4^{th} order one-sided FD stencil.

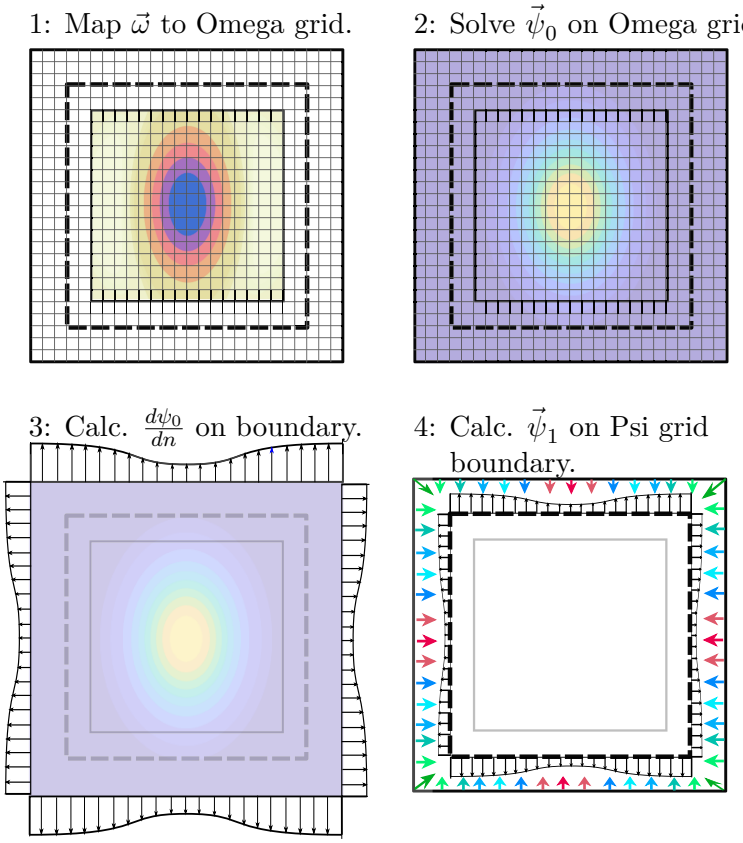

Figure 4.7: The four steps of the James-Lackner routine in the `PML` solver. Three distinct regions are seen. The box domain $\mathcal{B}$ (inner region), the Psi grid (dashed line), and Omega grid (large outer domain). The full solution to $\vec{\psi}$ on the boundary of the Psi domain is attained by adding ψ_0 and ψ_1.

Single layer solution This is calculated by numerically integrating Eq. (2.23) over the boundary $\partial\mathcal{D}_\Omega$ to calculate the influence of the boundary forcing on the boundary of the solution grid $\partial\mathcal{D}_\Psi$. The influence of a given boundary node can be treated as an integration over a source panel. This can be practically achieved with a Gaussian quadrature rule. A single-point Gaussian rule has been applied here for large separations and a 6×6 sub-paneling is applied for small separations. The influence of each source point i on probe boundary point j can hence be calculated for a box template and

results in an interaction template a_{ij}. It is seen in Eq. (2.22) that the SF Green's function $\mathcal{G}$ is integrated over the panel to calculate this influence. Although numerically it presents no challenges to implement the regularised form of this, better results were found by using the singular form of the SF. The singularity which arises for evaluation points which lie close together is effectively removed by the use of sub-paneling. This template is calculated not only for the single-layer self-influence, but also for neighboring boxes. This is calculated once at the beginning of the simulation for a box template family, resulting in 9 a_{ij} interaction templates for the 2D case and 27 a_{ij} interaction templates for the 3D case. This is illustrated in Figure 4.7.

This methodology demonstrates how with application of the JL method, the near field evaluation is reduced to a matrix multiplication (plus additional FP solver call), making the problem again suitable for execution on the GPU. The `POISSON_JL` routine is currently configured such that the 27 interaction matrices a_{ij} are concatenated into a single very large matrix. All possible near field interactions of each **active** box are calculated, and the desired influences $\partial\mathcal{D}_{\Omega,i} \rightarrow \partial\mathcal{D}_{\Psi,j}$ are then extracted and added to the BC of $\vec{\psi}_1$ for the each proactive box. This method reduces the necessary memory passing processes which must occur and allows the computational burden to be isolated to a single very large matrix multiplication.

Complete solution The total stream function BC for each proactive domain boundary is found by adding the homogeneous and single-layer solutions of the domain itself (if **active**) plus the active neighboring domains. This is illustrated in Figure 4.7 for a single domain (showing only self-influence). This constitutes the near field contributions. When the far field influence is added (see proceeding section), the total boundary condition is known and the FP solver can be again called to yield to full volume description of the stream function $\vec{\psi}$.

Extraction of Field Quantities from Grid Data

Once the stream function $\vec{\psi}$ over the domain of interest is known, the field data required to determine the evolution of the particle field can be calculated. This involves specification of a number of additional variables. The Eulerian grids store in theory the necessary information for the calculation of these parameters over their entire volume. In practice however, the information is only desired at the grid points surrounding the probe node position, with the stencil width corresponding to the mapping option selected. This total number of calculations can be restricted by carrying out FD evaluations only

where it is necessary. This is accomplished by marking the grid cells during probe binning procedure as being **proactive**. The nodes surrounding these proactive cells are then *marked* and listed for the FD calculation. This ensures that each FD calculation is only carried out once.

Velocity As described in Chapter 2, the velocity is extracted by carrying out a simple FD calculation on the grid. The order of accuracy of this calculation can be specified, however for the work done here a 2^{nd} order central FD scheme was applied. For nodes which are on an Eulerian domain boundary, a 2^{nd} order one-sided FD scheme is used. In practice however, due to the overlapping regions, the one-sided FD calculations are rarely required

Stretching A similar procedure here is carried out for the stretching component. This is however calculated with nodal velocity, obtained in the previous step. In order to avoid spurious results at *edges* of the active regions, the velocity must be calculated at additional boundary nodes. This is accounted for by padding the proactive nodes. It is stated in Cocle et al. [28] that using the conservative form of the stretching term $\nabla \cdot (\vec{u}\,\vec{\omega})$ led to better conservation of vorticity moments, however this was not observed in the work here. The quantity $\nabla\vec{u}$ (a second order tensor) is calculated on the grid and together with grid value $\vec{\omega}$ allows for the specification of the stretching term.

Viscous diffusion Inspection of Eq. (2.9) demonstrates that it is necessary to calculate the Laplacian of the vorticity field $\mathcal{L}(\vec{\omega})$ in order to specify the viscous diffusion. This is accomplished here using isotropic FD templates, in particular the second-order template (no. 5) from Patra & Karttunen [119]. These remove the directional bias and ensure that the FD stencil size is not greater than that of the stretching calculation carried out in the previous step. This approach was also used in Cocle et al. [28], however an alternative stencil was used.

Turbulent shear stresses As described in Chapter 2, two methods have been applied here to calculate viscous shear stresses. Both methods are carried out on the local Eulerian grid within each base box in order to facilitate parallel processing. Depending on the choice of turbulence model, different quantities must be resolved:

- **HV** model: The laplacian of the vorticity field $\nabla^2\vec{\omega}$ having already

been calculated for resolution of the viscous stresses enables calculation of the bilaplacian $\nabla^4\vec{\omega}$ by simply reapplying the Laplacian stencil to this field. This carries a very low computational expense.

- **RVM** model: Two field quantities must be resolved in order to apply this method. The first is the small-scale vorticity $\vec{\omega}_s$. This is achieved by applying the discrete operator $\mathcal{F}(x)$ to the vorticity field:

$$\bar{\omega}(x) = \mathcal{F} \circledast \omega(x) = \omega(x) + \frac{1}{4}[\omega(x+H) - 2\omega(x) + \omega(x-H)]\,. \quad (4.10)$$

Here filtering is applied in the x direction. The small-scale component is then found by subtracting this from the vorticity field $\omega_s = \omega - \bar{\omega}$. This is carried out first in x direction, then repeated in the other spatial dimensions $\mathcal{F}(y)$ and $\mathcal{F}(z)$. This produces the fully filtered small-scale field $\vec{\omega}_s^1$. The procedure is repeated on this field n times to produce a filter of order n. The second component is the Smagorinsky sub-grid scale viscosity $\nu_{sgs} = C_r^n \Delta^2 \sqrt{2S_{ij}S_{ij}}$. The entries of S_{ij} were calculated previously for the stretching terms $\nabla\vec{u}$. The stretching calculation step can hence be exploited to additionally calculate this term. These are stored such that the scalar quantity $\sqrt{2S_{ij}S_{ij}}$ on the grid is known. The bilaplacian stencil is then applied in the following form to resolve the turbulent shear stresses:

$$\nabla \cdot \nu_{sgs}[\nabla\vec{\omega}_s + (\nabla\vec{\omega}_s)^T] = C_r^n \Delta^2 \mathcal{L}\left(\vec{\omega}_s \sqrt{2S_{ij}S_{ij}}\right)\,. \quad (4.11)$$

4.5 Monolithic `PML` Solver

A second architecture of the `PML` was carried out for the purposes of achieving the goal outlined in the objectives of optimising for a CPU-GPU environment. This is marked in the `PML` solver as `POISSON_MONO`, which stands for the monolithic Poisson solver. This is so named as the FP solver is used to solve not for each individual base box, but rather for a monolithic Eulerian grid $\mathcal{D}_{\text{mono}}$ which contains all vorticity sources. The stream function BC over the monolithic grid boundary $\partial\mathcal{D}_{\text{mono}}$ are calculated with the MLMIC method, greatly accelerating the evaluation. Care is taken to ensure that the boundary nodes of the monolithic Eulerian grid are always in the far field of the vorticity sources, ensuring full exploitation of the efficiency of the MLMIC method. This implementation provides an MLMIC equivalent to a number of well-established monolithic Poisson solvers in the literature such as the `FLUPS` library [96] or the `MIRAS` solver [72, 73].

Although this method is not strictly in alignment with the fundamental ideology of the `VPML` library– as it does not exploit a *tightly* constrained

solution domain– it allows for a much faster solution for certain geometries. The reason for this is that the ideal scaling of the FP solver is combined with the speed up afforded by the MLMIC method.

Monolithic grid In alignment with the `VPML` framework, $\mathcal{D}_{\text{mono}}$ is constructed using a rectangular prism of base boxes $\mathcal{B}_{0,i}$, with the `BLOCK` grid option specified. At the beginning of each time step, the particle set extrema are checked to ensure that any boundary nodes on $\partial\mathcal{D}_{\text{mono}}$ are within the far field of the sources. If this is not satisfied, $\mathcal{D}_{\text{mono}}$ is grown with a buffer factor to ensure that domain resizing occurs infrequently. During creation of $\mathcal{D}_{\text{mono}}$, all base boxes $\mathcal{B}_{0,i} \in \mathcal{D}_{\text{mono}}$ are created and stored in a static list.

Parallelisation The binning procedure occurs exactly as with the standard `PML` solver. Source and probe data preparation and anterpolation are carried out in parallel for all active and proactive boxes $\mathcal{B}_{0,i}$. The Eulerian grid within each base box $\mathcal{B}_0$ is then mapped to the Eulerian grid of $\mathcal{D}_{\text{mono}}$. An equivalent procedure is carried out to map the solution $\vec{\psi}$ to the proactive base boxes $\mathcal{B}_0$. The entire FD and probe mapping procedure can therewith proceed for each base box in parallel exactly as with the standard `PML` implementations.

Boundary conditions When the monolithic grid is created, all $\mathcal{B}_0$ on $\partial\mathcal{D}_{\text{mono}}$ are stored in a designated list. These are automatically marked as being `ML_Proactive` for the multilevel expansions. After evaluation of the far field, the values of the stream function at the boundaries of interest are extracted from the multilevel expansion of the necessary boxes and interpolated to the corresponding boundary node positions of $\partial\mathcal{D}_{\text{mono}}$. This is illustrated in Fig. 4.8. The far field calculation of the stream function proceeds exactly as with the standard `PML` implementations.

Problems Exhibiting Periodicity The monolithic `PML` solver has also been configured to simulate cases with periodic symmetry. It is common for FP solvers to have the capability to handle periodic BCs in one or numerous spatial dimensions. In this case the BCs for certain faces need not be specified within the solver as they are fully specified by the periodic boundary condition:

$$\vec{\psi}(x) = \vec{\psi}(x + L_{per}) \ , \quad \frac{\partial\vec{\psi}}{\partial x}(x) = \frac{\partial\vec{\psi}}{\partial x}(x + L_{per}) \, . \tag{4.12}$$

Here and hereafter it is assumed that the flow has periodicity in x direction with a periodic domain length of L_{per}, as in Fig. 4.5. The concept is however applicable also to other Cartesian directions. This implies that the BC of ψ over the $\partial\mathcal{D}_{\text{mono}}$ faces $x = const.$ need not be calculated. The MLMIC routine is configured such that these boundary faces are not included within the multilevel expansions, and the specification of the boundary condition proceeds otherwise exactly as for the standard monolithic solution routine. Additional care needs to be taken to ensure that the vorticity field is continuous in the periodic direction. This is ensured by implementing the monolithic Eulerian grid such that any overlapping regions ($x < 0$ or $x > L_{per}$) are automatically mapped back to the domain $0 < x < L_{per}$. This is carried out for numerous procedures for the VP method including remeshing, divergence filtering and mapping results from the monolithic grid to individual box grids.

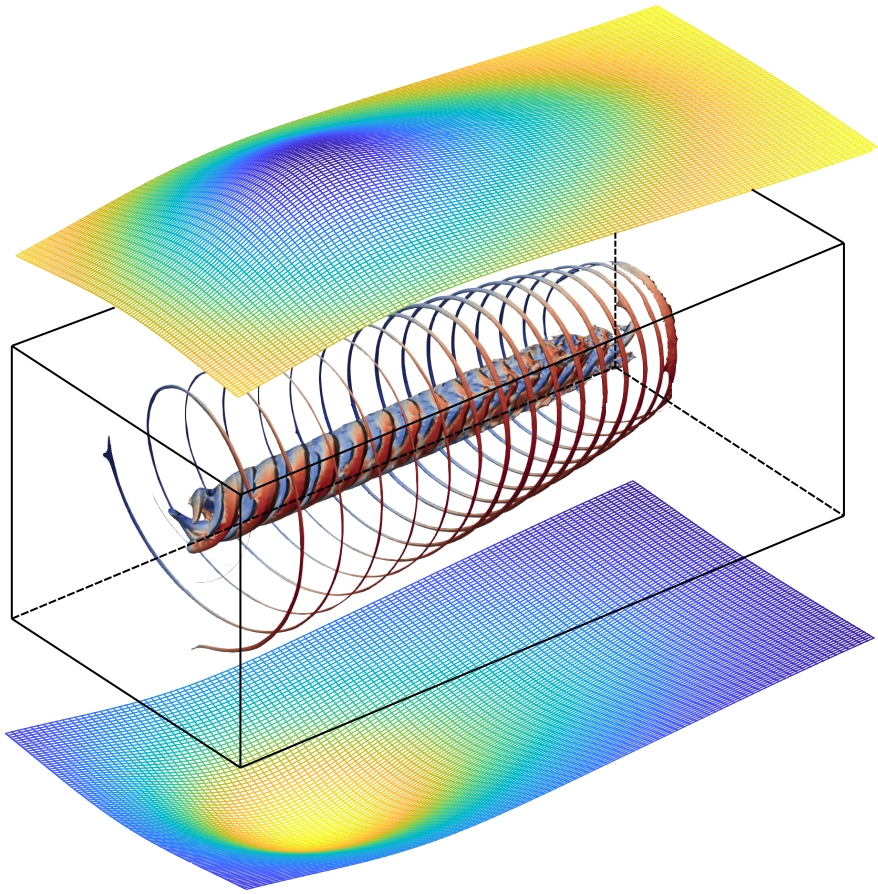

Figure 4.8: The monolithic Eulerian grid around a turbine simulation. Isosurfaces of vorticity of the turbine are shown. The monolithic grid contains all sources of vorticity and the FP solver is executed once for the entire volume of interest. The stream function boundary conditions are calculated with the MLMIC method. A visualisation of ψ_y calculated over the z_+ and z_- boundaries calculated by the ML method is shown here.

Chapter 5 Flow Solver Validation

The GML and PML solvers have been validated for numerous flow cases against both analytical and numerical results from the literature in Section 5.1. The flow cases simulated here involve vortex ring geometries, as these demonstrate numerous well-understood physical phenomena for vortical flows of an incompressible fluid. A detailed overview of the theory of the vortex ring has been included in Appendix E. The cases demonstrated here represent *dense* 3D vorticity fields for which vortex filaments are spatially resolved. A number of additional validation cases were simulated to demonstrate solver sensitivities and lower complexity phenomena.

In Section 5.2 a performance analysis of the GML and PML libraries is carried out and the optimal computational expense $\mathcal{O}(N)$ for the algorithm is observed. The scaling of the solvers for HPC applications is investigated. A summary describing optimal solver choice is then described. For all results shown here the PML solver has been used with the POISSON_JL option enabled, the near field is therefore being calculated with an interaction matrix.

5.1 Flow Solver Validation

These cases validate the solvers in order of complexity of the physical phenomena:

1. **Single vortex ring** Demonstration of correct calculation of the velocity field;
2. **Translating and colliding vortex rings** Accurate capture of vortex time evolution, vortex stretching and viscous merging for a laminar flow case;
3. **Low Re unstable vortex ring** Instability growth and turbulent breakdown using the HV turbulence model;
4. **High Re unstable ring** Instability growth and turbulent breakdown using the RVM turbulence model.

Supplementary validation cases For brevity numerous cases have been omitted and included in the Appendices as a supplement to the validations

performed here. These cases act to further demonstrate the efficacy of the solvers against analytical solutions. A brief summary is provided here and the reader will be guided towards relevant results throughout this chapter:

1. **2D Solver** Appendix A: The solver is validated for two 2D vortex cases: Inviscid shearing of an elliptical vortex and viscous merging of a vortex couple to a steady analytical solution (Burgers Vortex)
2. **Filament Cases** Appendix B: The `GML` solver is validated for two cases where a sparse vortex particle set is investigated, an infinite helical vortex and a vortex ring. This demonstrates the application of the MLMIC method for sparse particle sets, which have practical applicability to low-resolution simulations.
3. **Steady Flow** Appendix C: The Hill's vortex is an ideal inviscid case for which there exist analytical expressions for all fields of interest. This is inspected to demonstrate accuracy and sensitivity of both solution methods to grid resolution.
4. **Filament Cases** Appendix D: In order to isolate stretching phenomena two dense inviscid vortex rings are collided to demonstrate conservation of vorticity and efficacy of the stretching schemes.

5.1.1 Grid and Field Parameters

In general a single parameter must be specified for a given simulation, namely the grid size H. This specifies the characteristic spacing between particles after remeshing is carried out and at the beginning of a simulation. All other parameters are specified based on this as described in Chapters 3 and 4. The values which depend upon this are given in Table 5.1. The

Table 5.1: Characteristic grid parameters

Parameter	Unit	Description	`GML`	`PML`
dV	m^3	Particle volume	H^3	H^3
σ	m	Particle core size.	$1.1\,H$	$0.251\,H$
H_b	m	Multilevel grid size.	$8\,H$	$16\,H$
P	$-$	MLMIC polynomial order	5	5

choice of $H_b = 8H$ for the `GML` solver is motivated by the results of Section 3.5.3, where a choice of $H_b > 6\sigma$ ensures satisfactory far field calculation with the MLMIC method. For the `PML` solver this is increased to $H_b = 16H$ in part because of the results of Section 3.5.4, but also for the reasoning

outlined in Section 4.4. In general, choosing this factor to be a power of two is convenient for implementation reasons.

Eulerian mapping parameters If the `PML` solver is being used, the M_4' scheme has been used for mapping field quantities to and from the Eulerian grid. FD calculations on the grid have been carried out with a 2^{nd} order FD stencil. In cases where surrounding grid nodes exist, a centered scheme has been used. On Eulerian domain boundaries a one-sided scheme has been used. For the calculation of the boundary forcing density $\vec{\gamma}$, a 4^{th} order one-sided FD stencil has been used.

Regularisation parameters Unless otherwise stated, a Gaussian regularisation has been used and the transpose scheme is used to calculate stretching with the `GML` solver– see Appendix F. When using the `PML` solver, the near field interaction matrix is calculated based on a singular Green's function and hence the regularisation becomes redundant as described in Section 4.4.

5.1.2 Unsteady Flow Parameters and Diagnostics

For all tests which follow, the flow is allowed to evolve in time. This implies that a time integration of the particle set occurs. The particle position is being updated with Eq. (2.8) and the particle strength with Eq. (2.9). As described in Chapter 4 this is carried out with the Adams-Bashforth Leapfrog integration routine `AB2LF`. If the particle set has been resized (e.g. after remeshing), the Runge-Kutta 2^{nd} order scheme `RK2` has been applied. This ensures a consistent 2^{nd} order time integration scheme. Time step size is given by dT and is generally specified based on the characteristic time of the problem under consideration T_{char}.

Particle set filtering The particle set is regularly updated with numerous checks to ensure that the particles are uniformly distributed and the particle set is consistent as described in Section 2.5.

- **Remeshing**. For consistency with the Eulerian grid mapping, the M_4' scheme has been used and unless otherwise stated, occurs every 10 time steps. This value was found to be suitable for the investigations carried out here, although it is noted that more frequent remeshing may be required depending on flow topology and integration scheme

or step size. This is monitored by ensuring that maximum particle translation is small between remeshing.

- **Magnitude filtering** To ensure removal of diffused and/or superfluous particle regions, particle magnitude filtering also occurs after remeshing. Unless otherwise stated the magnitude filtering factor $\kappa = 1\text{e}{-3}$ is used.
- **Divergence filtering**. The scheme used to ensure $\nabla \cdot \vec{\omega} = \vec{0}$ is applied in both the `PML` and `GML` solvers. This is carried out after remeshing of the particle set. The frequency of this is a simulation parameter. Experience has shown it is practical to carry out this step after remeshing.

Flow diagnostics The accuracy of the particle set evolution can be checked by monitoring flow diagnostics which are known to obey certain functional relations. When discussing a real flow, conserved quantities are referred to as flow *invariants*. When discussing a simulated flow these are referred to as flow *diagnostics* and can be monitored throughout a simulation [23]. It shall be hereafter assumed that the fluid has unit density. Three **linear** invariants of an unbounded flow exist which describe conservation of circulation, linear and angular momentum, respectively:

$$\vec{\Gamma} = \iint_S \vec{\omega}\, dV \qquad \overset{\text{disc}}{=} \sum_p \vec{\omega}_p\, dV_p\,, \tag{5.1}$$

$$\vec{L} = \frac{1}{2}\iint_S \vec{x} \times \vec{\omega}\, dV \qquad \overset{\text{disc}}{=} \frac{1}{2}\sum_p \vec{x}_p \times \vec{\omega}_p\, dV_p\,, \tag{5.2}$$

$$\vec{A} = \frac{1}{3}\iint_S \vec{x} \times (\vec{x} \times \vec{\omega})\, dV \qquad \overset{\text{disc}}{=} \frac{1}{3}\sum_p \vec{x}_p \times (\vec{x}_p \times \vec{\omega}_p)\, dV_p\,. \tag{5.3}$$

The discrete expressions here demonstrate calculation for a particle set. For a compressible fluid the term dV_p must be updated to account for dilatation, this however remains constant in an incompressible fluid [5]. Linear invariants describe conservation of the moments of vorticity, which upon inspection are also connected directly to the impulse of the fluid. The linear impulse $\vec{I}$ (5.1) can be considered the total mechanical impulse of non-conservative body forces required to instantaneously generate from rest the motion at any time t. An equivalent consideration for the torque can be made for the angular impulse $\vec{A}$ (5.3). Provided the vortex particle representation of the vorticity field is divergence free, the linear impulse should be conserved [22]. Both linear and angular impulse are conserved in a viscous flow [85]. In the case that regularised vortex particles are being used, angular impulse must be appended with an additional term $\vec{A}$: $-\frac{1}{3}C\sigma^2\vec{\Gamma}$.

In the case of a Gaussian regularisation it can be shown that $C = 3\sqrt{2\pi}$. In addition to these there exists three **quadratic** diagnostics.

$$E = \frac{1}{2}\iint_S (\vec{u}\cdot\vec{u})\,dV \qquad \overset{\text{disc}}{=} \frac{1}{2}\sum_p (\vec{u}_p\cdot\vec{u}_p)\,dV_p\,, \tag{5.4}$$

$$\mathcal{E} = \iint_S \vec{\omega}\cdot\vec{\omega}\,dV \qquad \overset{\text{disc}}{=} \sum_p \vec{\omega}_p\cdot\vec{\omega}_p\,dV_p\,, \tag{5.5}$$

$$\mathcal{H} = \iint_S \vec{\omega}\cdot\vec{u}\;dV \qquad \overset{\text{disc}}{=} \sum_p \vec{\omega}_p\cdot\vec{u}_p\,dV_p\,. \tag{5.6}$$

In an inviscid flow kinetic energy E (5.4) is conserved. In a viscous unbounded flow however, it can be demonstrated from kinematical considerations [1], that:

$$\frac{dE}{dt} = -\nu\mathcal{E}\,. \tag{5.7}$$

Enstrophy (5.5) is directly related to the kinetic energy and corresponds to viscous dissipation. This can be used as a check in simulated flows that viscous effects are correctly being captured. It was shown in Moffatt [120] that the helicity (5.6), a measure of the net linkage of vortex lines, is in fact a fourth invariant in an inviscid flow, however is not conserved in a viscous flow.

5.1.3 Case 1: Single Vortex Ring

The case of a single translating vortex ring presents a suitable initial test to ensure that the velocity field is correctly calculated from a given vorticity distribution. For a specific ring vorticity distribution, an analytical solution is known which allows for validation of the velocity calculation at a frozen point in time. The influence of time stepping parameters, integration scheme and particle evolution equations hence play no role in the solution, these are described for the unsteady tests cases in the proceeding sections. As previously described, other steady cases were carried out which demonstrate the order of accuracy of the method as a function of grid size H, particularly the case of the Hill's spherical vortex in Appendix C. For the case inspected here an overview of the underlying theory of the vortex ring is provided in Appendix E. The vortex core of the ring is aligned along a circle of radius R. The quantity inspected here is the global normalised translation velocity U of the vorticity centroid as a function of ring core size. An analytical expression for this was derived by Saffman [121] and an improved, less conservative estimate of this result was provided by Stanaway et al. [122]. These were extracted by allowing the particle set to convect for a single time

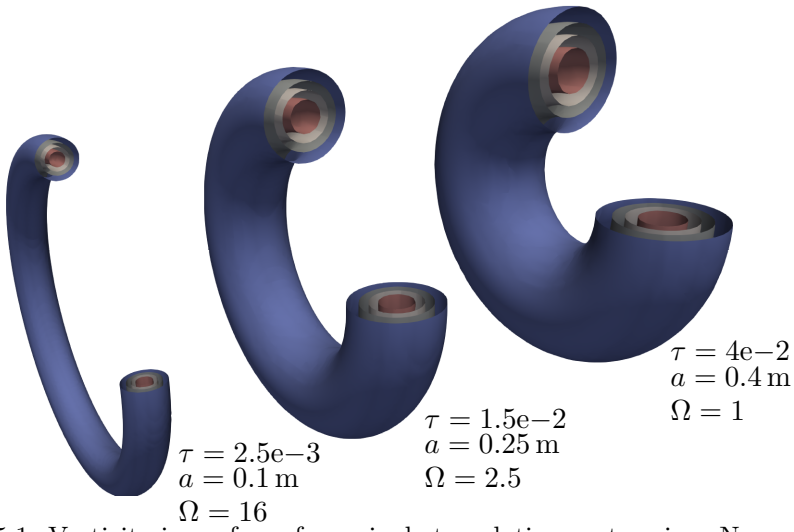

Figure 5.1: Vorticity isosurfaces for a single translating vortex ring. Normalized time τ shown. Colour scheme normalised with $\omega_{max} = \Omega \cdot 10^3 \text{m}^2\text{s}^{-1}$. Translation due to self induction is seen along with diffusion as ω_{max} decreases and the core expands.

step dT and evaluating the change in the position of the vorticity centroid X_1. The vorticity distribution around the vortex core is as described in Saffman [121]:

$$\omega_0(r) = \frac{\Gamma}{4\pi\nu t} \exp\left\{-\frac{r^2}{4\nu t}\right\} = \frac{\Gamma}{\pi a^2} \exp\left\{-\frac{r^2}{a^2}\right\}, \tag{5.8}$$

where Γ is the ring circulation, r represents distance from the vortex core and ν represents the kinematic viscosity of the fluid. This allows the specification of a characteristic time $t_0 = R^2/\Gamma$. This allows specification of a normalised time $\tau = t/t_0$. The initial vorticity distribution is generated on a regular grid with spacing H. The variable $a = \sqrt{4\nu\, t}$ is the viscous core size of the ring. A single, translating vortex ring is illustrated in Fig. 5.1. The particle count increases quickly for decreasing a. This is because the global dimension of the problem (R) remains constant, while H must be continuously decreased to ensure sufficient resolution of the vortex core. This was achieved by setting $H = 0.1\,a$ in the test here. As described in Stanaway, the effects of curvature on the assumption of a perfectly Gaussian vortex ring cross-section increase with the vortex Reynolds number $Re_\Gamma = \Gamma\nu^{-1}$, for which reason this is chosen here to be relatively small: $Re_\Gamma = 1\text{e}{-2}$.

The results were compared to the analytical expression for a range of a values and are shown in Fig. 5.2. The agreement with the improved estimate due to Stanaway is seen to be excellent for both solvers. As expected by the

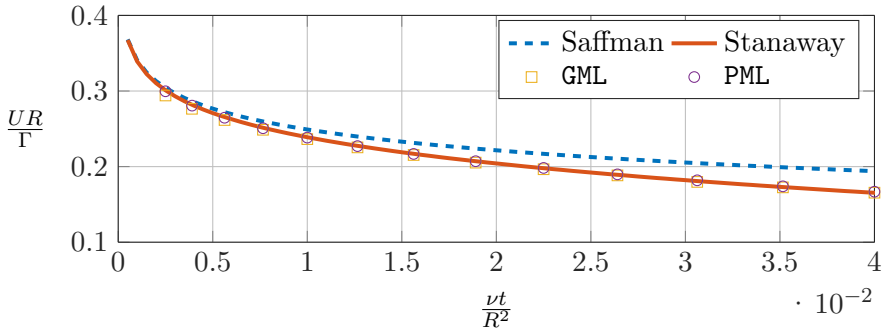

Figure 5.2: Translation velocity of a Saffman vortex ring as a function of normalised time.

theory, the effect of viscosity is to decrease the velocity as the vortex core spreads due to viscous diffusion. In the limit as $\nu t \to 0$ (which approximately is satisfied for these points) the core size of the filament is small enough that stretching terms are essentially negligible here.

5.1.4 Case 2: Translating and Colliding Vortex Rings

This test case has been divided into two sub-cases. The first inspects the simpler case of a translating inviscid vortex ring. This validates the time integration and particle set filtering procedures. The second investigates the fully viscous merging process undergone by two obliquely colliding viscous vortex rings and validates the treatment of both stretching and viscous diffusion in the solvers.

Translating inviscid vortex ring

The first unsteady test case investigated is that of a simple translating inviscid vortex ring. The initial vorticity distribution generated as with the Saffman vortex ring (Eq. (5.8)) with core size such that $a^2 = 4\nu\, t = 1\text{e}{-2}$. Without the effects of the viscosity, the perfectly axisymmetric vortex ring can be assumed to remain undeformed and translate under the action of self influence alone. The effects of finite core size introduce non-azimuthal stretching terms which will eventually act to compress the torus cross-section into an elliptical form [122], eventually introducing perturbations which will lead to azimuthal instabilities, these are however left for later cases as the goal here is simply to demonstrate the time evolution of the simulation.

Evolution of particle set The evolution of the system can be effectively checked by inspection of the particle set diagnostics. These are displayed in Fig. D.3 against normalised simulation time $\tau = tR^2\Gamma^{-1}$. The time series shown represents a simulation with over 10^4 time steps. It is seen in Fig. D.3 that both solvers advance as expected in time with conservation of circulation and angular momentum. Linear momentum is seen to slightly decay with time, this is a consequence of the vorticity field not remaining divergence free [23]. This is not due to an erroneous implementation of the divergence filtering routine, but rather follows from the relatively large choice of $\kappa = 1\text{e}{-3}$. This phenomenon shall be further explored in the next section. Either reduction of the time step or increased frequency of the divergence filtering remedies this issue. Finally, as expected for an inviscid flow the helicity is seen to be preserved quite well. In general for all quantities the `PML` solver appears to perform slightly better than the `GML` method.

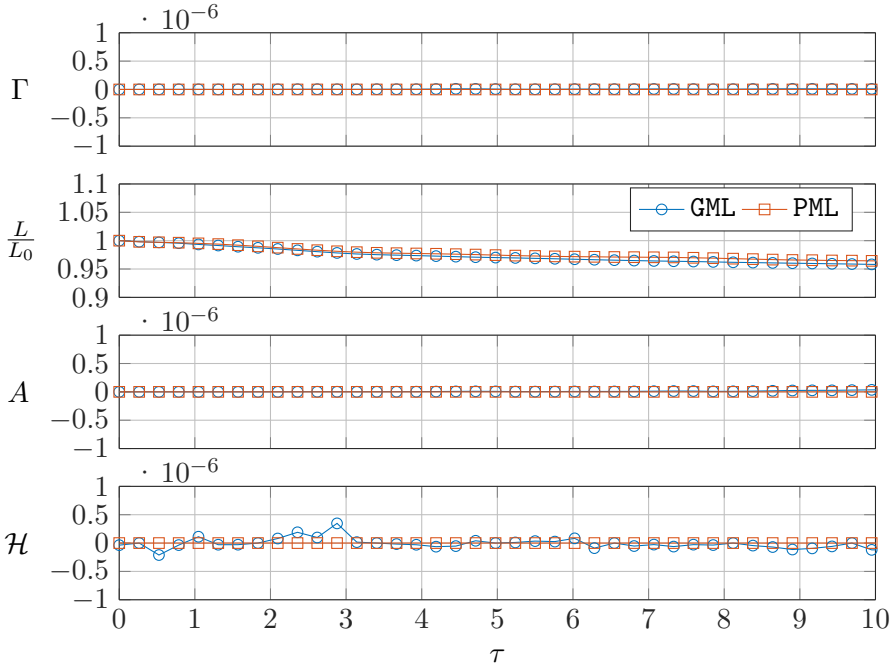

Figure 5.3: Flow diagnostics for the case of an inviscid translating vortex ring. Simulations here were advanced through 10^4 time steps.

Colliding vortex rings

An intermediate case for demonstration of vortex stretching would include dilation of the vortex ring under inviscid conditions. This case is included in Appendix D for brevity. In the case presented here two vortex rings are initially aligned obliquely such that they collide during their translation. During the collision process, the vortices merge together and form a single vortex filament. The process of vortex merging is a viscous process distinct from diffusion where regions of high vorticity merge together to form new strong vortex filaments. This is illustrated for the case here in Fig. 5.4. The initial vorticity field has been generated as in the previous cases with $a = 0.1\,\mathrm{m}$ and $\Gamma = 10\,\mathrm{m^2 s^{-1}}$. For consistency with the test case in Winckelmans & Leonard [23] the rings have radii $R = 1$, initial separation (centre-to-centre) of $\Delta y = 2.7\,\mathrm{m}$ and an initial inclination angle $\theta = 15°$. The kinematic viscosity of the fluid has been set such that the vortex Reynolds number $\mathrm{Re} = \Gamma\nu^{-1} = 400$.

Evolution of particle set Linear diagnostics for the evolution of the particle set are shown in Fig. 5.5. As with the previous case it can be seen that the linear diagnostics behave essentially the same as with the inviscid case: circulation and angular impulse are perfectly conserved and linear impulse decays slowly with time. This was detailed previously and shall be explained further here. It was observed that if the magnitude filtering parameter is too large, removal of particles decreases the total linear impulse of the system, particularly for large simulation times. In order to illustrate this, the case $\kappa = 1\mathrm{e}{-5}$ is shown in Fig. 5.5. For the latter case, linear impulse conservation is seen to be drastically improved for both solvers. This however has a direct influence on the computational expense of the simulation, as the particle count increases greatly as κ is reduced. The current implementation ensures conservation of circulation. Based upon the points outlined above however, it would be advantageous to have an implementation which ensures linear impulse conservation, independent of the choice of κ. As with the previous case for the dense vortex it is observed that the `PML` generally performs better across all diagnostics.

Merging process Inspection of quadratic diagnostics for this case allows for the demonstration of the decay of kinetic energy through viscous diffusion. These diagnostics are shown in Fig. 5.6. Here the quantity dE/dt has been approximated with a simple 1^{st} order Eulerian forward finite difference. The functional relationship of Eq. (5.7) is generally well captured through both diffusion and the connection process, demonstrating that viscous processes

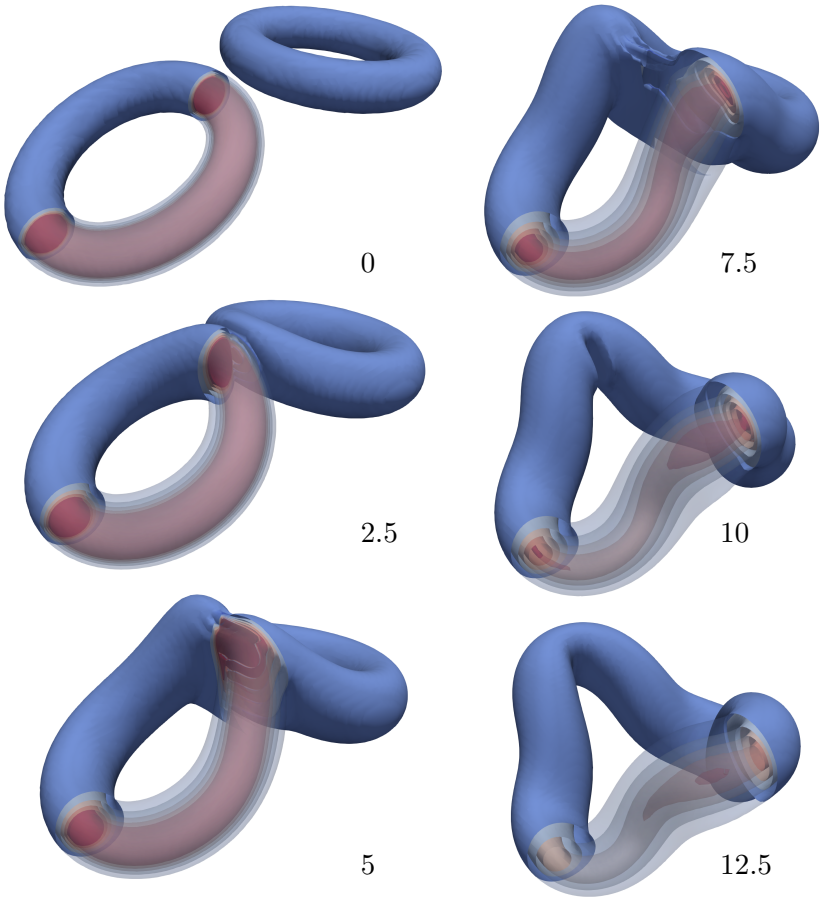

Figure 5.4: The collision and viscous merging of two vortex rings. Normalized time τ shown.

are well represented. A qualitative check is provided by observing the progression of the ring motion displayed in Fig. 5.4. For the times $\tau = 2.5$ the attraction typically seen in merging scenarios is observed [123]- this is also observed for the 2D case in Appendix A. At $\tau = 7.5$, the two filaments have paired off to form a new single continuous vortex filament which continues to translate downwards. This reconnection due to viscous effects is well documented for a range of initial ring geometries and the behaviour here is consistent with known results [23, 124]. The qualitative results of the `PML` and `GML` solvers were practically identical, the solid contours in Fig. 5.4 are in fact those from the `PML` solver and the semi-opaque contours from the `GML` solver.

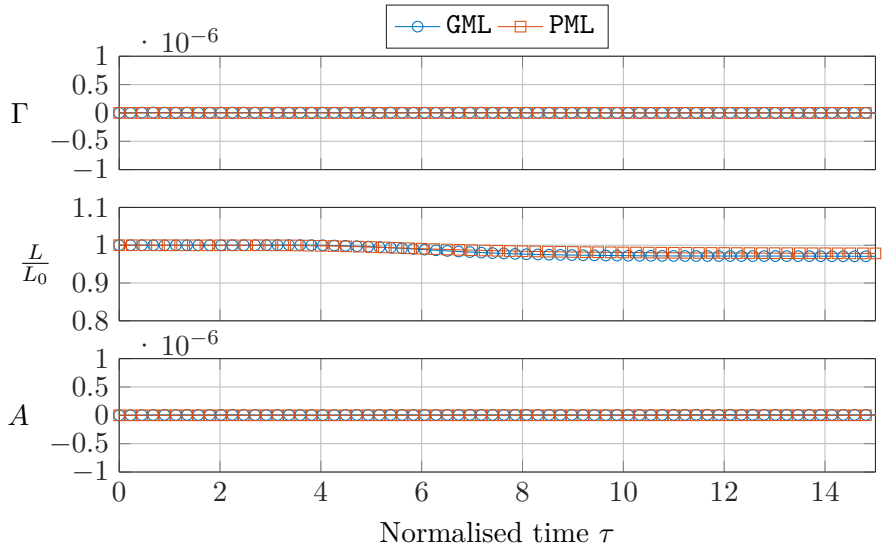

Figure 5.5: Progression of flow diagnostics for the case of colliding viscous vortex rings. Simulations here were advanced through 10^4 time steps.

5.1.5 Case 3: Low Re Perturbed Vortex Ring

Application of the turbulence models requires access to higher order spatial derivatives (or spatial filtering for the `RVM` model) which excludes the use of the `GML` solver. For this reason the following results represent the ability of the `PML` solver to capture the effects of turbulent diffusion. This is demonstrated by simulating the evolution of an azimuthally perturbed vortex ring. In this test the hyperviscosity turbulence model (`HV`) is investigated.

Unstable vortex rings Within an experimental environment, the generation of a perfectly axisymmetric vortex is quite challenging. This is usually carried out by forcing a slug through a nozzle with a sharp lip. Generally for $Re < 600$ a stable, laminar vortex ring is formed. For $Re > 600$, perturbations in the initial vorticity distribution are observed to grow as the vortex translates, this is shown with a dye injection experiment in Fig. 5.7. Perturbations are seen to grow linearly until non-linear effects dominate and the flow reaches the saturation point, where the ring becomes fully turbulent and a breakdown process occurs. The physical mechanism for the initial stages of perturbation growth is considered to be well understood and is described further in Appendix E. By considering any perturbation in a spectral sense to be composed of a set of sinusoidal excitations [125, 126], the growth of these perturbations can be predicted. For these perturbations,

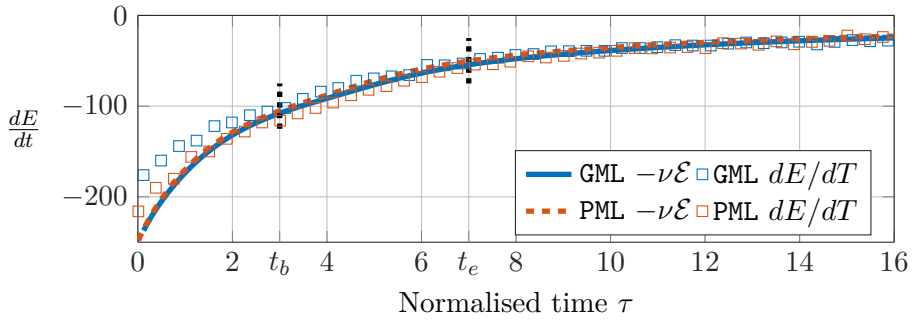

Figure 5.6: The rate of decay of kinetic energy of the particle set is demonstrated here as dictated by Eq. (5.7). The merging process occurs approximately within $t_b \leq \tau \leq t_e$.

the wave number with the fastest growth rate k_{max} will dominate the breakdown process. k_{max} depends heavily upon the initial conditions of the ring and kinematic parameters such as core size a and ring Reynolds number $\mathrm{Re} = \Gamma\nu^{-1}$. This will be investigated here by simulating a vortex ring for which there exists previous numerical results.

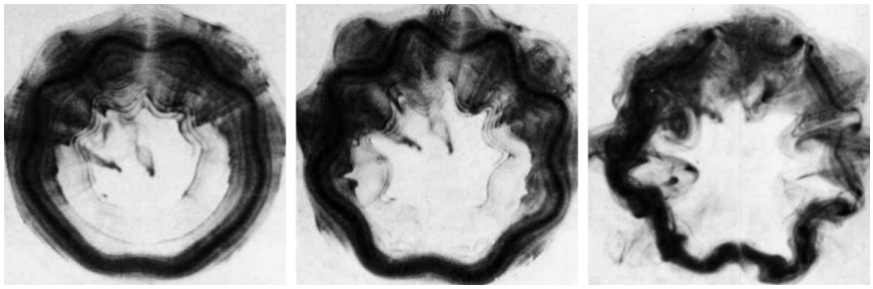

Figure 5.7: Azimuthal instabilites of a vortex ring at $\mathrm{Re} \approx 2000$ made visible by injecting dye into the vortex core. Reproduced from Van Dyke [127].

Initial vorticity distribution The case here follows the test case given in Shariff et al. [128] (hereafter referred to as SH). The vortex ring is generated as in the previous cases, with the exception that the ring radius is perturbed with an azimuthal excitation $R(\theta) = R(1 + \epsilon g(\theta))$. Here $g(\theta) = \sum_m cos(m\theta + \phi_m)$ is a sum of Fourier modes with unit amplitudes and random phases. The first m azimuthal modes are hence uniformly excited. The initial vorticity distribution accounts not only for the perturbation of the ring radius due to $g(\theta)$ at each azimuthal position, but also for the perturbed orientation of the vorticity distribution by realigning the tangential vector along $\vec{e}_{mod} = \vec{e}_\theta + \frac{dg}{d\theta}\vec{e}_r$ in order to maintain consistency.

Simulation parameters The simulation here echoes Case 1 from SH. The ring has a vortex Re of 5500. The time step has been chosen such that $dT = 0.01\tau$ where $\tau = R^2\Gamma^{-1}$. The ring has a core size $\sigma = 0.4131\,R$ and the grid has a resolution of $H = 0.04\,R$. The perturbation parameter is chosen to be low enough that it does not affect the dynamics during the initial translation phase: $\epsilon = 2\text{e}{-4}\,R$ and the first 24 azimuthal modes are perturbed. Particle set filtering is carried out every 10 time steps and the magnitude filtering factor is set to $\kappa = 1\text{e}{-3}$. The simulation was carried out for $0 \leq t \leq \tau$. The global time constant has been set to $T_0 = \tau$ and the Smagorinsky constant set to $C = 2.5\text{e}{-2}$, following Cocle et al. [28](hereafter referred to as COCLE).

Spectral analysis In order to inspect quantitatively the evolution of the modal energies, a spectral analysis was employed. This was carried out by performing an azimuthal Fourier transform of the flow quantities, defined for the azimuthal mode m of a stream function ψ in cylindrical coordinates as:

$$\hat{\psi}_m(r,z) = \frac{1}{2\pi}\int_0^{2\pi} \vec{\psi}(r,\theta,z)e^{-im\theta}\,d\theta\,. \tag{5.9}$$

This is performed numerically by discretising the field azimuthally in segments of arclength $d\theta = 2\pi RH^{-1}$. Only the modes for which $m{\geq}0$ are inspected as the symmetry of the modes can be exploited: $E_m = E_{-m}$. This allows the modal energy to be expressed as [28]:

$$E_m = \iint [\hat{u}\cdot\hat{u}^*]r\,dr\,dz = \frac{1}{2}\iint [\hat{\psi}\cdot\hat{\omega}^* + \hat{\omega}\cdot\hat{\psi}^*]\,r\,dr\,dz\,, \tag{5.10}$$

where $(\,\cdot\,)^*$ denotes complex conjugation. This formulation allows direct extraction from the stream function solution. This has been carried out by discretising the field in radial and axial directions ($dr = dz = H$) and integrating numerically. Each Fourier transform is hence denoted as a complex vector. The first N_θ (most energetic) modes have been used such that the total energy of the field is given as $E_{tot} = \sum_m^{N_\theta/2} E_m$. This method follows that described in COCLE.

Flow stages Four distinct stages can be identified during the simulation.

1. **Transient stage** $\tau \leq 25$: The ring geometry relaxes and approaches a steady flow pattern. Excitation modes are energetically too weak to be easily identified.
2. **Linear stage** $25 \leq \tau \leq 80$: The fastest growing (linear) eigenmodes begin to dominate transients.

3. **Non-Linear stage** $80 \leq \tau \leq 150$: Higher wave number modes grow more rapidly than the first eigenmodes.
4. **Saturation & decay stage** $150 \leq \tau$: Instabilities saturate and the turbulent decay process begins.

These phases shall be inspected chronologically in the following. A visualisation of the process is given in Fig. 5.8 and 5.9.

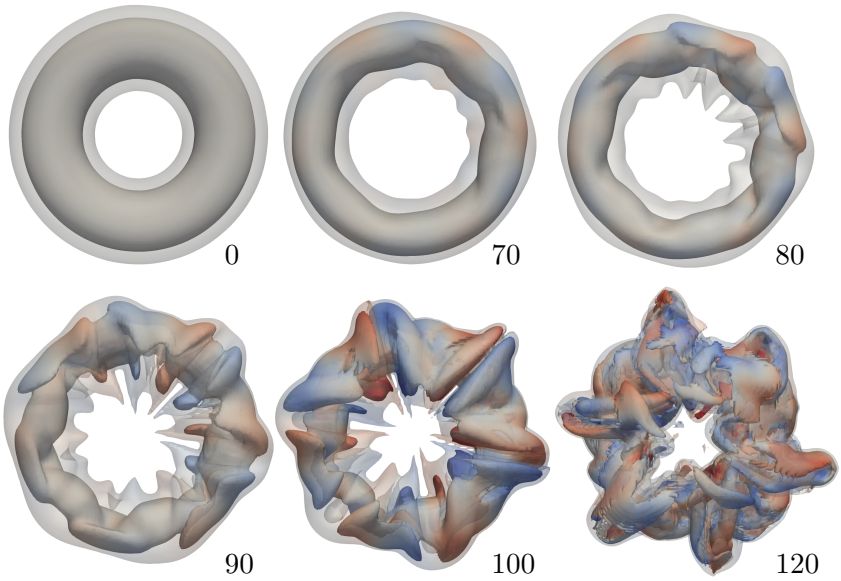

Figure 5.8: The evolution of the vortex ring. Two isocontours of vorticity are visualised. The color scale indicates ω_z, initially zero. Normalized time τ shown.

Transient stage As described in the heuristic model of Maxworthy [129], a region of low energy vorticity is shed by the ring after it has been entrained from the upstream fluid. This can be identified with a dividing streamline which becomes steady within the transient stage. Prior to this, a low-energy vortical wake is left behind by the translating ring. This wake is initially visible and eventually vanishes due to viscous diffusion. This is observed in the simulations and is shown in Fig. 5.10. After a certain translation distance the initially toroidal isocontours of the ring become flattened out such that they take on a more elliptical shape. This is due to the self-induced strain field of the vortex ring, as described and observed in Stanaway et al. [122]. This is seen by inspecting the vorticity contours for $\tau = 20$ in Fig. 5.10. These are reminiscent of the Hill's spherical vortex as described

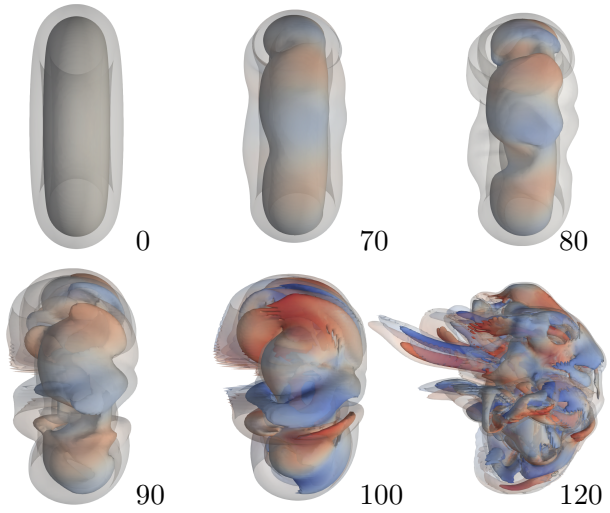

Figure 5.9: The evolution of the vortex ring perpendicular to the axis of symmetry z. Visualised are two isocontours of vorticity. The color scale indicates ω_z, initially zero. Normalized time τ shown.

in Appendix C or the family of quasi-steady Hill's-like axisymmetric vortex rings described in Norbury [130].

Linear stage The development of the flow field in time is seen in Fig. 5.11. The initial distribution is essentially unperturbed and the vortex ring translates as expected. The relative modal energies is shown in Fig. 5.12 for the first 10 modes. It can be seen that modes 6, 7 and 8 appear to show the quickest growth in the simulations here, in agreement with COCLE. The modal energies have been used to numerically estimate the modal growth rates $\alpha_m = (2E_m)^{-1} dE_m/dt$, shown in Fig. 5.13. The agreement of the 7^{th} and 8^{th} modes in particular is seen to be excellent. The factors for the second mode α_2 is not shown as E_2 still displays oscillatory behaviour at $\tau = 45$, this is also seen to occur in both COCLE and SH. In SH it is suggested that these modes are perhaps forced by the axisymmetric core unsteadiness, which has a much larger time scale compared to the growth rate of E_2. It is observed that the 6^{th} azimuthal mode dominates the linear stage as shown in the cross-sectional vorticity seen at approximately $\tau = 80$ in Fig. 5.11 and 5.8. This is in disagreement with COCLE, where $m = 7$ dominates, is however in agreement with SH, where modes 5-7 dominate. It is suggested in COCLE that this may be erroneous due to the use of periodic boundary conditions in the translation direction.

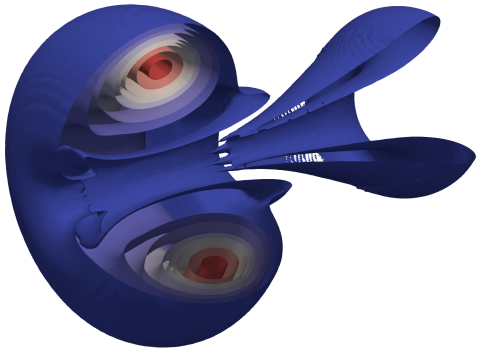

Figure 5.10: The low energy wake trailing the ring at $\tau = 20$. The streamlines around the ring have begun to form elliptical contours.

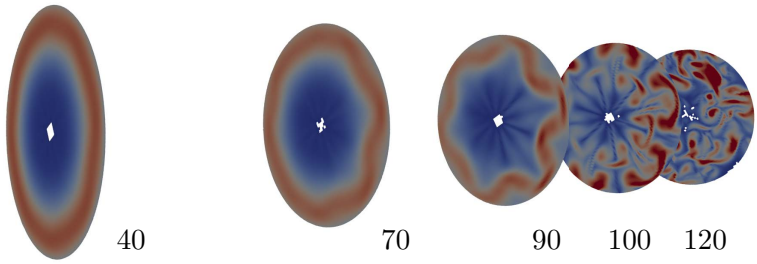

Figure 5.11: Planes through the vorticity centroid of the translating ring- colour scale represents vorticity magnitude. Normalised time τ shown. The amplification of the 6^{th} azimuthal mode can be seen, along with the transition into the non-linear stage at $\tau = 90$. The breakdown beyond saturation is also observed.

Non-linear stage It is seen in Fig. 5.11 and 5.8 that as the nonlinear stage begins, clefts of concentrated vorticity around the ring are seen. This is indicative of secondary radial extrema. A lateral visualisation looking perpendicular to the azimuthal axis is given in Fig. 5.9. There it is seen that these clefts locally induce a displacement in axial direction of the surrounding ring segments. The modal growth rates for $\tau = 100$ are shown in Fig. 5.13. The growth rate of dominant low wave number α_2 is seen to be captured well. In addition to this the amplification of E_{12} is seen to agree in both cases. As most linearly amplified mode predicted is $m = 6$, it is clear that E_{12} represents the first harmonic of the dominant mode. The higher modes are also seen to be growing $m = 17$, 18 and 19, these are then the third harmonics. Besides slightly higher prediction of the growth rates of modes 5 and 6 (and their harmonics), the general trends align well with COCLE.

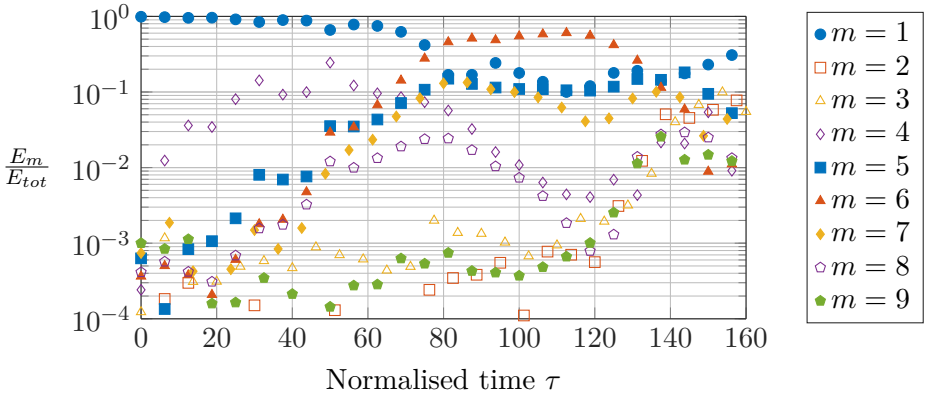

Figure 5.12: Fractional modal energy. It seen that modes 6-8 have the quickest growth through the linear phase.

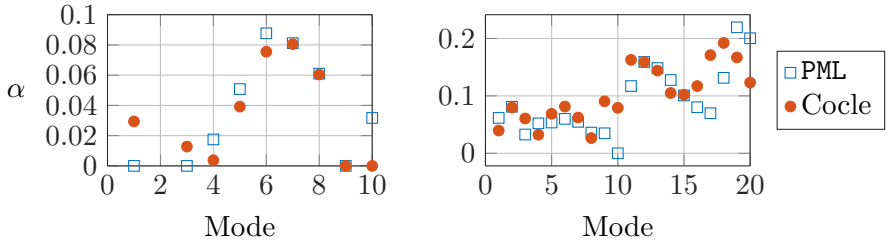

Figure 5.13: Modal growth rates α_m of the azimuthal modes for the low Re vortex ring. Displayed are the growth rates for the linear stage $\tau = 45$ (left) and for the transition stage $\tau = 100$ (right).

Saturation and decay stage As is seen in Fig. 5.11 at the end of the nonlinear stage, the saturation point is reached: The modal energy growth is constrained by the appearance of strong turbulent diffusion and the flow enters into a turbulent decay regime. The breakdown of the ring is accompanied by the injection of mass into the wake as seen in Fig. 5.9. The azimuthal modes become visually indiscernible and all large scale energy cascades down into turbulent eddies. The spectral energy distribution has been plotted in Fig. 5.14. The dominance of the 6^{th} mode is seen in the linear stage at $\tau = 80$. After saturation and partially into the turbulent decay phase, it can be seen that the $k^{-5/3}$ Kolmogorov scale characteristic of isotropic turbulence is approached [131, 132].

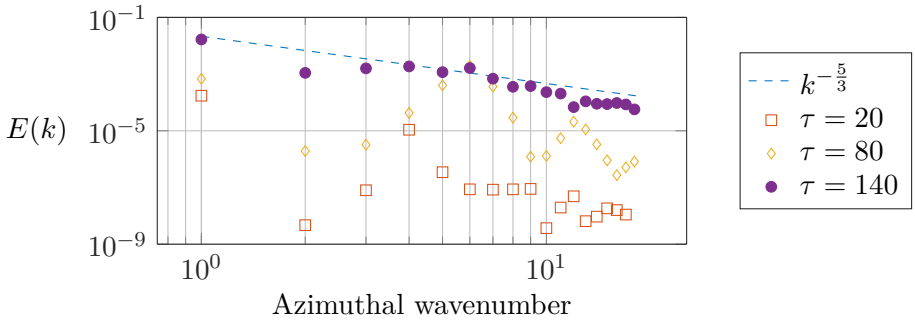

Figure 5.14: Spectral energy distribution as a function of azimuthal wavenumber in the transient, linear and decay phases.

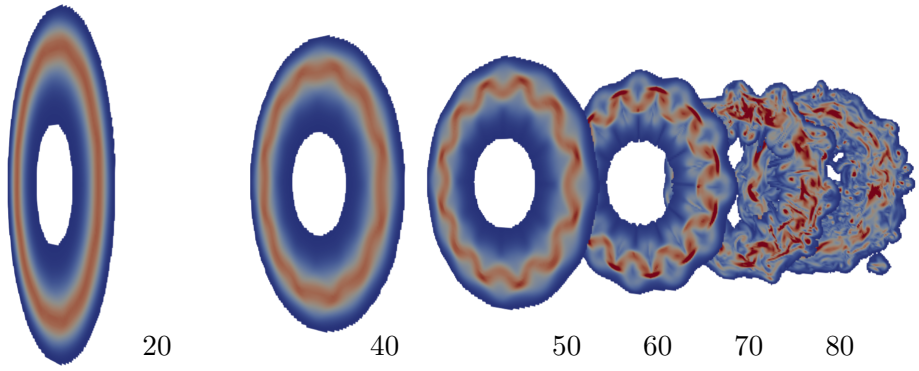

Figure 5.15: Planes through the vorticity centroid of the translating ring. Normalised time τ shown. The amplification of the 11^{th} azimuthal mode can be seen. The breakdown beyond saturation is also observed. Color scale represents vorticity magnitude.

5.1.6 Case 4: High Re Perturbed Vortex Ring

A second unsteady vortex ring test is carried out here here for a higher Re number ring. This case was chosen for the `RVM` turbulence model, as the decay phase should be better captured by this model. This is because the RVM model is guaranteed not to dissipate in the well-resolved flow region [76]. The case investigated is essentially identical to the previous unstable ring test, however with a modified initial ring distribution.

Simulation parameters The ring has a higher Reynolds number of Re = 25K and a smaller initial core size of $\sigma = 0.2\,R$. For this reason the grid is also refined to $H = 0.02\,R$. The excitation has been reduced to $\epsilon = 1\text{e}{-3}\,R$. The modal excitation was again carried out for modes up to

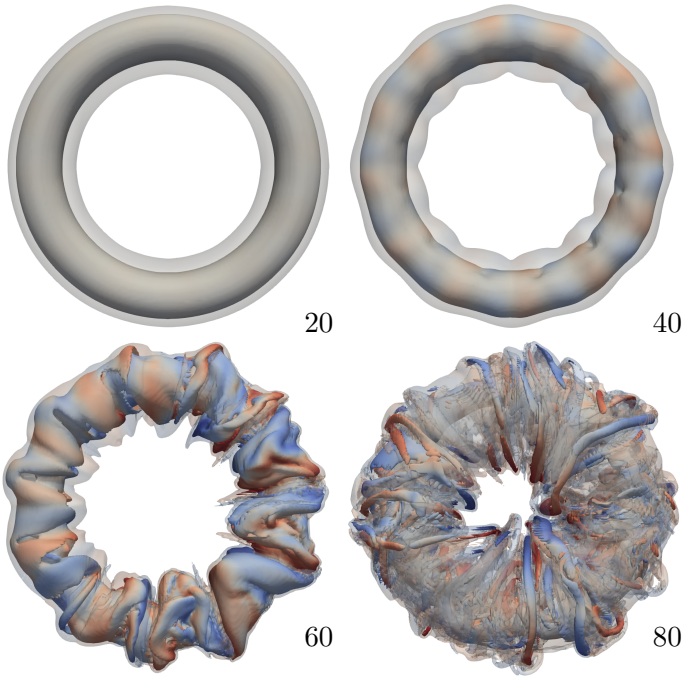

Figure 5.16: The evolution of the vortex ring for the case Re = 25K. The color scale indicates ω_z, initially zero. Normalised time τ shown.

$m = 24$. The small-scale filtering as described in Chapter 2 was carried out on the vorticity field twice in order to achieve a 2nd order filter. The Smagorsinki constant for this filter was taken from the PhD thesis of Cocle [76]: $C_r^{\{2\}} = 0.0476$.

Linear stage In general, increasing the ring Re results in a higher dominant unstable mode. This is observed to be the 11th azimuthal mode for this test case as seen in the vorticity cross-sections for $\tau = 50$ in Fig. 5.15. This is also seen in the growth of E_{11} in Fig. 5.18. This initial growth of the perturbations of the 11th mode are observed in Fig. 5.16 where not only radial perturbations are visible, but also a nonzero ω_z component appears. An approximation to the most amplified wave number as a function of time $N(t)$ based on analytical arguments was given by Saffman [126]. This is a function of the velocity profile in the core. The reader is referred to SH and COCLE, the calculated function will be stated directly here as $N(t) = 2.51\, R\, [1.12091\sqrt{\sigma^2 + 4\nu t}]^{-1}$. In the linear stage of the evolution of a ring with these kinematic parameters, this predicts that the 11th mode will grow most quickly. The simulated behaviour here therefore agrees with

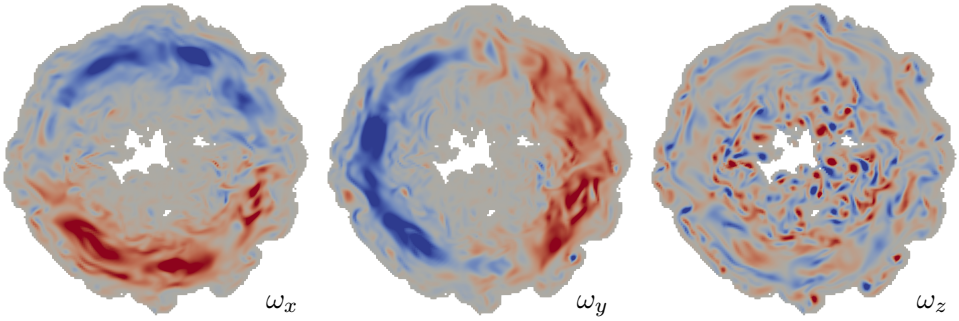

Figure 5.17: Vorticity at $\tau = 80$ through the vorticity centroid of the ring along the plane z=const. The formation of eddies of variable scale is visible, particularly in axial direction.

the predictions of Saffman.

Nonlinear stage The nonlinear stage is reached much earlier in the present simulation ($\tau \approx 60$) than for the equivalent test in COCLE. This is certainly explained by the coarser resolution in the present simulation, which is roughly twice the grid size used in the work presented there. The simulation may hence be slightly under-resolved here which amplifies the nonlinear effects. For purposes of inspection of the `RVM` model this is secondary. Similar flow phenomena are observed as for the low Re case where azimuthal segments of the ring fold upon themselves and act to amplify the nonlinear growth as is seen in Fig. 5.16.

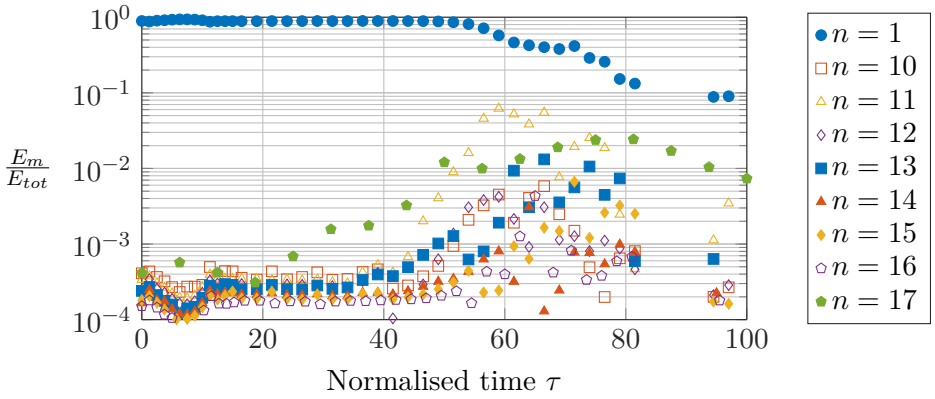

Figure 5.18: Fractional modal energy. It is seen that mode 11 has the quickest growth through the linear phase.

Saturation and decay stage Fig. 5.18 shows that the modal growth has essentially saturated by $\tau \approx 70$. The larger flow structures, although somewhat still present at $\tau = 80$, have begun to decay into smaller eddies. This is visualised for the ring cross section in Fig. 5.17. It is also observed in Fig. 5.16 that fine-structured secondary filaments are formed which wrap around and permeate the vortex ring. The spectral energy distribution is given in Fig. 5.19. Here, as with the previous case, the dominant modes are seen in the linear phase and the $k^{-\frac{5}{3}}$ Kolmogorov scale characteristic of isotropic turbulence is approached.

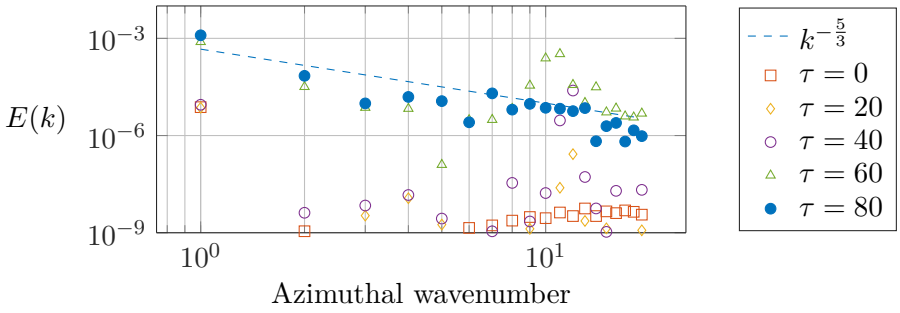

Figure 5.19: Spectral energy distribution as a function of azimuthal wavenumber in the transient, linear and decay phases.

5.2 Flow Solver Performance Analysis

In order to demonstrate the performance of the library and theoretical benchmarks, an investigation of the calculation time and memory requirements for a range of problem sizes has been carried out. To enable a better overview of the results and implementation, a review of the parallel processing architectures is first given. Following this, the efficacy of the MLMIC method has been evaluated using direct evaluation as a benchmark. Computational expense along with memory demands of each solver are individually investigated and important bottlenecks and opportunities for optimisation are highlighted. The scalability of the solvers is then investigated for application in high-performance computing (HPC) cluster environments. Finally a comparison between all implementations under equivalent conditions is carried out to identify the most practical use cases.

5.2.1 Parallel Processing

Two parallel processing architectures have been employed here to improve computational efficiency. These are described here prior to their application in the performance evaluation which follows. Suitable hardware parameters are discussed along with ideal application cases.

OpenMP The shared-memory multi-processing platform `OpenMP` has been applied here to allow parallel processing on central processing unit (CPU) nodes [115]. In order for portions of the calculation to be efficiently computed concurrently, there must be very little or no data dependencies between the threads. An example of this in the `VPML` library is the source preparation (hereafter **source prep**) step. This begins with source binning, where nodes are sorted into their corresponding spatial box, a process of almost negligible computational expense. Prior to the global anterpolation and interaction steps of the MLMIC method, the data within each box must be locally prepared for subsequent steps (this varies for each solver and shall be detailed in the proceeding section). This process requires essentially no communication between boxes, and is hence suitable for parallel calculation with `OpenMP`. In fact, the `VPML` library has been deliberately programmed in such a way that processes which occur on data **within** boxes have been compartmentalised within box objects in order to enable parallel calculations. These processes include **source**, **probe**, and **output** prep steps. These processes can be executed in parallel over all box objects. Minimal memory overhead is incurred and theoretical scaling proportional

to the number of available processors on the machine N_c should be attained. This formulation furthermore makes the library amenable to execution within a HPC framework with CPU nodes.

OpenCL The platform-heterogeneous framework `OpenCL` has been applied here for data-parallelised computing carried out on the graphical processing unit (GPU). The GPU is generally more suitable for cases with a very large number of low-complexity calculations with high data-dependency. An ideal application of this is the direct evaluation problem, where the influence at each evaluation point requires accessing the data of all source particles. The high bandwidth achieved by having a very large number of processors acting on a simplified kernel makes this $\mathcal{O}(N^2)$ process highly amenable to GPU treatment. Following the same argument, matrix multiplication is an ideal application being calculated on the GPU. This is a high-bandwidth problem with a very low-complexity kernel. The library has been prepared in such a way that high bandwidth processes such as these can be calculated on the GPU. A disadvantage here is the overhead incurred by data transfer between CPU and GPU. Depending on problem size this can constitute a significant portion of the execution time and is highly hardware dependent.

Hardware description The tests here were carried out on a desktop with the following hardware:

- CPU: AMD Ryzen Threadripper (1950X): 16-cores, 3.40 GHz clock frequency
- GPU: AMD Radeon RX Vega (GFX900): 64 compute units, Max work group size 256, 1247 base frequency.
- Installed Memory: 32 GB RAM

The GPU however certainly does not reflect the state of the art and for some of the software implemented is not an optimal setup. This will be detailed in the following sections.

5.2.2 Direct Evaluation

The computational expense of carrying out the direct particle interaction for a range of problem sizes has been investigated. This allows comparison to the MLMIC method in the following section and furthermore allows investigation of theoretical problem scaling. The geometry of a vortex ring has again been used and was generated as described in the previous section.

The problem size was modulated by adapting the grid size H- the grid sizes are described in Table 5.2.

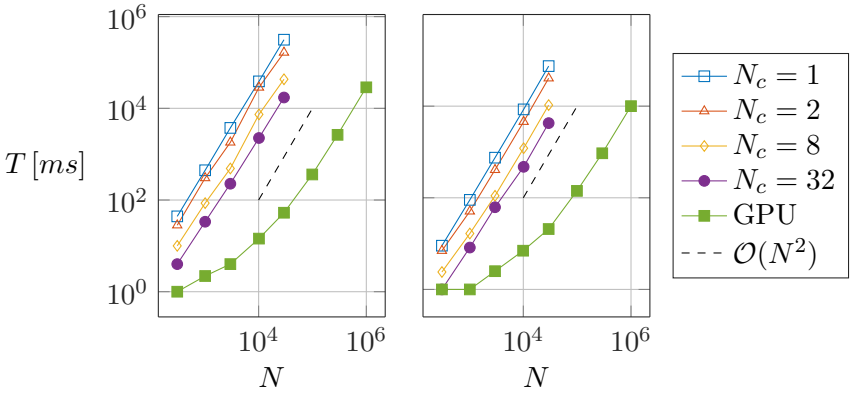

Figure 5.20: Direct evaluation of the test geometry with N particles. Left: `GML` (velocity and stretching field), Right: `PML` (stream function). The dashed line represents the theoretical scaling $\mathcal{O}(N^2)$.

OpenMP Displayed in Fig. 5.20 is the computation time T (wall time) for a direct evaluation of the particle set for a range of problem sizes on the CPU with N_c parallel threads. A few points should be made here: It can be seen that T in inversely correlated with N_c, the number of concurrent threads being used suggesting that the parallelisation is functioning accurately. This will be detailed further in the scalability examinations carried out in the proceeding section. The theoretical scaling of the direct evaluation $\mathcal{O}(N^2)$ is also captured. Both kernel types of interest in the work here have been tested, the BS kernel for the `GML` solver and the SF kernel for the `PML` problem. The problem scaling in both cases is seen to be essentially identical, however the computation time for the SF kernel is significantly lower due to the greater complexity of the BS kernel. It is also observed that as suggested above, the overhead incurred by applying `OpenMP` is independent of N_c.

OpenCL Also shown in Fig. 5.20 are the computation times on the GPU. Here one sees precisely how suitable direct evaluation problem is for GPU calculation, for larger problem sizes the GPU time is more than two orders of magnitude faster than the equivalent calculation on the CPU. This however is constrained by the additional overhead in passing the data between CPU and GPU as described previously. This is seen by the flattening curve for lower particle counts, where for small N the overhead required makes calculation on the GPU less beneficial. Despite the drastic increases

in computation potential achieved, the unfavorable $\mathcal{O}(N^2)$ scaling of the evaluation is still observed.

5.2.3 MLMIC Evaluation

Each implementation has applicability depending on available resources and choice of solution method. For each solver, a range of resolutions shall be investigated in order to demonstrate the scaling of the libraries. The three MLMIC solver configurations employed in the work here have been investigated: the `GML` solver, the `PML` solver and the monolithic `PML` solver. Computational steps have furthermore been classed into four categories: input preparation, far field calculation, near field calculation and output preparation in order to identify bottlenecks and optimisation potential.

Table 5.2: Simulation Parameters: Number of active boxes N_B

$H\,[m]$	N	N_B-`GML`		N_B-`PML`		N_B-`PML` mono	
0.258	308	8	(1)	8	(1)	152	(2)
0.175	1016	24	(1)	8	(1)	152	(2)
0.123	2984	32	(1)	8	(1)	152	(2)
0.082	10147	64	(2)	24	(1)	240	(2)
0.057	29580	160	(2)	32	(1)	240	(2)
0.038	99078	368	(3)	64	(2)	344	(3)
0.0265	292522	992	(3)	200	(2)	552	(3)
0.0176	999268	2816	(4)	448	(3)	872	(3)
0.0122	3003902	8018	(4)	1168	(3)	1408	(4)
0.00815	10077655	25352	(5)	3734	(4)	2864	(4)

Problem description The ring geometry generated for the previous test cases has been exploited once more with variable grid sizes H and variable polynomial order P. The problem size for each grid resolution is given in Table 5.2 for each solver. The number of active boxes N_B is shown, along with the branch level reached in the MLMIC integration in parentheses.

`GML` Solver

The performance of the `GML` solver is shown in Fig. 5.21. The calculation time for different values of P is given there. It can be seen that the behaviour for $10^5 \leq N \leq 10^6$ appears to behave linearly, however the cost increases beyond this point. The explanation for this is found when analyzing the

computation of the near field step, described below. For large N, the near field step dominates the computational expense. Although results for $P = 1$ are possible, these are omitted here for consistency with the `PML` solver routines. The integration has been broken up into four steps with the relative time of each step $\chi = T_i/T_{tot}$ displayed in Fig. 5.21.

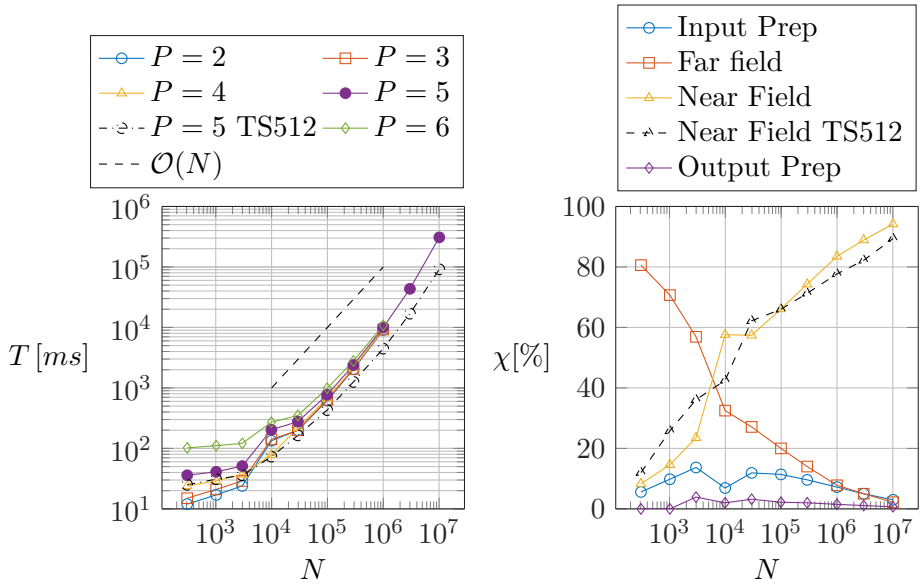

Figure 5.21: Left: Calculation time as a function of particle size for different polynomial orders: The dashed line represents an $\mathcal{O}(N)$ complexity. Right: Relative time for calculation steps ($P = 5$). The performance increase by using tile size $TS = 512$ is also demonstrated.

Input preparation: This is composed of source and probe binning and anterpolation to the base box source nodes. This is seen inf Fig. 5.21 to be a relatively negligible portion of the calculation time, particularly for very large particle counts.

Far field calculation: This is composed of anterpolation up the to necessary box level (shown in parentheses in Table 5.2), calculation with the interaction templates, and finally interpolation down to the base box level. The majority of the calculation time of this step is the multiplication of the interaction matrices. The multiplication is carried out on the GPU using the open-source software `CLBlast` [112]. It can be seen that this consumes a very large portion of the calculation for smaller particle sets and constitutes

the main reason why the MLMIC method is not beneficial for particle sets with small N.

Near field calculation: This step calculates the interaction between neigboring boxes and is seen to dominate the computation time for large N. This has been implemented in OpenCL as this problem is well handled there. For optimised execution on the GPU these interactions are grouped into tiles of size TS, which dictates the number tasks on each GPU processor node. TS greatly impacts performance as a choice below the average number of nodes in a box results in a scaled number of tile-tile interactions which increases overhead and switching time on the GPU. The GPU chosen here has a maximum work group size of 256. For the choice $H_b = 8\,H$, a densely populated box contains up to 512 source nodes, implying additional partitioning. To investigate the effect of this, an equivalent calculation was carried out on a more performant GPU with $TS = 512$. The results are seen to drastically reduce the computational expense and the scaling of the expense approaches the optimal complexity $\mathcal{O}(N)$ as expected from the theory.

Output preparation: Far field influence (velocity and stretching term) is interpolated from receiver nodes within the base box level to probe nodes. Calculation time here is greatly reduced by storing interpolation matrices in the input prep step. This too contributes negligibly for N large.

`PML` Solver

The performance of the `PML` solver with solver is shown in Fig. 5.22. There it can be seen that the behaviour for large N optimally scales with complexity $\mathcal{O}(N)$. As with the `GML` solver, the near field step generally dominates computational expense. Solution with $P = 1$ is not possible as interpolation at the box boundary for the Poisson boundary condition requires at least two receiver points in each spatial direction. It is also observed in Table 5.2 that the number of active boxes N_B is greatly reduced as compared to the `GML` method. This lies purely in the fact that for the `PML` solver $H_B = 16\,H$ rather than $8\,H$ for the `GML` solver, and hence base boxes are physically larger (the motivation for this is described in Section 5.1.1). The integration has been broken up into four steps with the relative time of each step $\chi = T_i/T_{tot}$ displayed in Fig. 5.22. Each step is executed in parallel with `OpenMP` with the exception of the near field step, which is carried out on the GPU with the near field interaction matrix. As shall be detailed later, the

evaluation of the FP solver within the input prep and output prep steps does not occur in parallel.

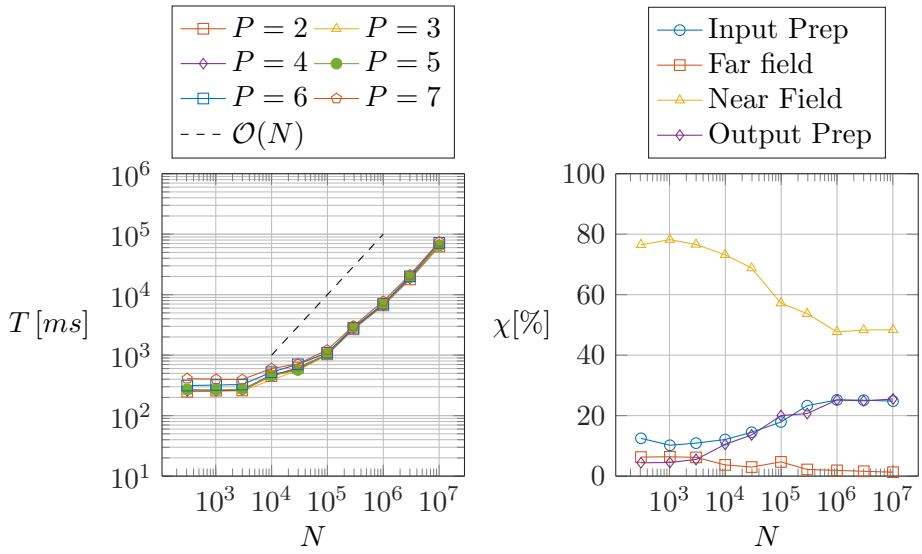

Figure 5.22: Left: Calculation time as a function of particle size for different polynomial orders: The dashed line represents an $\mathcal{O}(N)$ complexity. Right: Relative time for calculation steps ($P = 5$).

Input preparation: This step is much more involved than for the `GML` solver. Initially source and probe binning is carried out. Anterpolation to the base boxes source nodes is then carried out. Thereafter, the sources are mapped to the Eulerian grid. The first FP computation is executed and the boundary forcing $\vec{\gamma}$ is calculated from the homogeneous potential $\vec{\psi}_0$ for use in the near field step.

Far field calculation: This is essentially identical to the `GML` case, however in this case the SF interaction template is being calculated which requires only three large matrix multiplications. This is seen in Fig. 5.22 to constitute a minimal portion of the calculation.

Near field field calculation: The interaction template calculated in the pre-processing step is used to express the near field calculation as a very large matrix multiplication. In this case the matrix multiplication is seen to dominate the calculation expense. Calculations with a more performant GPU were seen to greatly reduce the expense of this step.

Output preparation: The second FP computation is executed and finite differences are calculated within each base box. This constitutes the majority of the expense in this step. Optimisation of the FD calculation for example on the GPU would likely greatly improve performance here.

Monolithic PML Solver

The performance of the monolithic PML solver is shown in Fig. 5.23. The procedure here is similar to that carried out for the PML solver. The resolution of $\vec{\psi}$ is achieved however on a single monolithic Eulerian grid, to and from which $\vec{\omega}$ and $\vec{\psi}$ must be transferred from the individual boxes. Here, the desired optimal scaling is again observed, however the computation time is seen to be significantly lower than for the other solver configurations. This is primarily due to the reduction in expense achieved by utilising the FP grid solver. The integration has been broken up into four steps with the relative time of each step $\chi = T_i/T_{tot}$ displayed in Fig. 5.23. As a global boundary solve is executed here there is no need for a near field evaluation. The solution of the boundary condition $\vec{\psi}$ is conceptually similar to the far field evaluation of the previous solvers as this is performed with the MLMIC method.

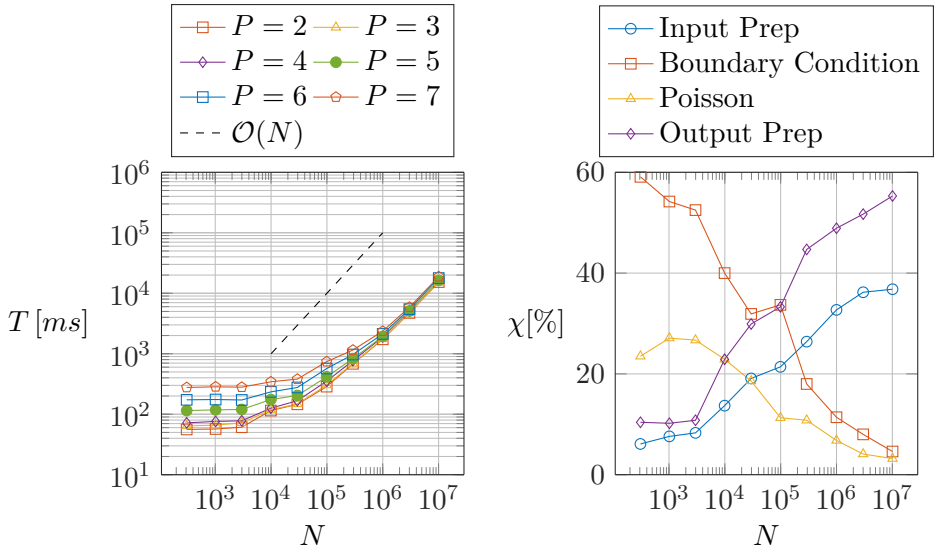

Figure 5.23: Left: Calculation time as a function of particle size for different polynomial orders: The dashed line represents an $\mathcal{O}(N)$ complexity. Right: Relative time for calculation steps ($P = 5$).

Input preparation: This is carried out as with the `PML` method, however rather than carrying out the first FP solve, $\vec{\omega}$ is mapped to the monolithic grid.

Boundary condition: In this step, anterpolation, interaction and interpolation are carried out to calculate the value of $\vec{\psi}$ on the boundary of the monolithic grid. This explains why, for an equivalent particle count N_B is greater for the monolithic `PML` solver as compared to the `PML` solver in Table 5.2. Furthermore, beyond a given dimension the number of active boxes N_B becomes smaller than for the standard `PML` solver. This can be explained by considering a monolithic grid with side length N_B. The number of proactive boxes evaluated in the MLMIC method scales for the monolithic solver as $N_{B,S}^2$, as values on the boundary values are evaluated for the BC calculation. In comparison, for the volume in the `PML` solver the number of proactive boxes scales as $N_{B,S}^3$. For large N, this constitutes only approx. 5 % of the calculation time.

Poisson calculation: Fig. 5.23 demonstrates how optimised the FP solver is, as the entire volume calculation constitutes only approx. 5 % of the calculation time for large volumes.

Output preparation: This process is carried out as with the `PML` solver, however rather than carrying out the second FP solve, $\vec{\psi}$ is mapped back to individual boxes and consumes the majority of the calculation time.

The analysis demonstrates that there exists significant optimisation potential for the use of the monolithic solver, provided the input and output steps are carried out on the GPU rather than the CPU. An important observation here is that for a large N, the expense of the boundary condition evaluation is equivalent to that for the FP solver.

5.2.4 Scalability

Results presented previously are based upon execution on a device with both CPU and GPU nodes, which allows for the optimisation of key processes. The results presented here are carried out purely only multiple CPU nodes the analyse the transferability of results to a HPC architecture. Certain processes within the `VPML` library have **not** been parallelised, eg. the Octtree data structure. These processes generally contribute negligibly to the calculation expense and are not expected to be bottlenecks within

the range investigated here. Two forms of scaling with multiple processors are commonly investigated for a problem with N_c nodes and U unknowns: **Weak scaling**, where U is increased proportionally to N_c and **strong scaling**, where the U is held fixed for variable N_c [76]. In the `VPML` library, box operations are carried out in parallel over N_c nodes and the box count is proportional to the grid size $H_b = n\,H$. A given problem hence scales with the number of resources, corresponding to weak scaling [96]. Let $T(N_c, U)$ be the calculation time necessary to solve a problem of U unknowns using N_c cores, then the weak scaling η_{weak} is given by:

$$\eta_{\text{weak}} = \frac{T(N_c^{ref}, U^{ref})}{T(N_c, SU^{ref})} \quad , \qquad S = \frac{N_c}{N_c^{ref}} \, . \tag{5.11}$$

The ring geometry used in the previous test cases is unsuitable as U^{ref} does not scale linearly with N_c due to the geometry. For this reason the four-vortex wake system described in Chapter 6 has been used here and configured to ensure linear scaling. Results are shown in Fig. 5.24. The `GML` solver is

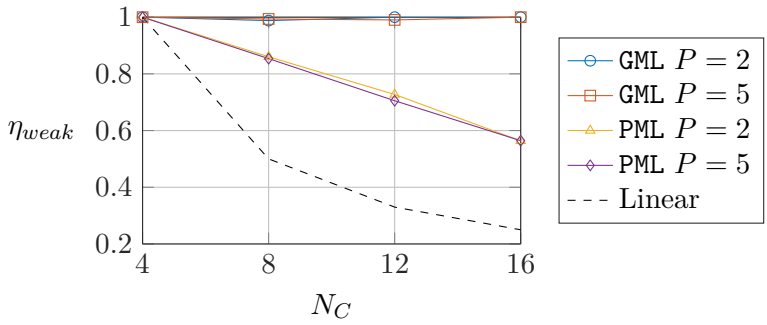

Figure 5.24: Weak scaling of the problem based on $N_c^{ref} = 4$. Dashed line represents a serial problem scaling.

seen to scale optimally. For the `PML` solver however a decrease in efficiency is seen. The bottleneck of the simulation in this case is the FP solver utilised, which was unable to be executed over parallel threads. Within a devoted HPC environment this bottleneck is avoided due to multiple library instances. Despite this, the relatively low cost of the FP solver over each box is seen to not be a complete bottleneck as the efficiency is still well above that of serial scaling, represented by the dashed line in Fig. 5.24. The use of hyperthreading was observed to reduce performance when $N_c > 16$ was used.

5.2.5 Solver Comparisons

In order to identify optimal application cases the solvers are compared here. The static and dynamic memory demands of the solvers are first inspected, followed by simulation times. Based on the results of this section a general summary is provided.

Memory Overhead

The run-time memory requirements of the solvers has been investigated for a range of problem sizes. The results here are valid for single-precision floating point accuracy and are displayed in Fig. 5.25. The memory consumption is broken up into static and dynamic memory overhead.

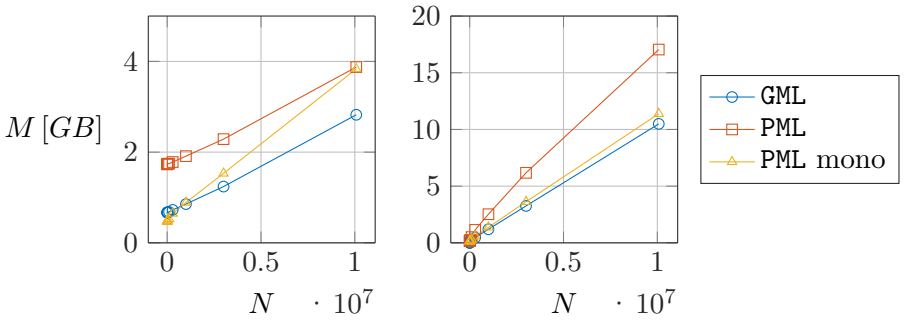

Figure 5.25: Memory requirements for a range of problem sizes. Left: Static memory, Right: Dynamic memory.

Static overhead This includes memory allocated in pre-processing, prior to the solution step. It is seen on the left hand side of Fig.5.25 that the `GML` solver has the lowest static overhead. In this case the static overhead is simply the three interaction templates of the BS kernel. The overhead increase with problem size for all three solvers is due to storage of the particle set data. Although the `PML` solver only requires a single interaction matrix for the SF kernel, a much larger initial overhead is required due to storage of the near field interaction matrix necessary for the James algorithm. In the case of the monolithic Poisson solver the monolithic grid is allocated in the pre-processing step, indicating why the overhead grows and eventually becomes expensive for greater particle sets.

Dynamic overhead This is allocated during an integration step. It includes the storage of all MLMIC tree elements, source and receiver grid

node data and an/interpolated data values. On the right hand side of Fig. 5.25 the GML and PML monolithic are seen to have linearly increasing overhead. This is a logical progression as the problem size directly influences the box count N_b, which for uniformly distributed particles grows linearly with particle set size for a given minimum box size $H_b = n\,H$. The PML solver is seen to consume approximately double the memory. This is well explained due to the implementation of the PML solver, which for each base box allocates two Eulerian grids: one for solution of $\vec{\psi}_0$, and a second for the full solution $\vec{\psi} = \vec{\psi}_0 + \vec{\psi}_1$ as described in Chapter 4.

Simulation Time

The total simulation time for a single integration for the implemented solvers and direct evaluation is compared for two polynomial orders in Fig. 5.26. In general it seen that direct evaluation is the most efficient method for $N < 10^5$. This limit is certainly hardware specific and should be expected to increase as general GPU technologies improve. Above this count the unfavorable $\mathcal{O}(N^2)$ complexity becomes impractical. The GML solver is seen to be more efficient than the PML solver for $N \lesssim$ 3e5. This number however is expected in general to be much lower when the FP solver no longer causes a multithreading bottleneck, in which case the entire PML curve practically will *shift* down by a constant factor. The overhead of the monolithic Poisson solver is seen to be impractical compared to the GML method for $N < 10^4$. For $N > 10^4$ however the monolithic Poisson solver is seen have the best performance.

Summary and optimal solver choice

The choice of an optimal solver is very much application-dependent. If turbulent diffusion is to be modelled, then a PML solver must be implemented, in this case if the overhead memory costs of the monolithic PML solver are not constrained by the hardware, it achieves the quickest result. It is, as outlined in the objectives in Chapter 1, the optimal choice for a single desktop environment with a CPU-GPU combination. If however a HPC cluster is being used, the standard PML solver demonstrates the most potential to reduce calculation time as the procedure is the most amenable to distribution over a high computational node count.

If turbulent diffusion is not being modelled, for particle counts $N < 10^5$, direct evaluation on the GPU produces the quickest results.

It should be additionally noted that in the case the *sparse* particle sets are

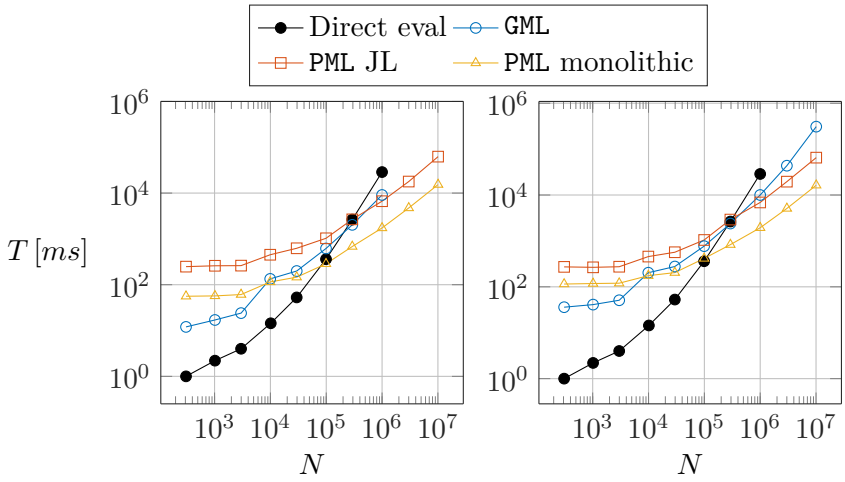

Figure 5.26: Calculation time for VPML solvers and direct evaluation. Left: $P = 2$, Right: $P = 5$.

being modelled, the PML solver option is unsuitable as described previously. In this case either direct evaluation or the GML solver should be applied, the optimal choice is again dependent upon particle set size.

Chapter 6 Flow Solver Application

A number of application cases have been simulated here in order to demonstrate the domain of applicability of the `VPML` library. Following the objectives outlined in Chapter 1, the desired application here is the calculation of the wake of a wind turbine, however the solver should be capable of simulating other aerodynamic flows. For this reason four application cases have been simulated:

1. Spatially developing wake of an elliptic wing
2. Periodic four-vortex wake of an aircraft
3. Spatially developing wake of a horizontal axis wind turbine, and:
4. Periodic wake of a horizontal axis wind turbine.

These cases demonstrate the ability of the solver to handle a range of aerodynamics flows. Furthermore, all simulations have been carried out on a desktop computer, demonstrating the ability of the solver to resolve highly detailed flows without requiring high power computing resources. It should be noted here that the simulated cases do not represent exhaustive studies, but rather are intended to illustrate the domain of application of the `VPML` library.

Treatment of Vorticity Flux

For cases of validation in Chapter 5, an initial vorticity field was specified and allowed to evolve in time. As no circulation was introduced into the system, one expects the net circulation of a material volume containing all particles to remain constant, this was indeed observed numerically. In many cases of practical interest, circulation flux into the system occurs through regions of shear flow. The introduction of a vorticity flux is achieved by introducing the concept of a **spatial** and **material** source elements.

Material source particles are convected with the local velocity field. Position and strength are updated at every time step as per the VTE by treating each source as a probe. All validation cases in Chapter 5 were composed completely of material source particles.

Spatial source particles have prescribed position and strength and are hence not treated as probes. These are not convected and represent the

effect of vorticity flux into the flow domain. Spatial source elements are not remeshed.

For simulations with vorticity flux, the two particle types listed above can be combined to yield a representative vorticity field. The location and strength of the spatial elements is prescribed, this is further detailed in Sections 6.1 and 6.3. Vorticity flux into the flow domain is achieved by generating material elements at regular intervals which are duplicates of the spatial elements. These, unlike the spatial elements are allowed to freely convect, this is illustrated in Fig. 6.1 for a simplified 2D flow.

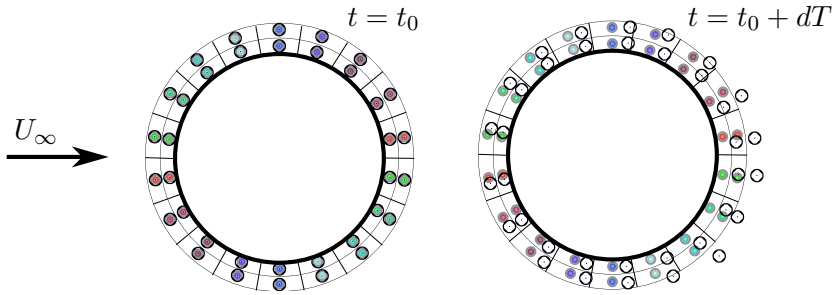

Figure 6.1: An illustrative flow around a 2D cylinder. Vorticity generating regions are represented with spatial source particles (filled points), these are specified and occupy source cells. Vorticity flux into the flow is represented with material source particles (hollow points). Only material elements are convected and updated.

The cells in the flow domain occupied by spatial source elements– hereafter referred to as **source cells**– are marked at the beginning of a time step and they alone specify the vorticity contribution of elements within this cell. This is equivalent to stating that the global vorticity field around a body is dominated by the effect of the body on the flow field. This approximation is valid for many cases of practical interest and avoids numerous numerical issues which result from overlapping material and spatial source particles.

6.1 Case 1: Space-Evolving Wake of a Wing

In this case the ability of the solver to capture the development of the space-evolving wake behind a lifting surface shall be demonstrated. The wing is modelled as an equivalent vorticity source distributed along the y axis in a flow field with freestream velocity $U_\infty \vec{e}_x$. The wing has span $b = 1\,\mathrm{m}$, extends along the y axis between $-\frac{b}{2} \leq y \leq \frac{b}{2}$ and has the circulation distribution $\gamma(y)\,\vec{e}_y$.

Specification of circulation γ It is assumed that the wing has an elliptic planform and constant geometric angle of attack $\alpha(y)$. Under the assumption of potential flow and thin airfoil theory, the classical lifting line theory due to Prandtl [133] gives for the circulation distribution:

$$\gamma = \Gamma \left[1 - \left(\frac{2y}{b}\right)^2\right] \quad , \quad \frac{\partial \gamma}{\partial y} = \frac{4\Gamma}{b^2} \left[1 - \left(\frac{2y}{b}\right)^2\right]^{-1} . \tag{6.1}$$

The lifting line here is here approximated by a set of vortex segments aligned along the y axis with strength $\gamma(y)\, dy$ where dy is the length of the segment. The wing segments are distributed with a cosine spacing in order to improve resolution near the blade tips, where the trailing vorticity is greatest. An inspection of the Laplace equation for potential flow (the mathematical expression of Helmholtz' second theorem) implies that any variation in circulation generated along the spanwise direction is accompanied by an equivalent change in streamwise direction. In this case this is a vorticity shed into the wake with magnitude $\frac{\partial \gamma}{\partial y}\, dy\, \vec{e}_x$. For the simulations here the wing circulation has been chosen to have unit strength, $\Gamma = 1\,\mathrm{m^2 s^{-1}}$.

Spatial and material source generation This distribution is sufficient for the description of the `GML` particle distribution, where each segment introduces an equivalent vortex particle into the flow region. For the `PML` method, a continuous distribution of vorticity is desired. This is achieved by performing a spatial convolution over the lifting line with an appropriate kernel, a 2D Gaussian kernel has been chosen here:

$$\omega(y,z) = \frac{1}{2\pi\sigma^2} \int_{-b/2}^{b/2} \gamma(Y) \left[\exp\left(-\frac{(y-Y)^2 + z^2}{2\sigma^2}\right)\right] dY \,, \tag{6.2}$$

where this is carried out sequentially on each vortex element. This method allows a generalized vortex filament to be represented as a volume distribution. Both representations are visualised in Fig. 6.2. It is observed that the segment distribution for Eq. (6.2) does not need to be uniformly distributed, allowing an improved convolution in regions of higher gradient, such as near the wing tip.

Simulation parameters The wake has been simulated with both the `GML` and `PML` solvers. For the `GML` solver, the lifting line alone has been used to generate the source particles. Here no remeshing or vorticity divergence filtering has been carried out. For `PML` solver, source particles are generated with the convolution of Eq. (6.2) with a Gaussian smoothing parameter

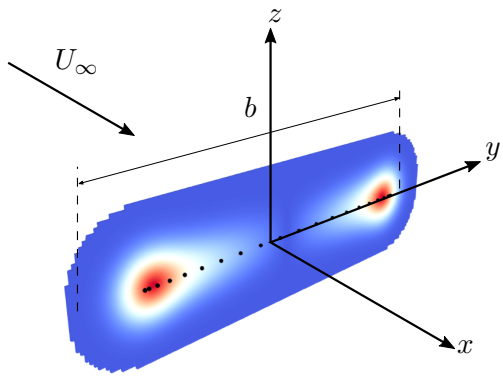

Figure 6.2: The wake vorticity distribution of an ideal elliptical wing. Spatial source elements for the GML solver are shown with points. For the PML solver the continuous distribution achieved through convolution (6.2) is also displayed.

$\sigma = 0.05\,b$. The problem here is treated to be symmetric about the y axis. Employing the symmetric treatment described in Section 4.1.6 hence reduces computational expense by a factor of two. Material particles are generated every 10 time steps. The wing remains stationary in an inflow field with $U_\infty = 2.5$, 5, and $10\,\mathrm{ms}^{-1}$. Material particles are generated every ten time steps, $N_{gen} = 10$ and the simulation time step dt is chosen such that a material source particle is convected a distance dx approximately one cell distance away H from the spatial source positions every N_{gen} time steps: $dx = H \approx dt\,U_\infty\,N_{gen} \rightarrow dt = H/(U_\infty\,N_{gen})$. The cell grid size is chosen to be $H = 0.02\,\mathrm{m}$. This gives a resolution in spanwise direction of 50 elements. Particle remeshing is carried out only with the PML solver and with the M_2 mapping procedure.

Wake development

Here the results for a range of inflow velocities are presented and the behaviour of the two models is investigated.

Velocity induced by the wing To demonstrate the accuracy of the induced velocity, a case was generated which mimics the assumptions of the Prandtl lifting line theory. In this theory the wake is convected only in x direction and extends to $x = \infty$ in the plane $z = 0$ behind the wing. The particles therefore do not convect in the z of y-directions. A wake sheet is composed of infinitely many *horseshoe* vortices. Under this assumption an analytical result for the induced downwash U_z can be found [134]. In the case of an elliptical wing, the analytical value of U_z is constant and

has value $-\Gamma/2$. The downwash over the semi-span is shown in Fig. 6.3. It

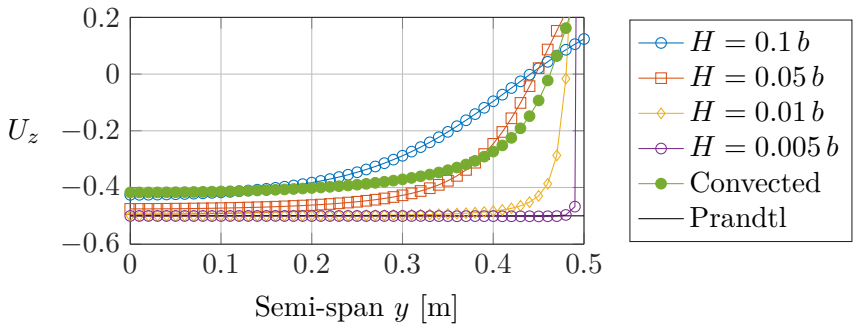

Figure 6.3: The induced downwash due to the idealised wake treatment of the Prandlt lifting line theory. The prediction of the `GML` method for numerous grid sizes are shown. The case of a freely convected wake is also displayed.

can be seen as the grid resolution is increased the induced velocity indeed approaches that of the analytical solution. The influence which is seen by allowing the wake to deflect under self-induction for the case $H = 0.02 \cdot b$ is also shown. This is seen to reduce both the average downwash and and the downwash in the tip regions.

Tip roll-up The large reduction in circulation at the blade tips implies that the trailing vorticity shed into the wake is strongest in the tip region. It is thus expected that the wake sheet rolls up on itself in this region. The behaviour of the wake is shown in Fig. 6.4 for the three different freestream velocities $U_\infty = 2.5$, 5 and $10\,\mathrm{ms}^{-1}$. The motion of the wake is demonstrated for both solvers here. For the `GML` solver discrete particles are seen as remeshing is not carried out and the particles are generated only along the lifting line. For the `PML` solver, the field is continuously represented. For this reason, contours of constant vorticity have been extracted for visualisation. The stronger core seen for $U_\infty = 10\,\mathrm{ms}^{-1}$ is explained through the shorter convection time to the given x plane in the image. Insufficient time has passed for the viscous core to diffuse out, as compared to the core for $U_\infty = 2.5\,\mathrm{ms}^{-1}$, where the convection time is four times greater. Furthermore, the greater convection time for $U_\infty = 2.5\,\mathrm{ms}^{-1}$ is seen in the `GML` output. Here the weak center wake sheet is convected further in negative z direction due to the downwash induced by the tip vortices.

Starting vortex In a realistic scenario, the lifting vortex does not instantaneously develop, but slowly increases in strength during take-off. Much

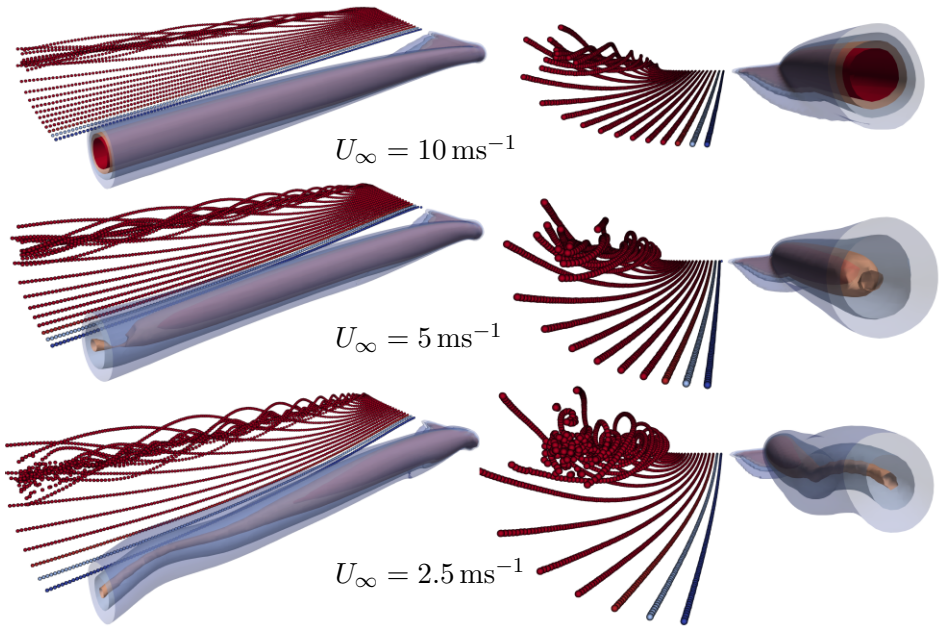

Figure 6.4: Wake roll-up. Left: Oblique view. Right: View looking down negative x axis (upstream). Vorticity contours are equally scaled. Each image demonstrates the results of the `GML` solver (discrete particles) and the `PML` solver (surfaces extracted from the vorticity distribution).

as with the spatial variation of the vorticity over the span, the change in vorticity is shed into the wake of the aircraft, this leads to a *rolling-up* of the starting vortex. This is simulated in the simulations here by adding a vorticity component to the spatial source particles in the spanwise (y) direction with magnitude equal to the trailing vorticity magnitude in the initial stages of the simulation. The formations of the starting vortex is shown for two velocities for the `PML` solver in Fig. 6.5. This was also observed for the `GML` solver, however the discrete particle treatment led to a less continuous starting vortex sheet.

Oscillating wing As an illustrative example, the strength of the vortex was varied sinusoidally in time with a full cycle occurring every 1000 time steps. Analog to the case previously described, any change in spanwise vorticity must be accompanied by a change in the shed streamwise vorticity, this is accomplished by a vorticity component in the y direction, which gives rise to a *von-Karman* style vortex street pattern. This is shown in Fig. 6.6 for the initial simulation stage. It is seen that both solvers gives rise of a repeated folding of the tip vortices due to the oscillating spanwise

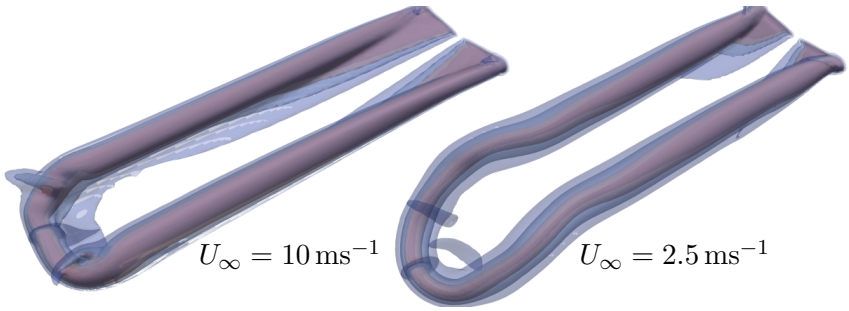

Figure 6.5: Starting vortex development.

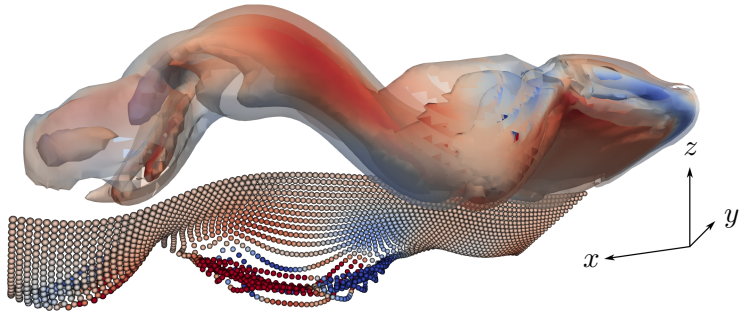

Figure 6.6: Wake of an oscillating wing. For the time step shown the wake has extended $2b$ downstream of the wing. The particle field representation from the `GML` solver is shown for $y < 0$ and contour plots of the vorticity from the `PML` solver are shown for $y > 0$.

circulation being shed. This flow was simulated with the `PML` until the wake had reached a distance $10\,b$ downstream of the wing. Contours of vorticity for this case are shown in Fig. 6.7. It is seen that the folded regions of the tip vortex eventually form together in viscous merging processes, forming ring-like structures with their axis parallel to the freestream direction. This is conceptually similar to the case simulated in Section 5.1.4.

6.2 Case 2: Four-Vortex Aircraft Wake System

In this section the ability of the method to treat high resolution periodic problems shall be demonstrated. An optimal case for this is a converged aircraft wake, as the steady pattern formed can be well approximated with a periodic flow. For the case of a two-vortex system, the well known Crow instability is known to occur [78]. The slightly more complicated system

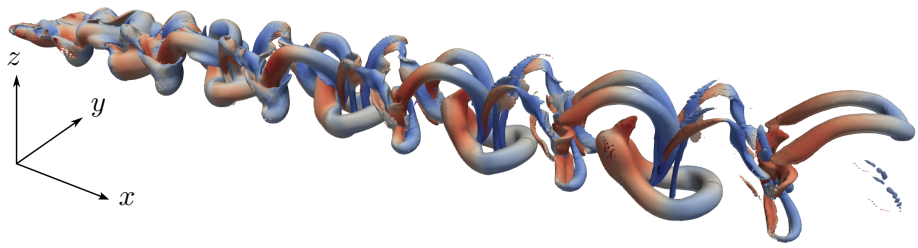

Figure 6.7: Wake oscillations $10b$ behind a sinusoidally oscillating wing. Vorticity magnitude contours of the PML solver are shown with colour scale based on ω_z magnitude.

investigated here consists of a counter-rotating four vortex system. In this case not only the long wavelength Crow instability arises, but additionally a medium wavelength and short wavelength (Widnall) instability also arises [80]. The flow field is an approximation of the flow field behind an aircraft and is as described in Fabre et al. [81]. It was found in initial testing that the use of discrete filaments (filament in this sense refers to a *line* of particle, in alignment with the GML methodology) acted to amplify certain unrealistic modes (this effect was seen elsewhere– see Winckelmans et al. [82]). This effect can be overcome with the application of suitable filtering, this however was not done here as vortex filament stability analysis is less illustrative of the capabilities of the developed solvers. For this reason, the results presented here pertain only to the use of the PML solver.

Flow description The dominant outer vortices are characterised by a circulation Γ_1 and separation distance b_1, the inner vortices by a circulation Γ_2 and separation distance b_2, as shown in Fig. 6.8. At initialisation, the inner vortices are driven downwards by the much stronger outer vortices. For the counter-rotating configuration, where $\Gamma_2/\Gamma_1 < 0$, the inner vortices enter into orbits around the larger outer filaments. A set of natural reference scales for the system can be defined as follows: Circulation $\Gamma_0 = \Gamma_1 + \Gamma_2$, mean vortex separation $b_0 = (\Gamma_1\, b_1 + \Gamma_2\, b_2)/\Gamma_0$ vortex mean descent velocity $U_0 = \Gamma_0/(2\pi\, b_0)$ and mean timescale $T_0 = b_0/U_0$.

The inner and outer vortices have core sizes σ_2 and σ_1, respectively. The parameters chosen here are $\Gamma_2/\Gamma_1 = -0.3$, $b_2/b_1 = 0.3$, $\sigma_1/b_1 = 0.075$ and $\sigma_2/\sigma_1 = 2/3$ following numerous references in the literature [81, 82, 28] which demonstrate that these parameters lead to rapidly growing instabilities. Filament vorticity is distributed initially with an algebraic core $\omega_x(r) = \frac{\Gamma}{\pi}\frac{\sigma^2}{(r^2+\sigma^2)^2}$ where r is the distance to the filament centre. The periodic length

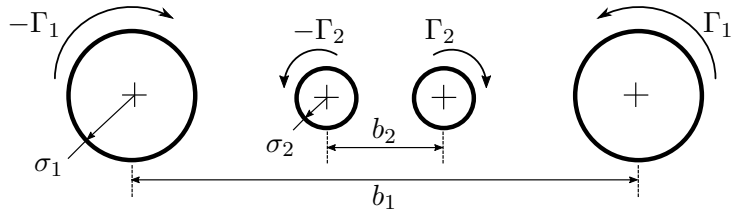

Figure 6.8: Geometry of the model four-vortex wake system shed by an aircraft.

of the domain in the x direction is L_x.

Demonstration of periodicity The periodicity is demonstrated by inspecting the induced velocities along a vortex filament. In the case of a periodic representation, the velocity should be independent of downstream position x due to the 2D nature of the system. This is demonstrated in Fig. 6.9 where it is seen that the induced velocity becomes constant with increasing periodic reflections N_{per}. It is recommended in Cocle et al. [28] that $N_{per} = 50$ is sufficient to remove any far field effects, however in the work here $N_{per} = 30$ proved to be perfectly sufficient.

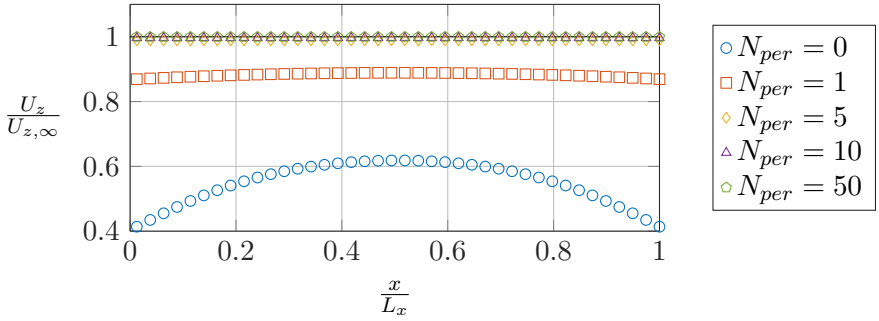

Figure 6.9: Induced velocity along an outer vortex

Simulation parameters The geometric parameters defined above allow for the complete description of the initial flow field, provided any of the length parameters is explicitly specified. In this case the periodic domain length L_{per} is used, as this is a modelling parameter for the periodic solver. This factor is specified with the number of base boxes in the periodic direction within the periodic domain NB_{per}. The simulation parameters as described in Section 5.1.1 have been used with a grid spacing of $H = 0.02\,\text{m}$.The filaments were specified by defining initially the core centre and then extruding this in the periodic direction. The initial core position was perturbed sinusoidally parallel to the periodic direction with the function

$dy = g\sin(fx)$, where the wave number f corresponds to the dominant instability $f = 6.39/b1$ and $g = b_1 \cdot 10^{-6}\,\mathrm{m}$.

Wake Evolution

Two configurations have been investigated, a truncated domain and a full length domain. These are discussed separately.

Truncated domain- Medium wavelength instability: The periodicity and stability of the configuration is checked by initially carrying out a truncated simulation at the most unstable wavelength for this problem. This was found to be the medium-wavelength instability, which has a wavelength of $L_x = 0.983\,b_1$ as described in Winckelmans et al. [82]. The number of base boxes in periodic direction is set to $NB_{per} = 5$. This gives a resolution of 80 elements in periodic direction. The evolution of the particle field is given for multiple time steps in Fig. 6.10. These results agree qualitatively

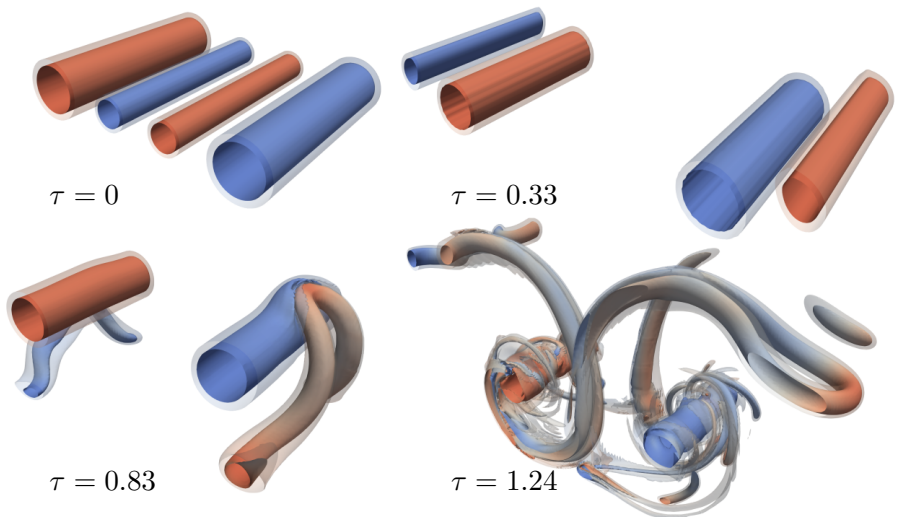

Figure 6.10: Simulation of medium wavelength instability for a single wavelength. Here no turbulence model was used. Two isocontours of vorticity are shown. Normalized time $\tau = t/T_0$ shown.

very well with those presented in Chatelain et al. [83], where a much higher resolution was applied. It is observed that the weaker inner vortices rotate around the stronger outer vortices until the perturbation grows and a segment of the weaker vortex becomes entrained in the stronger vortex. The process is characteristic of the nonlinear evolution of such an instability and the resulting *hooks* which remain are referred to as Omega-loops, due

to the similarity with the Greek letter. This is will discussed for the full periodic domain in the following test. If allowed to evolve further, the smaller vortices are eventually consumed by the larger vortices with only tailing regions of vorticity remaining. This is comparable to the process of 2D viscous vortex merging processes where one (stronger) vortex dominates and consumes the second [123]. A Fourier analysis of the modal energies has been carried out on the flow field with essentially the same procedure as described in Section 5.1.5. In this case the integration is carried out in axial (periodic) direction, as opposed to the azimuthal direction for the vortex ring. This allows the identification of the modal energy growth of the developing instabilities. This is shown in Fig. 6.11. It is seen that the medium wavelength instability corresponding to $m = 1$ quickly grows beyond $\tau > 0.5$.

Full periodic domain In this case a much larger periodic domain has been taken. This domain length corresponds to the long wavelength Crow instability of the corresponding two-vortex system: $L_x = 8.53\, b_0$. In this case $NB_{per} = 45$ was taken to ensure adequate resolution in periodic direction. The `HV` turbulence model has been enabled with Smagorinski constant $C = 6.8$ and global time constant T_0 following Cocle [76]. The evolution of this system is shown in Fig. 6.12. In this case numerous instabilities are

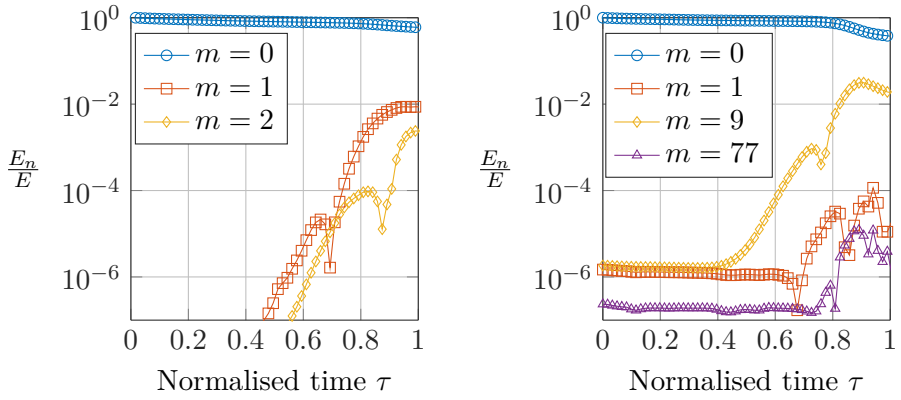

Figure 6.11: Relative modal energy of the four-vortex wake system simulations. Left: Short domain corresponding to the medium wavelength instability. Right: Long domain corresponding to the long wavelength Crow instability. Normalized time $\tau = t/T_0$ shown.

seen to occur. The medium wavelength instability seen in the previous test case also occurs here. In this case the short-wavelength Widnall instability is also seen to occur within the waves of the medium-wavelength instability,

particularly during initial growth at $\tau = 0.75$. This mode quickly reaches saturation and the flow becomes highly turbulent by $\tau = 1.19$. The very high degree of filamentation here is undoubtedly a result of the specification of the hyper-viscosity factor C to 6.8, which pushes the dissipation incurred by the turbulent breakdown to smaller scales. This is observed furthermore in the spectral analysis of the flow, where the Widnall mode, corresponding to $m = 77$ is seen in Fig. 6.11 to grow around $\tau = 0.7$. The dominant mode remains however the medium wavelength instability, which for this domain length corresponds to $m = 9$. It is useful at this stage to observe that the Omega loops observed in these tests cases are also experimentally observed. Shown in Fig. 6.13 are experimental visualisations performed in a towing tank which demonstrate the occurrence of the omega loops in the wake of a simple lifting body.

6.3 Case 3: Horizontal Axis Wind Turbine

For this case the solver will be applied to simulate the wake of a horizontal axis wind turbine. A laboratory-scale experimental turbine is investigated. A number of important flow features shall be shown to be captured by the `VPML` library methods. In the cases simulated, the particle count using the `GML` solver was relatively low, for this reason direct evaluation using the GPU within the `VPML` solver was applied, this shall be discussed later. The ability of the `PML` solver to model turbulent shear stresses in the wake shall be demonstrated.

MexNext experiment The turbine investigated is the experimental turbine of the *MexNext* project [136]. The experiments used as a comparison here were carried out in 2014. The turbine in operation is illustrated in Fig. 6.14. These experiments were a continuation of the 2006 MEXICO (Model EXperiments in COntrolled conditions) project carried out in the same wind tunnel in 2006 [137, 138]. This provides an ideal comparison case as a relatively complete turbine definition was provided which allows for numerical reproduction of the experiment. A range experimental data was data were collected including wake velocity measurements, on-blade pressure measurements and blade and rotor forces. The blade normal forces F_x will be used to approximate the blade circulation, which then allows specification of the shed wake vorticity. These experiments were carried out in the Large Scale Low Speed Facility (LLF) of the German Dutch Wind Tunnels (DNW). The turbine is operated in an open test section. This greatly reduces blockage effects, estimated to be less than 1% [136].

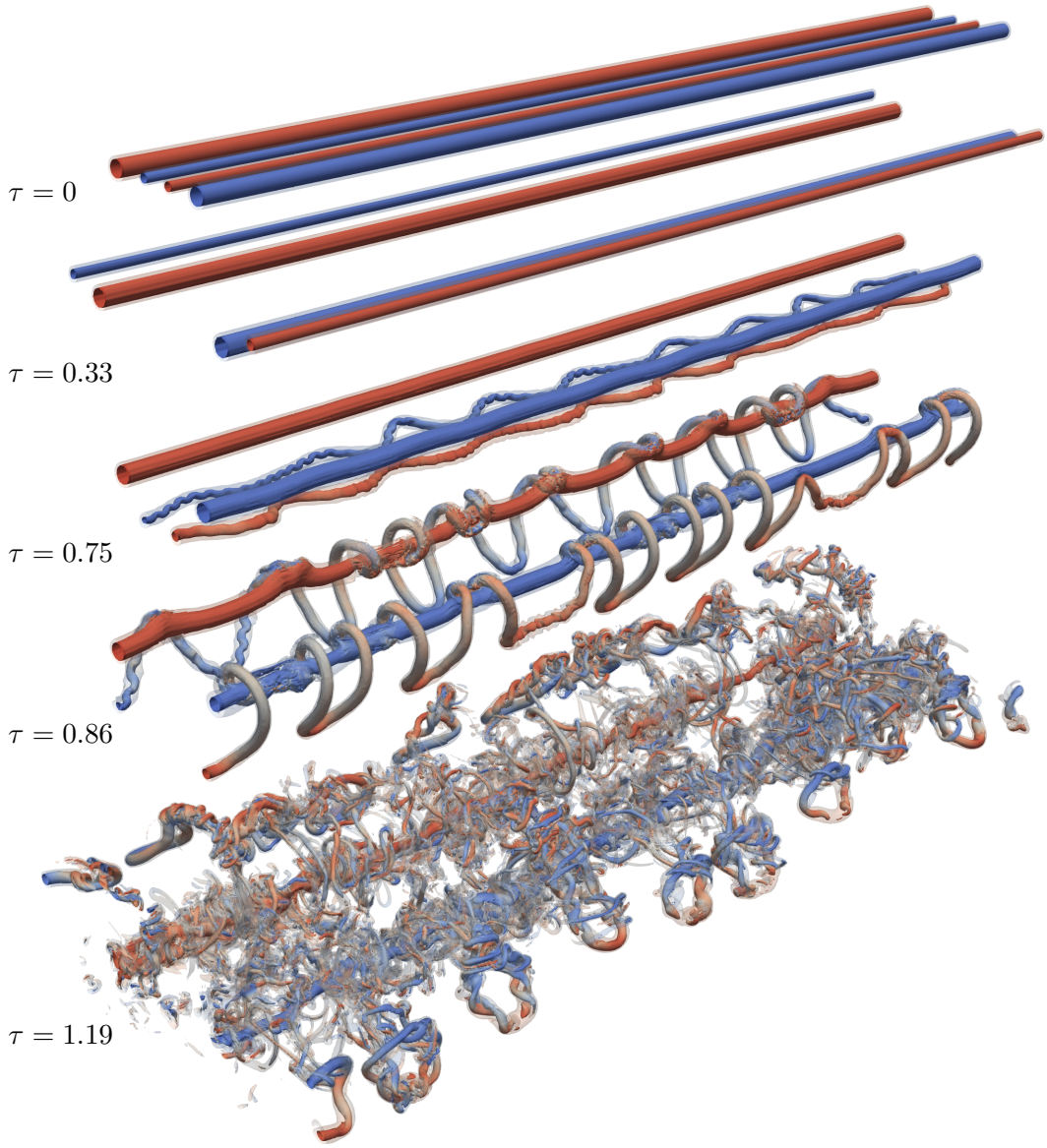

Figure 6.12: Simulation of the medium wavelength instability of a four-vortex aircraft wake. The domain length in this case corresponds to the wavelength of the long wavelength Crow instability.

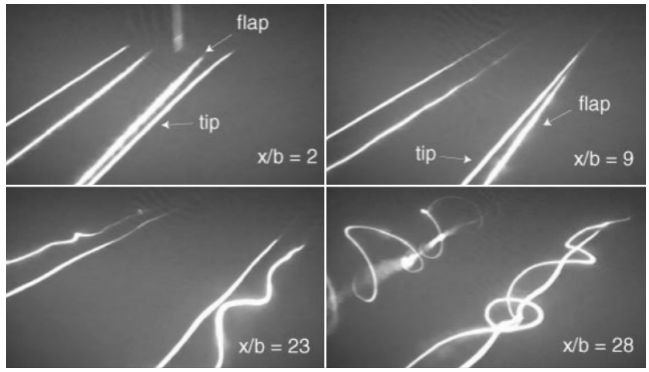

Figure 6.13: A visualisation performed with luminescent dye in the wake of a lifting body within a towing tank from Ortega et al. [135]. The Omega-loops of the inner vortices are reminiscent of those seen in Fig. 6.12.

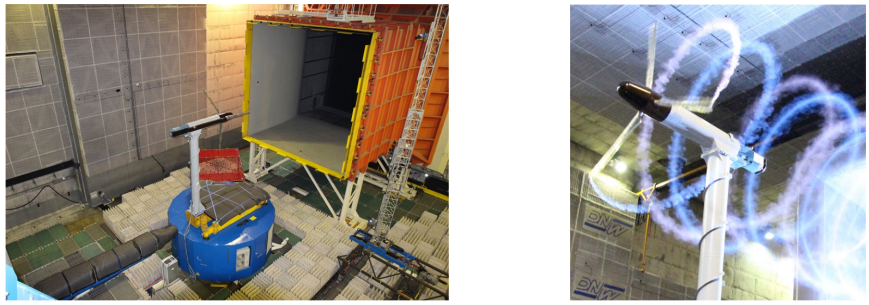

Figure 6.14: Left: The MEXICO rotor within the open test section. Right: Smoke visualisation of the tip vortices at TSR = 4.44, taken from [64].

Rotor description The MEXICO rotor has a tip radius $R = 2.25\,\mathrm{m}$ and a root radius of 0.25 m. The blade is composed of three distinct wing profiles, visualised in Fig. 6.15. These airfoils are not directly modelled within `VPML` but they shall be used for extraction of the blade circulation, described in the following section. The blade chord and twist distribution are shown in Fig. 6.16. The chord distribution is again used only for calculation of a suitable circulation distribution, the twist however is used to rotate the circulation distribution out of the rotor plane.

The turbine was operated at a range of tip speed ratios TSR, defined as $\mathrm{TSR} = R\Omega/U_\infty$ where Ω is the rotation rate of the rotor and U_∞ is the freestream velocity. For consistency with the experiments conducted, the rotational rate of the simulations is held constant at $\Omega = 425.1\,\mathrm{RPM}$. The tip speed ratio is modulated by modifying the inflow speed U_∞. The cases simulated here have been summarised in Table 6.1.

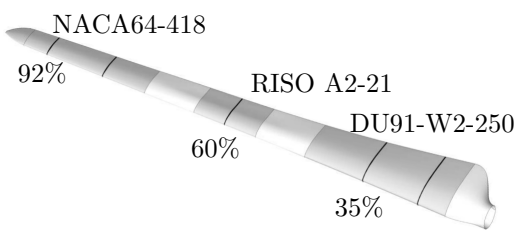

Figure 6.15: The MEXICO rotor blade. Dark regions indicate constant airfoil profile. Light regions indicate transition regions, taken from [108].

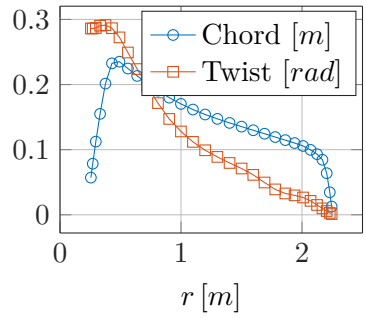

Figure 6.16: MEXICO blade chord and twist distribution.

Table 6.1: Simulated cases of the Mexico rotor.

TSR	$U_\infty\,[ms^{-1}]$	$\Omega\,[\mathrm{RPM}]$
4.1647	24.05	425.1
6.6509	15.06	425.1
9.9664	10.05	425.1

Treatment of blade within `VPML` For consistency with the formulation of the library, each blade is treated with an equivalent vorticity sheet as illustrated in Fig. 6.17. A convolution is carried out along the blade axis in order to produce a vorticity sheet representative of the trailing vorticity shed by each blade. It is assumed for simplicity that the shed vorticity is aligned tangent to the blade chord, as such the vorticity sheet is generated parallel to the freestream (x axis) direction. This is rotated at each time step with the rotational velocity of the rotor. The blade axis is defined with the same geometry as the experimental rotor. Specification of the circulation is required in order to carry out the convolution with Eq. (6.2).

The circulation distribution was calculated by generating a representative turbine definition within the open-source wind turbine simulation suite QBlade [139, 140]. Within QBlade a vortex filament lifting line free vortex wake (LLFVW) simulation can be carried out using vortex filaments. These have been shown to represent very well the induction due to the wake of a turbine [141, 142]. The LLFVW calculates blade loads by accounting for velocities induced by neighboring blade elements and wake filaments together with tabulated airfoil data to calculate the blade circulation distribution. The aerodynamic definition of the rotor was specified with the geometric and polar data provided within the MEXICO data set [136]. A time-domain simulation was then carried out until convergence in order to extract steady-state force values. Of the available experimental data for comparison, the

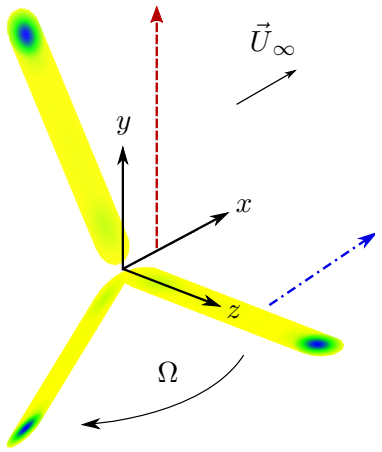

Figure 6.17: The material source planes which introduce vorticity into the simulation. The traverses used later for the velocity comparison are also shown. Direct behind the rotor is the radial traverse (red,dashed) and running downstream is the axial traverse (blue, dash-dot).

blade normal force appeared to be the most reliable, for which reason this was used in the comparisons. These forces were only recorded at five positions along the blade, as such the wake and polar parameters were adapted in order to achieve a visual best-fit for the data points, shown in Fig. 6.18. The blade circulation was then extracted for these conditions.

The circulation is then used within Eq. (6.2) in order to generate the vorticity sheet. It should be noted that for lower order methods (e.g. vortex filament methods) the core size σ of the convolution is optimally chosen as a function of the local chord length of the turbine [143]. This is seen here to influence the results significantly, this topic will be further detailed in the following discussion. The circulation distribution itself is not used, but rather the gradient $\partial\Gamma/\partial s \cdot ds$, where s is the coordinate in spanwise direction. This gradient was calculated from the circulation distribution with spline interpolation. This distribution must furthermore be multiplied with the factor $r\,d\theta$, where $d\theta$ is the segment of arc subtended by the blade in a time step. This ensures kinematic consistency of the vorticity shed in to the wake. It is observed in Fig. 6.17 that the small chord length and high vorticity gradient near the tip (see Fig. 6.18) leads to a strong tip vortex, as is physically expected.

Simulation parameters For the simulations here the grid resolution was set to $H = 0.07$, resolving the blade with approximately 30 elements in the

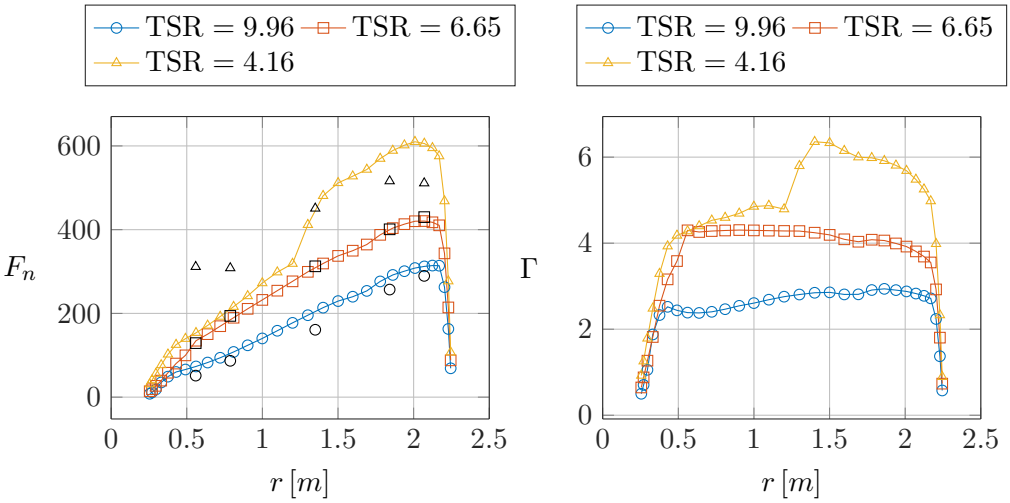

Figure 6.18: Left: Blade normal force values compared to experimental data (black markers). Right: Corresponding blade circulation distribution.

spanwise direction. The time step was chosen such that every ten time steps the rotor has rotated through 1°, a single rotation is therefore composed of 3600 time steps. The simulations were run for 20 rotations. Particle set remeshing, divergence filtering and magnitude filtering occurred every ten timesteps. New spatial source particle were also seeded into the simulation every ten timesteps. The blade circulation is initially linearly ramped up over the first 1.5 rotations of the simulation to avoid spurious initial transient wake effects. The `PML` solver was used for the simulations here. The kinematic viscosity was set to that of air $\nu = 1.5571 \cdot 10^{-1}\mathrm{m}^2\mathrm{s}^{-1}$. All other solver parameters as are specified in Table 5.1. The `RVM` turbulence model has been activated with a second-order small-scale filter and corresponding filter constant $C_r^{\{2\}} = 0.0476$ [76].

Qualititative Wake Development

The results for the `PML` method for each inspected tip speed ratio are inspected first. For these tests the James-Lackner method has been used to treat the near field interactions within the `PML` solver. The results of both solvers are then detailed in the following discussion section.

TSR 4.16 At the lowest TSR it is expected that the wake interaction is relatively weak due to the larger spacing between wake filaments as caused by the higher inlet speed (larger pitch). This is indeed observed in Fig. 6.19.

The tip vortices are seen to decay due to the action of viscous diffusion. The effect of the induction of the wake and rotor is seen to significantly decrement the flow velocity in the wake region. In reality, it would be expected that due to the very high angles of attack experienced by the blades in this operating state that significant regions of separation along the blade would be observed. This is somewhat predicted by the lifting line method as seen in the circulation distribution in Fig. 6.18. The `VPML` solver currently has no facility to account for this. This effect would also increase the aerodynamic blockage of the turbine, influencing the velocity distribution in the wake.

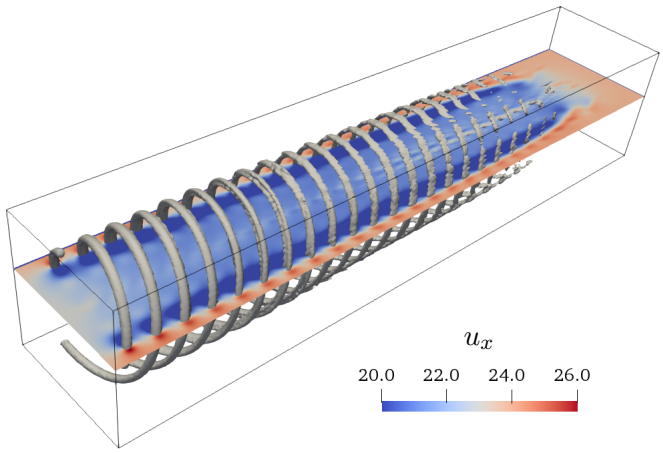

Figure 6.19: Velocity field of the rotor for TSR 4.16 using the `PML` solver. The contour shown is an isocontour of vorticity.

TSR 6.65 A visualisation of the flow field simulated by the `PML` solver is shown in Fig. 6.20. A number of phenomena can be observed here. The effect of the root vortex is seen to counteract the deceleration caused by the blades. This of course is purely a result of the fact that in the numerical simulations the nacelle is not modelled. In reality there is a dead water region aft of the nacelle and there would be separation at the root regions of the blade, it would hence be expected that the flow in this region would be highly disturbed. The second observation to be made is the apparent transition of the flow. The entire wake regions appears to transition in the region $D < x < 4D$ and visually appears to be fully turbulent for $x > 4D$. Although visually appearing accurate, it must be stated here that the ratio of turbulent kinetic energy was not monitored, as such it is quite likely

that this represents a numerical artifact of the actual turbulent flow, this is commonly referred to as under-resolved LES. It was furthermore observed that this transition region was a strong function of the grid resolution H, as such the filtering procedure appears to be quite grid dependent. As the purpose of this section is simply to illustrate the efficacy of the numerical model, a detailed study here is certainly desired to establish how physical this turbulent region is. A detailed cross-section of the velocity field is

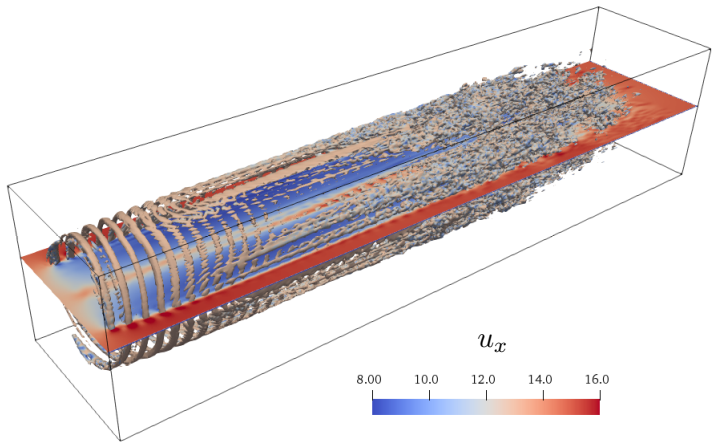

Figure 6.20: Velocity field of the rotor for TSR 6.65 using the PML solver. The contour shown is an isocontour of vorticity.

shown in Fig. 6.21. Here the entrainment of the irrotatational outer flow is seen to be occurring in the turbulent region. The ability to capture this effect is currently restricted to the PML solver, however both the HV and RVM models allows for the extraction of the SGS viscosity. A possible future application of the VPML library could be the use of the PML solver to extract an appropriate spatial description of the SGS viscosity. This could then be applied with a modified PSE method to model turbulence with the lower order GML solver for improved lower-order simulations.

TSR 9.96 A visualisation of the wake predicted by the PML method is given in Fig. 6.22. The wake expansion is much stronger than in the previous two cases. This is to be expected and is the consequence of the turbine entering a wake blockage state, where the rotor blockage is too high and energy extraction decreases. The second observation is that the wake appears to begin entraining the outer irrotational fluid at an earlier streamwise position $x \approx 3D$, as observed in Fig. 6.23. Equivalent observations can be made

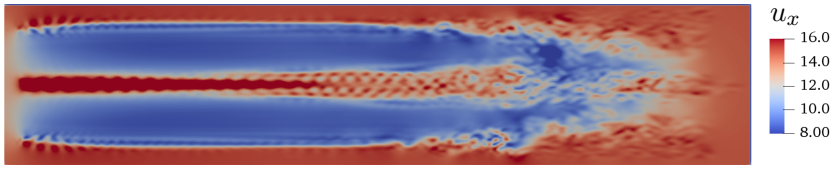

Figure 6.21: Velocity field of the rotor for TSR 6.65 of the PML solver.

about the turbulence modelling here as with the TSR = 6 case. In the case here it is desirable to simulated for a longer time period to ensure that the wake region is fully converged. It can be observed in Fig.6.22 how the wake

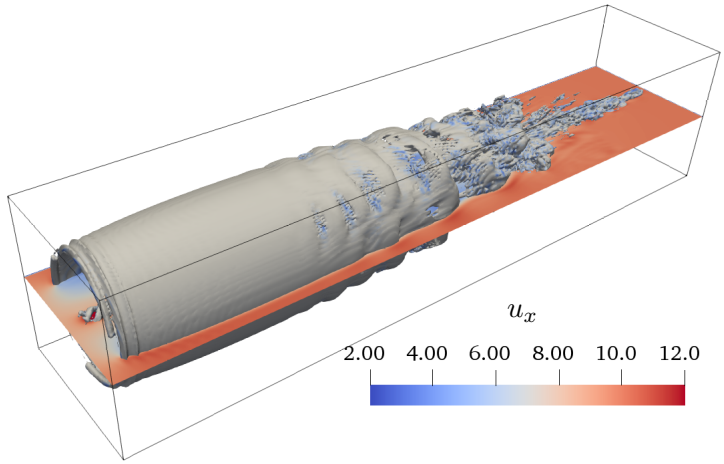

Figure 6.22: Velocity field of the rotor for TSR 6.65 of the PML solver. The contour shown is an isocontour of vorticity.

appears to behave almost as that of a bluff body. This has also been made experimentally [144]. In this case the individual helices merge together into a vorticity sheet due to their greater proximity to each other.

Velocity Predictions and Discussion

In order to inspect the performance of the models the calculated velocities have been inspected and compared to the experimental results and other results from the literature. As pointed out earlier, for this particle count

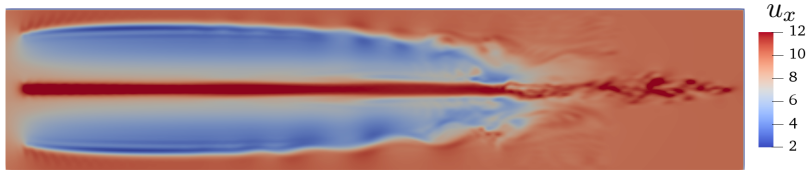

Figure 6.23: Velocity field of the rotor for TSR 6.65 and $H = 0.07\,\mathrm{m}$.

the analysis could be carried out with direct evaluation on the GPU. This was carried out in an earlier implementation of the Green's kernel (`GML`) method within the open-source software QBlade, for this reason the Green's results are marked as `GML` (QB) [70]. In addition to the results of the `VPML` library, the results of the LLFVW model within QBlade are demonstrated for comparison. Comparison is also made to the CFD results given in the reference document [136]. These were chosen to be the best and worst performing CFD solvers for each inspected field. The solvers will be described where necessary.

Radial velocity traverse The first case inspected is the radial velocity traverse. These measurements were carried out in the plane $x = const.$ located 0.3 m downstream of the turbine plane along the y axis and is illustrated with a dashed line in Fig. 6.17. This position is relatively close to the rotor, however the entire wake influences the induction through the rotor plane, as such the velocities here are indicative of the integral effect of the wake.

Comparison is made to the results of two CFD solvers given in the reference document. The first is the Wind-Multi Block finite-volume solver with a URANS turbulence model developed at the CFD laboratory of the University of Liverpool (Liverpool)[145, 136]. The second set of comparison CFD results were carried out in the Technion University in Haifa, Israel with the commercial software package STAR-CCM [136]. This also uses a finite-volume approach, however a steady simulation is carried out with a RANS k-ω shear stress transport (SST) turbulence model.

The axial velocity predictions are shown in Fig. 6.24. It is seen that for the case $U_\infty = 10.05\,\mathrm{ms}^{-1}$ (hereafter **high TSR** case) the `VPML` method slightly underpredicts the induced velocity as compared to the CFD methods. This disparity is seen to be stronger for the `PML` solver. It is suspected that this

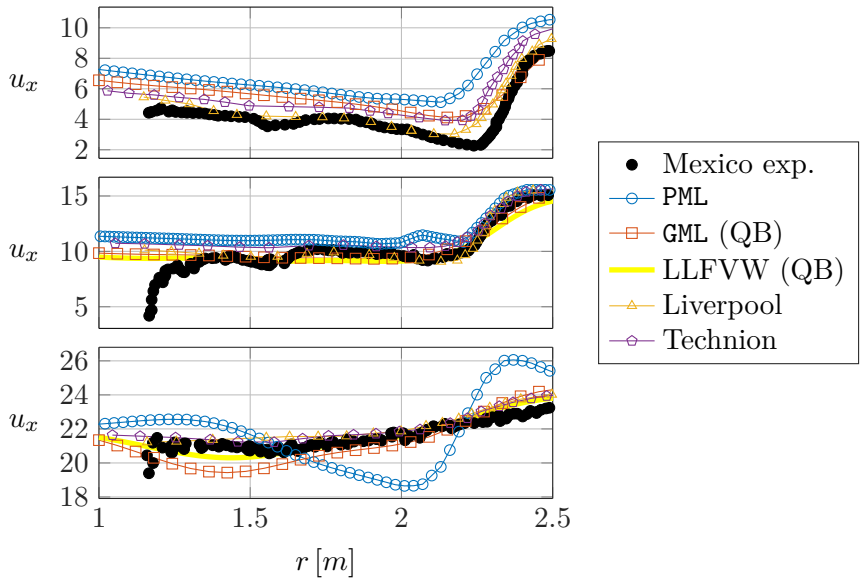

Figure 6.24: Comparison to experimental velocities collected along a radial traverse at $x = 0.3\,\mathrm{m}$. Velocity component in x-direction. From top to bottom: $U_\infty = 10.05$, 15.06 and $24.05\,\mathrm{ms}^{-1}$.

is a result of specifying the convolution core parameter σ too low, which leads to a artificially low induction. This shall be discussed in the following section. For the case $U_\infty = 15.06\,\mathrm{ms}^{-1}$ (hereafter **design TSR** case) this deviation is not observed as strongly. The agreement generally appears to be better for both solvers and agrees well with that of CFD methods. All solvers fail to predict the velocity deficit which appears at $r = 1\,\mathrm{m}$, this is likely caused by effects of the root vortex or nacelle. For the case $U_\infty = 24.05\,\mathrm{ms}^{-1}$ (hereafter **low TSR** case) a large deviation is seen to be observed from the `PML` solver, it is suspected that this is a result of the strong shed vortex at $r = 1.5\,\mathrm{m}$ which results from the steep change in gradient at this low TSR configuration, see Fig. 6.18. It can be argued that the circulation distribution predicted by the lifting line method results in an unrealistically discontinuous distribution which does not affect the `GML` due to the different core parameters used there. This lifting line methodology is based on a quasi-2D sectional blade treatment and this circulation jump may in reality be dampened out by 3D effects on the blade, this motivates a fully 3D treatment of the blade aerodynamics. In general however, it appears that the predicted trends agree well with the velocity distribution along the radial traverse.

The radial velocity distribution u_y along the radial traverse is shown in

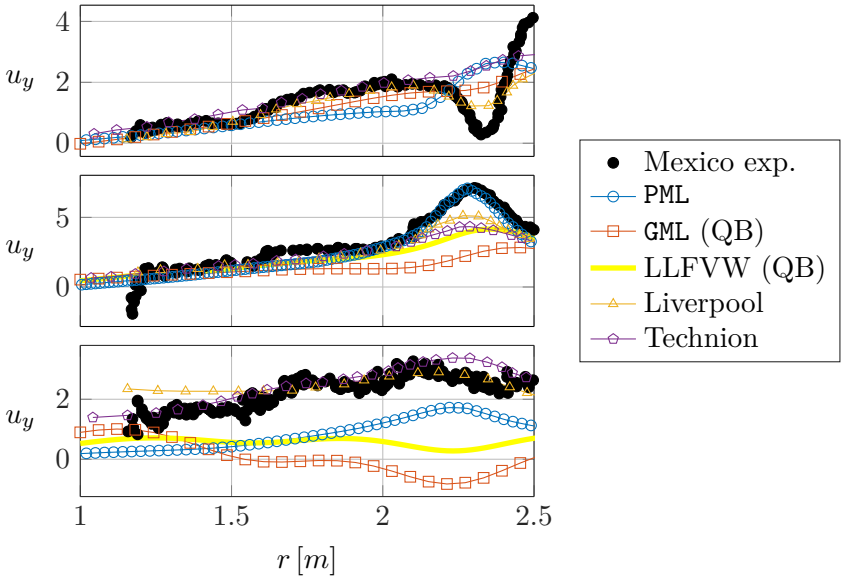

Figure 6.25: Comparison to experimental velocities collected along a radial traverse at $x = 0.3\,\mathrm{m}$. Velocity component in y-direction. From top to bottom: $U_\infty = 10.05$, 15.06 and $24.05\,\mathrm{ms}^{-1}$.

Fig. 6.25. For the high TSR case all solvers appears to fail to predict an experimental dip in the radial velocity which occurs due to the tip vortex, which it is seen has indeed drifted outwards due to wake expansion. The agreement however between all solvers appears to be quite good. For the design TSR case the `PML` solver appears to be the only solver to capture a peak in the radial velocity profile. All solvers also fail to predict this, including the `GML` method, where it is likely that not resolving the tip vortex leads to erroneous tip vortex wandering. Significant differences are already observed in the single airfoil case– see Fig. 6.4. Inspecting now the radial velocity for the low TSR case, there is quite a large disparity in the predictions. A possible explanation for this is the increased blockage caused by the larger separation over the blade and nacelle which, as outlined earlier, is completely neglected here. The steady solver from Technion appears to produce the best results here, likely due to the assumption of steadiness, which is likely a much better approximation for the low TSR case where induction is low.

Axial velocity traverse The velocity predictions are now compared to the experimental results collected along an axial traverse. In this case the second CFD solver compared is the Ansys Fluent solver applied by Snel

et al. [138], this also makes use of the SST k-ω model. For the results shown here the velocity profile has been plotted up to $x = 2D$. The axial velocity u_x is shown in Fig.6.26. Here it is observed for both the high and design TSR, the `GML` and `PML` methods both underpredict the induction velocity. This was observed earlier for the radial velocity measurements. It is believed that the main reason for this is an incorrect specification of the core size of the convolution (6.2). In general, decreasing σ increases the volume vorticity. As described in Section 6.3, this was chosen as per the standard methodology to be a function of the chord length of the blade. It is however possible that this parameter is better chosen based upon the grid parameters of the simulation, in this case grid size H.

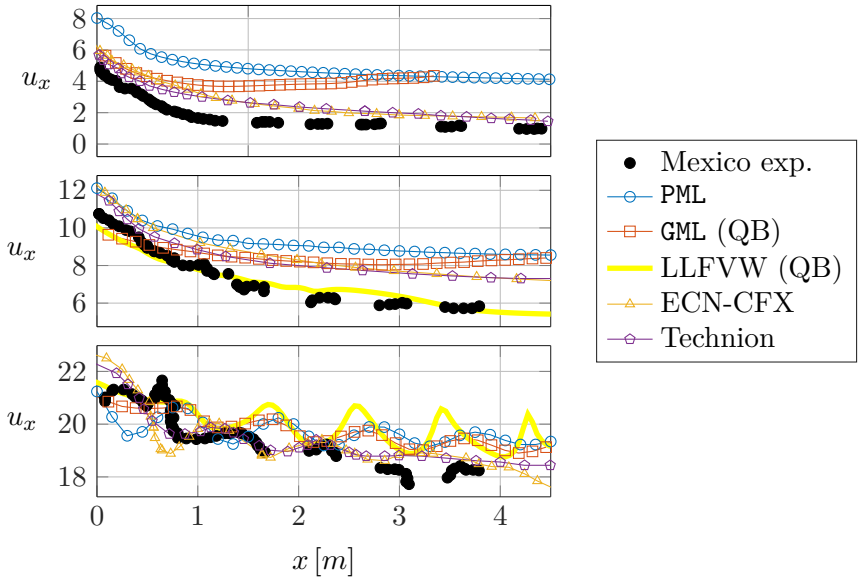

Figure 6.26: Comparison to experimental velocities collected along an axial traverse at $y = 1.8\,\mathrm{m}$. Velocity component in x-direction. From top to bottom: $U_\infty = 10.05$, 15.06 and $24.05\,\mathrm{ms}^{-1}$.

Although the complete mapping of the total shed circulation is essentially guaranteed by Eq. (6.2) as long as a sufficient mapping area is taken, for a simulation with a relatively high Re the tip and root vortex cores may translate over a large distance without significant viscous diffusion, implying stronger induced velocities. It is necessary to carry out a detailed parameter study of this in order to identify how best to choose this parameter based upon the blade model applied. In addition to this, it must be clearly stated that the blade treatment here is, at best, approximative. It takes very little blade geometry information into account and the Gaussian convolution

is an idealized mapping of the blade shed vorticity. The adoption of an actuator-line [9] or lifting-dragging line approach [146] would very likely produce much more realistic results.

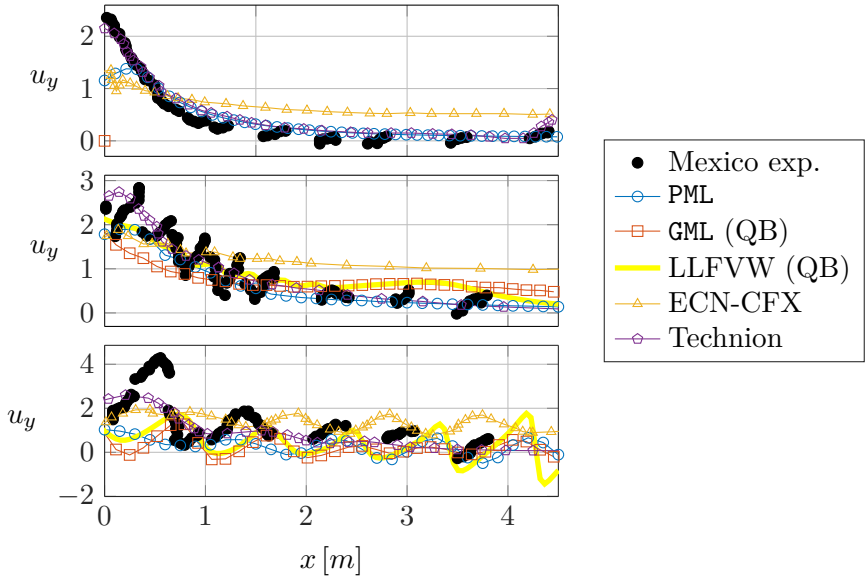

Figure 6.27: Comparison to experimental velocities collected along an axial traverse at $y = 1.8\,\mathrm{m}$. Velocity component in y-direction. From top to bottom: $U_\infty = 10.05$, 15.06 and $24.05\,\mathrm{ms}^{-1}$.

For the low TSR case, where this effect will be less noticeable due to the lower induction, the results align much better and all solvers appear to predict the oscillating velocity profile due to the proximity of the tip vortex to the velocity probe position. The distribution of the radial velocity is shown in Fig. 6.27. Here predictions are seen to be much better. The unsteadiness for the low TSR case is again observed, and the drop off for the high TSR case is well captured by the `PML` method. Unfortunately due to human error the results for the `GML` for this case was not recorded. The results for the optimum TSR also show relatively satisfactory results, however the unsteadiness which is this case is again visisble is not well captured by the solvers. Despite this, the mean magnitude appears to agree relatively well.

For all cases it would be helpful to analyse the velocity prediction into the far wake, e.g. $x > 6D$. The results here appear to demonstrate the need for a dedicated parameter study to identifying optimum core and convolution parameters for the `GML` and `PML` solvers. To inspect the velocities in these regions would very likely lead back to errors which propagate from

the aforementioned specifications. The CFD solvers against which these comparisons have been carried out are almost a decade old. The state of the art with regards to unsteady turbulence modelling and in particular, LES modelling, has already significantly advanced in this time. For the desired parameter studied mentioned above it would hence be desirable to carry out comparison to a state-of the art CFD solver with a validated LES turbulence model.

In summary, considering the relatively approximative nature of the treatment here, the `PML` and `GML` solvers appear to perform well. Further research is required to validate both the solver and turbulence model thoroughly against a comparable state of the art finite-volume approach with a similar LES turbulence model.

6.4 Case 4: Periodic Helical Wake

An application of the solvers to wind turbine aerodynamics is the problem of exciting the wake system of a horizontal axis wind turbine. As described in Section 1.2, the ability to excite the wake helix in such a way that breakdown occurs more rapidly has the potential to greatly increase the energy yield of turbines in a wind farm environment [147]. This requires an investigation of the fundamental modes of instability of a helical wake system.

For the purposes of investigating the instabilities present in a helical wake system, the physics of the problem can be greatly simplified by assuming periodicity along the axis of symmetry. This removes the influence of spatial development, and treats directly the converged helical wake.

In the case of a single helical vortex, there exists an analytical treatment of the linear stability problem which allows identification of the optimum modes of excitation of the wake helix. These modes have been applied to specify an initial perturbation in the cases here and the system is observed in order to track the development of the instability into the non-linear and saturation stages. The `GML` solver is more suitable to problems of linear stability where the filament treatment is sufficient to capture initial instability growth. For the case here it is desirable to inspect the effects of turbulent diffusion on the evolution and henc ehte `PML` solver has been applied. The James-Lackner algorithm has been used to solve the near field influence.

These cases demonstrate the potential of the solver to investigate fundamental modes of instability of a wind turbine wake. Although the cases here only contain a single helical filament, the extension of the problem to

multiple helices within the solver is trivial, this can be considered a task for future investigations. Furthermore, as the behaviour into the non-linear regime is complicated to analyse analytically, the results presented in this section are predominantly qualitative.

Widnall Solution

In her seminal paper, Widnall [74] investigated the modes of instability of a single helical vortex by treating the helix as a thin-cored vortex filament. The description of the helix differs from Eq. (3.16) in order to allow the introduction of a perturbation in the helix filament position. The filament is parametrised with the variable z, which represents a normalized azimuthal angle. The helix has radius R and pitch $h = 1/k$. The filament is perturbed sinusoidally with amplitude η in radial direction and ξ in axial direction with wave number γ. The two displacements are fluid-dynamically coupled, implying that for the linear analysis the perturbation only needs to be made in one direction. Following Walther et al. [75], these perturbations are introduced in the radial direction, as such $\xi = 0$. Through transformation of an appropriate coordinate system, the position of the vortex filament can be well approximated with a Taylor series as:

$$\vec{x} = \begin{bmatrix} R\cos k'z \\ R\sin k'z \\ z\zeta^2 \end{bmatrix} + \eta \exp\{i\gamma z\} \begin{bmatrix} \cos k'z \\ \sin k'z \\ 0 \end{bmatrix}, \tag{6.3}$$

where $\zeta^2 = (1 + k^2R^2)^{-1}$ is a scaling factor along the axis of symmetry and $k' = k\zeta^2$. By applying the Biot-Savart equation for the induced velocity due to a vortex filament, a linear system is derived which allows for an eigenvalue analysis. The core size of the filament was seen to strongly influence the stability of the helix filament. A stability plot for various core sizes and perturbation wave numbers is given in Fig. 6.28.

Modes of instability Widnall was able to identify three dominant modes of instability. The first, referred to as the long-wave instability, occurs when $\gamma/k' < kR$. This physically can be considered to be a displacement of a full helix winding. The second, referred to as the mutual-inductance instability occurs when successive turns pass within a distance of one radius. In this case neighboring vortex filaments attract each other, destroying the symmetry of the wake and eventually merging. The mutual inductance instability occurs for odd fraction multiples of γ/k', e.g. $3/2, 3/2, \ldots$. The third case, referred to as the short-wave instability, is analogous to the short-wave instability seen along a vortex ring, as simulated in Section 5.1.6.

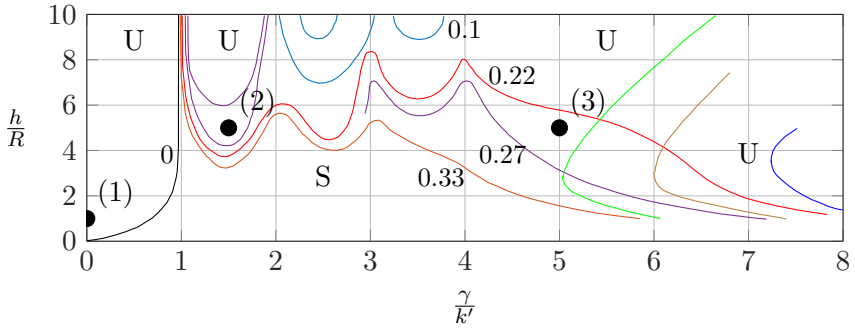

Figure 6.28: Stability boundaries based upon the treatment of Widnall. The parameter γ/k' represents the wave number ratio of the perturbation to the undisturbed helix. S and U indicate stable and unstable regions, respectively. Case 1,2 and 3 simulated here are marked on the graph in parenthesis. The values indicate the relative core size of the vortex filament, σ/R. Figure has been modified from [75].

Table 6.2: Simulation parameters for helical stability tests.

Case	Re	R [m]	L [m]	σ/R	γ/k'	NB_{per}	N_p
1	5000	0.48	2.4	0.125	0.0	48	3.68e5
2	2500	1.92	2.4	0.125	1.50	16	6.00e5
3	2500	1.92	2.4	0.125	5.0	16	6.05e5

Simulated cases

Three cases have been simulated here. Helix geometric and perturbation parameters are summarised in Table 6.2. Case 1 is the undisturbed helix baseline case. In case 2 a mutual inductance instability is be simulated. In case 3 a short-wavelength instability is simulated. The parameters here have been taken from an equivalent VPM simulation carried out by Walther et al. [75]. The helix core is parametrised with Eq. (6.3) and the vorticity of the core section is Gaussian, as given by Eq. (5.8) with core size σ. For all cases the vortex has circulation $\Gamma = 5\text{e}{-3}\,\text{m}^2\text{s}^{-1}$ and the perturbation magnitude is set such that $\eta = 0.02\,\sigma$. This introduces a very weak initial perturbation into the vorticity field at the desired wave number. This encourages development of the growth of the stability. As the linear stability analysis results are generally well known, the simulations here shall demonstrate the capture of non-linear effects by the solver. The normalised radius of the perturbed helix as a function of azimuth is given for each case in Fig. 6.29. For each case the helix has four full windings. This is

necessary to ensure that the full length of the perturbed helix segment is contained within a single periodic domain for all cases considered. It can be seen that the deviation from the unperturbed radius R_0 are very small. This simplifies the initial generation of the vorticity field and furthermore ensures that the perturbation is not strongly forced.

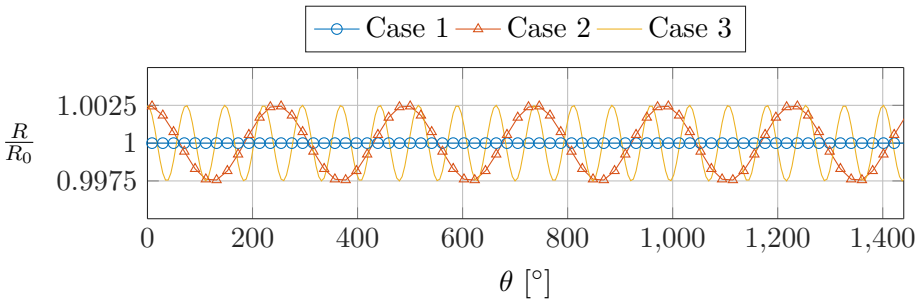

Figure 6.29: Perturbation of the initial filament radial position for the test cases simulated.

Simulation parameters For all simulations the grid resolution is set to $H = 0.05\,\mathrm{m}$. The resolution chosen for the simulations here is significantly lower than those in [75], as the simulations carried out here are for illustrative purposes rather than thorough investigation. As described in Section 4.1.7, the length of the periodic domain L_{per} is specified with the number of boxes which span this length NB_{per}. This also given in Table 6.2. The simulation time step has been set to $dT = T_0 \cdot 10^{-3}$ where $T_0 = \Gamma/4\pi R^2$ is the characteristic time of the problem.

Instability Growth

For the three cases displayed, the times shown are normalised time $\tau = t/T_0$ are shown. For cases demonstrating instability growth, the simulations are carried out until the instability reaches a saturation stage.

Case 1: Unperturbed helix For the case of the unperturbed helix, the choice of $NB_{per} = 48$ was made to ensure that the periodic solver is functioning by choosing a relatively long spatial domain. A visualisation of this is given in Fig. 6.30. It is expected that the single, undisturbed helix translate and rotates with constant rate along the axis of symmetry in a corkscrew fashion. This is indeed seen to occur here. The core is also seen to spread owing to the effects of viscous diffusion.

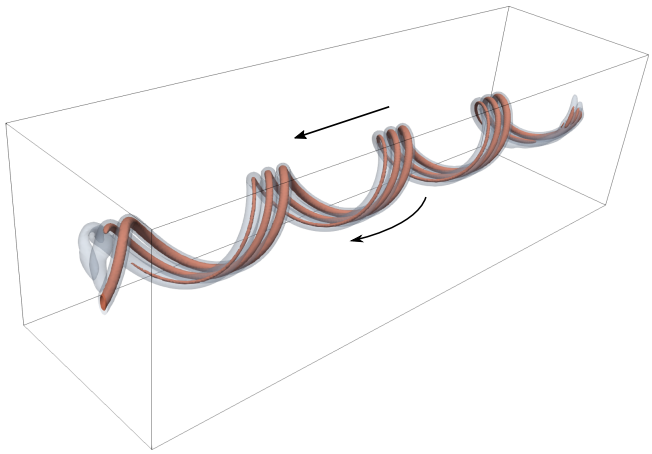

Figure 6.30: Undisturbed single helix shown at : τ = 0,0.5, 1. Direction of translation is shown, along with direction of rotation. Isocontours of vorticity shown $\omega = 0.2$ (opaque) and $0.1\,\mathrm{s}^{-1}$. It can be seen how the region of higher vorticity decreases due to viscous diffusion.

It was observed in implementation that care must be taken to ensure that particle flux over the periodic domain boundaries is carried out consistently. This can otherwise create problems in virtually all steps of solution, including filtering and remeshing procedures. The use of the `PML` method in this case together with the description in Section 4.1.7 led to relatively straightforward implementation of the periodic cases described here.

Case 2: Mutual inductance instability A visualisation of the mutual inductance throughout the evolution of the simulation is shown in Fig. 6.31. The relative pitch h/R in this case is much higher than in the undisturbed case as the more densely packed helices induce a more rapidly growing instability. The beginning of the growth stage is seen at $\tau = 0.375$, where neighboring windings are seen to be relatively displaced to each other. This effect amplifies rapdily by $\tau = 0.625$, at which point the outer contours of the helix begin to become more chaotic. The saturation stage is almost reached by $\tau = 1.125$ where turbulent filamentation is seen to occur. A visual comparison is made to the simulations of Walther in Fig. 6.32. Numerous simularities are observed here. The larger outer filaments which are diverging away can be seen. Furthermore, the inner region where fluid is being drawn through these larger rings is also captured. The symmetry observed in Walther is not observed here, but it is important to note that the simulations here include the `RVM` turbulence model, whereas the simulation is Walther

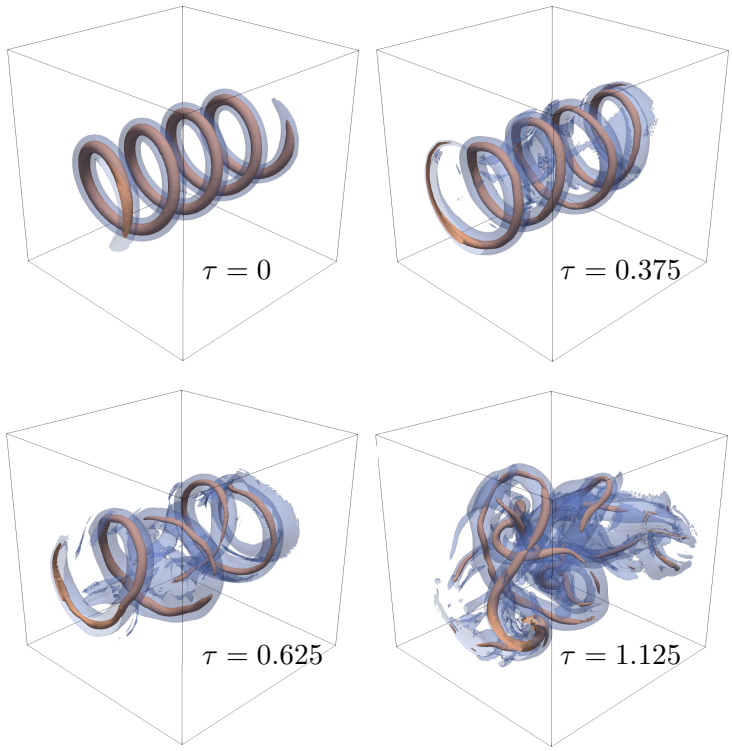

Figure 6.31: Mutual inductance instability development.

are purely DNS. In the simulation carried out here the helical filament entered the saturation stage earlier than the Walther case. The inclusion of turbulent diffusion is seen to influence both symmetry and also global structure of the non-linear evolution.

Case 3: Short wave instability In this case the helix has been excited with a much higher wave number, such that the instability which arises is not a result of interaction with neighboring helices, but rather due to the inherent short-wavelength Widnall instability which arises due to the embedding of a straight filament in a sheared flow. The evolution of this instability is shown in Fig. 6.33. Here the linear growth stage is observed at $\tau = 0.125$. As with the ring case simulated in Section 5.1.6, these regions act to draw surrounding fluid in asymmetrically, which for a helical flow creates the *mushroom* flow visible at $\tau = 0.375$. Advanced filamentation is already seen to occur at $\tau = 0.625$ and the saturation is reached much more quickly than for the mutual inductance case. The flow appears to be almost

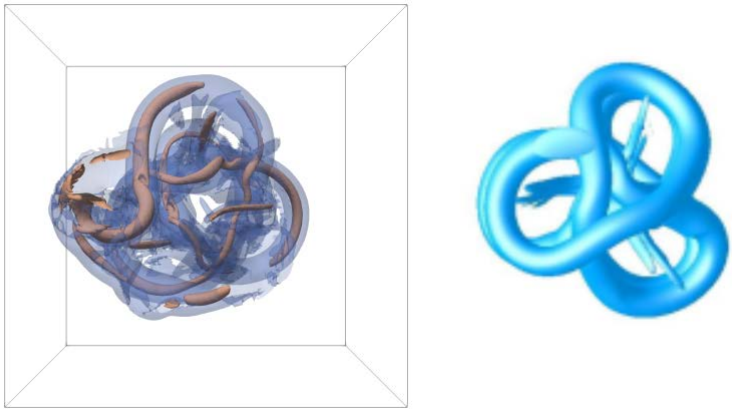

Figure 6.32: Comparison of the mutual inductance instability of the PML solver (left) with the results shown in Walther et al. [75] at $\tau \approx 1$.

fully turbulent by $\tau = 1$ in this case. These flow features are also observed in Walther. The effects of turbulent diffusion however appear much stronger here as the solution appears to have greatly broken down already by $\tau = 1.0$, whereas in Walther the global strucutre is still somewhat present at $\tau = 1.2$.

Discussion The ability of the flow solver to simulate these cases is an illustration of the potential application of the VPML library. The ability to excite a helical wake is a topic of great interest currently as this could have a significant influence on the turbine placement within a wind farm. Numerous methods have been recommended for the excitation of these instabilities, particularly of long-wave instabilities and mutual-inductance instabilities, however thus far little research has been carried out into practical excitation of the short-wave instability. The results above demonstrate that this excitation can possible lead to quicker breakdown of the wake. It is hence of interest to further investigate such flows in order to have a complete physical picture of the problem, if successful excitation measures are to be developed. For successful development, multiple design iterations are required, necessitating quick computational turnaround. The solution above were achieved on a desktop computer in less than 12 hours. With further optimisation, it is the opinion of the author that even further reduction in calculation time can be achieved.

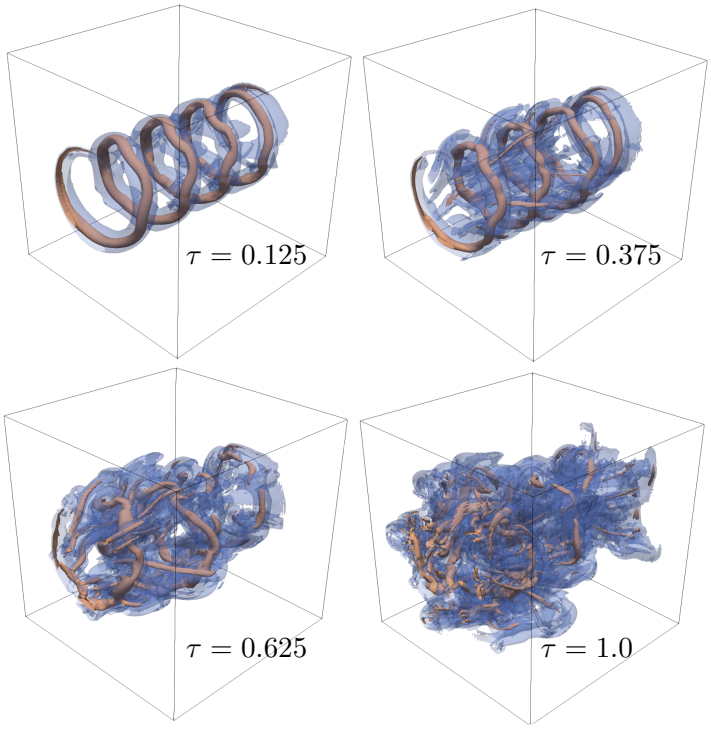

Figure 6.33: Development of the short-wavelength instability.

Chapter 7 Conclusions and Outlook

7.1 Conclusions

The ability to simulate the flow in the wake of lift-generating bodies has a range of important and interesting applications. One major application is to the problem of wind turbine wakes in order to be able to better predict the flow within a wind farm environment to allow for better turbine placement and operation. If detailed information about the flow field in the wake is to be known, then higher-order effects such as viscous and turbulent diffusion must be accounted for. A review of the current state of the art methods reveals a gap in the required technology between lower-order simulation methods and higher-order methods. The latter are generally far too computationally expensive to be applicable in a design environment.

This work set out to address this gap by applying a general model which is applicable to both medium- and high-order simulation methods. It was a key design constraint that the developed method should not require access to high-power computing (HPC) resources. The reason for this is that it should be applicable to a design environment, where multiple, rapid design iterations must be carried out. A review of the literature showed that two simulation frameworks can be adopted: Eulerian methods, where a grid must be generated, and Lagrangian methods, which avoid the use of a user-generated grid. At an early stage it was concluded that Lagrangian methods offer the ability to account for higher order physics and can be applied to both medium and higher-order treatments by applying the vortex particle method (VPM). The computational expense of directly evaluating the vortex particle problem, which has the unfavorable computational complexity of $\mathcal{O}(N^2)$ for an N-particle problem, presents a barrier to the use of these methods in higher-order simulations.

This barrier was overcome by applying the multilevel multi-integration cluster (MLMIC) method. This method uses spatial coarsening and polynomial interpolation of both the source and influence distribution in order to reduce the computational expense to the optimal complexity $\mathcal{O}(N)$. The method furthermore allows for the use of interaction templates, which allows for significant optimisation of the treatment of the far field influence. In Chapter 3 this method and the steps necessary to integrate a source field were described. For application to VPM, two kernels are of interest: the Biot-Savart kernel for velocity evaluation and the stream function kernel.

It was observed that these kernels both behave as a singular kernel for large evaluation distances, making them amenable to optimisation with the MLMIC method. In that chapter it was demonstrated that, provided the box size is adequately chosen, the error ϵ of interpolation can be completely controlled by specifying of the order of the polynomial interpolating function, P.

The MLMIC method was implemented within a general vortex particle multilevel (`VPML`) library. Two solver types were implemented within the `VPML` library. The first was the `GML` solver, which makes use of the Green's function of the induced velocity to calculate the evolution of the flow field. The second was the `PML` solver, which makes use of the fast Poisson solver to resolve the stream function and from this extract all fields of interest to calculate evolution of the flow field. The implementation was described in Chapter 4.

A validation of the solvers against a range of flow cases was carried out in Chapter 5. The vortex ring was chosen for these tests as this is a well-understood vortex-driven flow for which a range of analytical and numerical solutions exist. The level of complexity was gradually increased for each test case. The ability of the `GML` solver to capture viscous and stretching effects was demonstrated. It is also shown that, unlike the `GML` method, the `PML` method is able to capture turbulent diffusion of the flow. This is because the higher-order gradients necessary for the calculation of the sub-grid viscosity can be calculated easily within the `PML` solver. The final validation cases show the ability of the `PML` solver to accurately model the turbulent breakdown of an azimuthally excited vortex ring. The performance of the solvers was investigated and the optimum complexity $\mathcal{O}(N)$ was attained in all cases. The overhead of the solvers was investigated and criteria for the choice of solver depending on desired application were described.

In order to demonstrate the range of applicability of the method to the practical calculation of wake flows, four distinct cases were simulated in Chapter 6. The unsteady developing wake of a single elliptical wing was simulated demonstrating numerous well-known physical phenomena such as the starting vortex and wing tip roll-up. Following this, a four-vortex wake system behind an aircraft was simulated. Here it was demonstrated that the solver is capable of simulating periodic systems and is able to inherently capture the expected modes of instability of the wake. Following this, the application cases to wind energy were simulated. The wake of an experimental turbine was simulated. The transition of the wake appeared to be well captured. This correct velocity distribution was shown to rely heavily on grid size and convolution parameters, a topic which requires

further investigation. The final investigated case was the modes of stability of a helical vortex filament, where the filament was excited at known unstable frequencies and the linear, non-linear and saturation stages of the instability growth were qualitatively demonstrated. This demonstrates the ability to carry out stability investigations with the `VPML` library.

All of the simulations in this work were carried out on a desktop PC with a somewhat performant GPU. The `VPML` library was implemented such that GPU resources can be exploited for problems which are amenable to GPU calculation. This was seen to greatly increase performance. The results demonstrate that both medium- and high-fidelity simulations can be carried out without requiring HPC. It can hence be concluded that the objectives have been reached and the `VPML` will hopefully have a place amongst practical wake simulation tools in the future.

7.2 Outlook

The optimal scaling achieved implies that high-fidelity simulations can be carried out with greater speed than were previously attainable. This allows for the investigation of a range of phenomena for which there exist knowledge gaps in the literature. Furthermore, there are further optimisations of the solution method possible which expand the domain of applicability of the solver.

Best practices for particle method with wind turbine simulations It was demonstrated in Chapter 6 that the choice of grid size and convolution parameter greatly influence the velocity field and turbulence models of the `PML` method. For improved modelling of wind turbine wakes, a parameter study which outlines the best practices for this should be carried out. An improved blade treatment will also undoubtedly influence results. One example of a suitable improvement would be the implementation of the lifting and dragging line formulation of Caprace et al. [146].

`PML` and `GML` solver synergy The inability of the `GML` method to resolve turbulence is a significant drawback if accurate simulation of wake breakdown is desired with lower-order models. The `PML` method in comparison most likely has an overly high resolution for applications within the design environment. A suitable compromise would be to use the `PML` method to identify turbulent statistics, which are then used within the PSE routine of the `GML` solver to account for the effects of viscous diffusion. This requires

an adaptation of the particle strengths to replicate the effects of the higher order gradients present, however this would provide an opportunity for the somewhat more realistic treatment of turbulent effects in medium-order models.

Wind turbine design and multiple turbine interactions The unsteady loading of a wind turbine operating in the wake shadow of another turbine is a crucial factor in turbine placement within a wind farm and a topic of great interest currently. Ideally, the library developed in the work here will enable the inexpensive simulation of such events and allow for optimised designs of turbines operating in such environments.

Multi-resolution treatment The large range of scales present in a wind turbine flow suggests that an approach which allows for multiple grid fidelities would be advantageous. Formally, provided the box treatment is adhered to, the `VPML` library could be configured such that it is possible to treat multi-scale problems, particularly within the `PML` solver. A suitable metric must be adopted which specifies a threshold grid resolution for each flow region. This is a topic in which great progress has been made already within vortex particle methods, by e.g. making use of wavelet-adapted grids [148]. It is desired in the near future to implement such a capability into the `VPML` library.

Open-source repository It is planned to make the MLMIC method open-source with a version of the `GML` library to allow for use within both the wind turbine and aircraft research and design communities. It has been a dominant factor in the implementation of the `VPML` library that the solution procedure reflects the intuitively simple concepts behind the MLMIC method. It is hopes that this implementation style allows future engineers to apply this framework to other problems which are equally amenable to the application of the method. Two such examples are astrophysical N-body problems and problems in electrostatics.

Bibliography

[1] H. Lamb. *Hydrodynamics*. Dover Books on Physics, 1945. ISBN 9780521458689. doi:10.1038/114638a0.

[2] J. Ferziger and A. Milovan. *Computational Methods for Fluid Dynamics*. Aeronautical and Aerospace Engineering. Springer, Berlin, 2002. doi:10.1007/978-3-642-56026-20.

[3] Kitokinimi. Jets and cloud effects. Accessed: 2021-05-29, 2013. URL: https://kitskinny.wordpress.com/2013/07/09/jets-clouds-effects/.

[4] Vattenfall. Vattenfall builds denmark's largest offshore windfarm. Accessed: 2021-05-29, 2016. URL: https://news.cision.com/vattenfall/r/vattenfall-builds-denmark-s-largest-offshore-windfarm,c2021812.

[5] R. Aris. *Vectors, Tensors and the basic equations of fluid mechanics*. Dover books on mathematics, 1990. doi:10.1017/S0001924000062679.

[6] C. Hirsch. *Numerical Computation of Internal and External Flows: The Fundamentals of Computational Fluid Dynamics*. Butterworth Heinemann, 2 edition, 2007. doi:10.1016/B978-0-7506-6594-0.X5037-1.

[7] B. Cockburn, E. Karniadakis, and Shu C.W. *Discontinuous Galerkin Methods: Theory, Computation and Applications*. Springer, 2000. doi:10.1007/978-3-642-59721-3.

[8] D. Gottlieb and S.A. Orszag. *Numerical Analysis of Spectral Methods: Theory and Applications*. Society for Industrial and Applied Mathematics, 1987. doi:10.1137/1.9781611970425.

[9] J.N. Sørensen and W.Z. Shen. Numerical modelling of wind turbine wakes. *J. Fluids Eng*, (2):393–399, 2002. doi:10.1115/1.1471361.

[10] L.J. Vermeer, J.N. Sørensen, and A. Crespo. Wind turbine wake aerodynamics. *Progress in Aerospace Science*, 2003. doi:10.1016/S0376-0421(03)00078-2.

[11] G. Papadakis. *Development of a hybrid compressible vortex particle method and application to external problems including helicopter flows*. PhD thesis, National Technical University of Athens, 2014. doi:10.13140/RG.2.2.26480.92168.

[12] P. Sagaut and S. Deck. Large eddy simulation for aerodynamics: status and perspectives. *Phil. Trans. R. Soc. A*, 367:2849–2860, 2009. doi:10.1098/rsta.2008.0269.

[13] P. Sagaut, S. Deck, and M. Terracol. *Multiscale and Multiresolution Approaches in Turbulence*. Imperial College Press, 2006. doi:10.1142/p878.

[14] Rodriguez. I., R. Borell, O. Lehmkuhl, C. Perez Segarra, and A. OLiva. Direct numerical simulation of the flow over a sphere at re = 3700. *J. Fluid. Mech.*, 2011. doi:10.1017/jfm.2011.136.

[15] S.B. Pope. *Turbulent Flows*. Cambridge University Press, 2000. doi:10.1017/CBO9780511840531.

[16] W. Rodi, J.J Ferziger, M. Breuer, and M. Pourquie. Status of large-eddy simulation: results of a workshop. *Trans.- Am. S. Mech. Eng.*, 1997. doi:10.1115/1.2819128.

[17] A. Leonard. Energy cascade in large-eddy simulations of turbulent fluid flows. *Adv. Geophys.*, (18), 1974. doi:10.1016/S0065-2687(08)60464-1.

[18] J. Smagorinski. General circulation experiments with the primitive equations. *Monthly Weather Review*, (91):99–164, 1963. doi:10.1175/1520-0493(1963)091<0099:GCEWTP>2.3.CO;2.

[19] C. Mockett. *A comprehensive study of detached-eddy simulation*. PhD thesis, Technical University of Berlin, 2009. doi:10.14279/depositonce-2305.

[20] S.S. Girimaji. Partially-averaged navier-stokes model for turbulence: A reynolds-averaged navier-stokes to direct numerical simulation bridging method. *J. Appl. Mech.*, 2006. doi:10.1007/978-3-642-31818-4_3.

[21] E. Branlard. *Wind Turbine Aerodynamics and Vorticity Based Methods*. Springer International Publishing, 2017. doi:10.1007/978-3-319-55164-7.

[22] G.S. Winckelmans. *Topics in Vortex methods for the Computation of two- and three-dimensional incompressible unsteady flows*. PhD thesis, California Institute of Technology, 1989. doi:10.7907/19HD-DF80.

[23] G.S. Winckelmans and A. Leonard. Contributions to vortex particle methods for the computation of three-dimensional incompressible unsteady flows. *J. Comp. Phys*, 109:247–73, 1993. doi:10.1006/jcph.1993.1216.

[24] R. Mittal and G. Iaccarino. Immersed boundary methods. *Annu. Rev. Fl. Mech.*, 37:239–261, 2005. doi:10.1146/annurev.fluid.37.061903.175743.

[25] G. Daeninck. *Developments in Hybrid Approaches: Vortex method with known separation location. Vortex method with near-wall Eulerian solver. RANS-LES coupling.* PhD thesis, Universite Catholique de Louvain, 2006. URL: https://dial.uclouvain.be/pr/boreal/object/boreal:22806.

[26] Y. Marichal. *An Immersed Interface Vortex Particle-Mesh Method.* PhD thesis, Universite catholique de Louvain, 2014. doi:10.1016/j.jcp.2019.05.033.

[27] S. Degond and P. Mas-Gallic. The weighted particle method for convection-diffusion equations. i) the case of an isotropic viscosity. *Math. Comp.*, 53(188):485–507, 1989. URL: https://ui.adsabs.harvard.edu/abs/1989MaCom..53..485D/abstract.

[28] R. Cocle, G. Winckelmans, and G. Daeninck. Combining the vortex-in-cell and parallel fast multipole methods for efficient domain decomposition simulations. *J. Comp. Phy.*, (227):9091–120, 2008. URL: http://hdl.handle.net/2078.1/5211.

[29] G.S. Winckelmans. Some progress in large-eddy simulation using the 3-d vortex particle method. Technical report, Center for Turbulence Research, 1995.

[30] Winckelmans G. Jeanmart, H. Investigation of eddy-viscosity models modified using discrete filters: A simplified "regularized variational multiscale model" and an "enhanced field model". *Phys. Fl.*, 5(19), 2007. 055110. doi:10.1063/1.2728935.

[31] P. Chatelain, S. Backaert, G. Winckelmans, and S. Kern. Large eddy simulation of wind turbine wakes. *Flow Turb. Combust.*, 2013. doi:10.1007/s10494-013-9474-8.

[32] P. Chatelain, M. Duponcheel, D.G. Caprace, Y. Marichal, and G. Winckelmans. Vortex particle-mesh simulations of vertical axis wind turbine flows: from the blade aerodynamics to the very far

wake. *Journal of Physics: Conference Series*, (753), 2016. doi:10.5194/wes-2-317-2017.

[33] S.J. Lind, B.D. Rogers, and P.K. Stansby. Review of smoothed particle hydrodynamics: towards converged lagrangain flow modelling. *proc. R. Soc. A*, 2020. doi:10.1098/rspa.2019.0801.

[34] J.J. Monaghan. Smooth particle hydrodynamics. *Rep. Prog. Phys.*, 2005. doi:10.1146/annurev.aa.30.090192.002551.

[35] D. Violeau. *Fluid Mechanics and the SPH Method: Theory and Applications*. Oxford University Press, 2015. doi:10.1080/00221686.2015.1119209.

[36] A. Di Mascio, M. Antuono, A. Colagrossi, and S. Marrone. Smoothed particle hydrodynamics method from a large eddy simulation perspective. *Phys. Fluids*, (29), 2017. doi:10.1063/1.4978274.

[37] P.D. Koumoutsakos. *Direct Numerical Simulations of Unsteady Separater Flows Using Vortex Methods*. PhD thesis, California Institute of Technology, 1993. doi:10.7907/TCQ9-9C86.

[38] P. Koumoutsakos, A. Leonard, and F. Pepin. Boundary conditions for viscous vortex methods. *J. Comp. Phys.*, 113:52–61, 1994. doi:10.1006/jcph.1994.1117.

[39] P. Koumoutsakos. Inviscid axisymmetrization of an elliptical vortex. *J. Comp. Phys.*, (128):821–857, 1997. doi:10.1006/jcph.1997.5749.

[40] P. Ploumhans and G.S. Winckelmans. Vortex methods for high-resolution simulations of viscous flow past bluff bodies of general geometry. *J. Comp. Phys.*, 165:354–406, 2000. doi:10.1006/jcph.2000.6614.

[41] N.J. Zabusky, M-H Hughes, and K.V. Roberts. Contour dynamics for the euler equations in two dimensions. *J. Comp. Phys.*, 52:351–373, 1979. doi:10.1016/0021-9991(79)90089-5.

[42] D.G. Dritschel. Contour dynamics and contour surgery. *Computer Physics Reports.*, (102):77–146, 7 1989. doi:10.1016/0167-7977(89)90004-X.

[43] D.I. Pullin. Contour dynamics methods. *Annu. Rev. Fluid Mech*, 24:89–115, 1992. doi:10.1146/annurev.fl.24.010192.000513.

[44] D.B. Bliss. The dynamics of curved rotational vortex lines. Master's thesis, Massachusetts Institute of Technology, 1970. URL: http://hdl.handle.net/1721.1/13548.

[45] A. van Garrel. Integral boundary layer methods for wind turbine aerodynamics. Technical report, ECN, 12 2003. Technical Report ECN-C–04-004. doi:10.13140/RG.2.1.3035.9449.

[46] Leishman J.G., M.J. Bhagwat, and A. Bagai. Free vortex filament methods for the analysis of helicopter rotor wakes. *J. Aircraft*, 39:759–775, 2002. doi:10.2514/2.3022.

[47] E.J. Alvarez and A. Ning. Development of a vortex particle code for the modeling of wake interaction in distributed propulsion. In *AIAA Applied Aerodynamics Conference*, 6 2018. doi:10.2514/6.2018-3646.

[48] J. Saverin, D. Marten, G. Pechlivanoglou, and C.O Paschereit. Advanced medium-order modelling of a wind turbine wake with a vortex particle method integrated within a multilevel code. *J. Phys: Conf. Ser.*, (1037), 2018. doi:10.1088/1742-6596/1037/6/062029.

[49] J. Barnes and P. Hut. A hierarchical o(n log n) force-calculation algorithm. *Nature*, pages 446–449, 1986. doi:10.1038/324446a0.

[50] L. Greengard, J. Carrier, and V. Rokhlin. A fast adaptive multipole algorithm for particle simulations. *SIAM J. Sci. and Stat. Comput.*, (9):669–86, 1988. doi:10.1137/0909044.

[51] L. Greengard and V. Rokhlin. A new version of the fast multipole method for the laplace equation in three dimensions. *Acta Numerica*, (6):229–269, 1997. doi:10.1017/S0962492900002725.

[52] T. Berdowski. 3d lagrangian vpm-fmm for modelling the near wake of a hawt. Master's thesis, TU Delft, 2015. URL: http://resolver.tudelft.nl/uuid:99174fca-9568-4ed4-9876-b7771b6560c9.

[53] A. Brandt. Guide to multigrid development. In *Multigrid Methods*, page 220, 1982. doi:10.1007/BFb0069930.

[54] A. Brandt and A.A. Lubrecht. Multilevel matrix multiplications and fast solution of integral equations. *Journal of Computational Physics*, (90):348–370, 1990. doi:10.1016/0021-9991(90)90171-V.

[55] J.C. Adams. Mudpack: Multigrid portable fortran software for the efficient solution of linear elliptic partial differential equations. *Appl.*

Math. & Comp., 34(2):113–146, 1989. doi:10.1016/0096-3003(89)90010-6.

[56] Computational and Information Systems Lab. Fishpack documentation, 1999. URL: https://www2.cisl.ucar.edu/resources/legacy/fishpack/documentation.

[57] V. Fuka. Poisfft – a free parallel fast poisson solver. *Applied Mathematics and Computation*, 267:356–364, Sep 2015. URL: http://dx.doi.org/10.1016/j.amc.2015.03.011, doi:10.1016/j.amc.2015.03.011.

[58] A. van Garrel, C.H. Venner, and H.W. Hoeijmakers. Fast multilevel panel method for wind turbine rotor flow simulations. In *Proc. AIAA SciTech Forum*, Grapevine, Texas, USA, 1 2017. doi:10.2514/6.2017-2001.

[59] C.H. Venner and A.A. Lubrecht. *Multilevel Methods in Lubrication*, volume 37. Elsevier Science, 1993.

[60] M.O.L. Hansen. *Aerodyanmics of Wind Turbines*. Earthscan Publications Ltd., 2 edition, 2008. doi:10.4324/9781315769981.

[61] J.N. Sørensen. *General Momentum Theory for Horizontal Axis Wind Turbines*. Springer International Publishing, 2016. doi:10.1007/978-3-319-22114-4.

[62] F. Beyer, B. Luhmann, S. Raach, and P.W. Cheng. Shadow effects in an offshore wind farm potential of vortex methods for wake modelling. In *Proceedings of the Twelfth German Wind Energy Conference DEWEK*, Bremen, Germany, 5 2015. DEWEK. doi:10.18419/opus-3972.

[63] J.G. Schepers and S.P. van der Pijl. Improved modelling of wake aerodynamics and assessment of new farm control strategies. *J. Phys. Conf. Ser.*, 75(1):1–8, 2007. doi:10.1088/1742-6596/75/1/012039.

[64] K. Boormsa, M. Hartvelt, and L.M. Orsi. Application of the lifting line vortex wake method to dynamic load case simulations. *J. Phys. Conf. Ser.*, 753(2), 2016. doi:10.1088/1742-6596/753/2/022030.

[65] A. Platis, S.K. Sedersleben, J. Bange, A. Lampert, K. Bärfuss, R. Hankers, B. Canadilla, R. Foreman, J. Schulz-Stellenfleth, B. Djath, T. Neumann, and S. Emeis. First *in situ* evidence of wakes in far field behind offshore wind farms. *Scientific Reports*, 8, 2018. doi:10.1038/s41598-018-20389-y.

[66] J.H. Strickland, T.G. Smith, and K. Sun. A vortex model of the darrieus turbine: An analytical and experimental study. *The name of the journal*, 4(2):201–213, 7 1993. doi:10.1115/1.3449018.

[67] J.W. Oler, J.H. Strickland, B.J. Im, and G.H. Graham. Dynamic-stall regulation of the darrieus turbine. Technical report, Sandia National Laboratory, 1983. SAND-83-7029. doi:10.2172/5847876.

[68] K.R. Dixon. The near wake structure of a vertical axis wind turbine: Including the development of a 3d unsteady free-wake panel method for vawts. Master's thesis, Technical University of Delft, 2008. URL: http://resolver.tudelft.nl/uuid:ec380d2d-bdf0-4f1a-9149-b989a324f021.

[69] P. Balty, D.G. Caprace, J. Waucquez, M. Cocuelet, and P. Chatelain. Multiphysics simulations of the dynamic and wakes of a floating vertical axis wind turbine. In *Science of Making Torque from Wind (TORQUE 2020)*, 1618, 2020. doi:10.5194/wes-2-317-2017.

[70] J. Saverin, A. van Garrel, G. Pechlivanoglou, C.N. Nayeri, and C.O. Paschereit. Implementation of the multi-level multi-integration cluster to the treatment of vortex particle interactions for fast wind turbine wake simulations. In *ASME Turbo Expo*, Lillestrom, Norway, 6 2018. ASME. doi:10.1115/GT2018-76554.

[71] E. Branlard, G. Papadakis, M. Gaunaa, G. Winckelmans, and T.J. Larsen. Aeroelastic large eddy simulations using vortex methods: unfrozen turbulent and sheared inflow. *J. Phys. Conf. Seri.*, 625, 2015. doi:10.1088/1742-6596/625/1/012019.

[72] N. Ramos-Garcia, M.M. Hejlesen, J.N. Sørensen, and J.H. Walther. Hybrid vortex simulations of wind turbines using a three-dimensional viscous-inviscid panel method. *Wind Energy*, (20):1871–1889, 2017. doi:10.1002/we.2126.

[73] N. Ramos-Garcia, H.J. Spietz, J.N. Sørensen, and J.H. Walther. Vortex simulations of wind turbines operating in atmospheric conditions using a prescribed velocity-vorticity boundary layer model. *Wind Energy*, (21):1216–1231, 2018. doi:10.1002/we.2225.

[74] S.E. Widnall. The stability of a helical vortex filament. *J. Fl. Mech.*, 54:641–663, 1972. doi:10.1017/S0022112072000928.

[75] J.H. Walther, M. Guenot, E. Machefaux, J.T. Rasmussen, P. Chatelain, V.L. Okulov, J.N. Sørensen, M. Bergdorf, and P. Koumoutsakos. A

numerical study of the stability of helical vortices using vortex methods. *J Phys. Conf Series. The Science of Making Torque from Wind*, 75:397–415, 2007. doi:10.1088/1742-6596/75/1/012034.

[76] R. Cocle. *Combining the vortex-in-cell and parallel fast multipole methods for efficient domain decomposition simulations: DNS and LES approaches*. PhD thesis, Universite Catholique de Louvain, 7 2007. URL: http://hdl.handle.net/2078.1/5211.

[77] L. Jacquin, D. Fabre, D. Sipp, V. Theofilis, and H. Vollmers. Instability and unsteadiness of aircraft wake vortices. *Aero. Sc. & Tech.*, 7:577–593, 2003. doi:10.1016/j.ast.2003.06.001.

[78] S.C. Crow. Stability theory for a pair of trailing vortices. *AIAA*, (12):2172–2179, 12 1970.

[79] S.E. Widnall, D.B. Bliss, and C.Y. Tsai. The instability of short waves on a vortex ring. *J. Fl. Mech.*, 66(1):35–47, 1974. doi:10.1017/S0022112074000048.

[80] D. Sipp and L. Jacquin. Widnall instabilities in vortex pairs. *Phys. Fluids.*, 15(7):1861–1874, 2003. doi:10.1063/1.1575752.

[81] D. Fabre and L. Jacquin. Stability of a four-vortex aircraft wake model. *Phys. Fluids*, 12(10), 2000. doi:https://doi.org/10.1063/1.1289397.

[82] G. Winckelmans, R. Cocle, L. Dufresne, and R. Capart. Vortex methods and their application to trailing wake vortex simulations. *C.R. Physique*, (6):467–486, 2005. doi:10.1016/j.crhy.2005.05.001.

[83] P. Chatelain, A. Curioni, M. Bergdorf, D. Rossinelli, W. Anreoni, and P. Koumoutsakos. Billion vortex particle direct numerical simulations of aircraft wakes. *Comp. Meth, in App. Mech. & Eng.*, 197(13):1296–1304, 2008. doi:10.1016/j.cma.2007.11.016.

[84] H. Helmholtz. Über integrale der hydrodynamischen gleichungen, welche den wirbelbewegungen entsprechen. *Journal für die Reine und Angewandte Mathematik*, (55):25–55, 1858.

[85] G.K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge University Press, 1967. ISBN 9780511800955.

[86] E. Kreyszig. *Advanced Engineering Mathematics*. John Wiley & Sons, 10 edition, 7 2010. doi:10.1002/bimj.19650070232.

[87] J.P. Choquin and J.H. Cottet. Sur l'analyse d'une classe de methodes de vortex tridimensionnelles. *Comptes rendus de l'Académie des Sciences*, (306):739–742, 1988.

[88] G.S. Winckelmans and A. Leonard. Weak solutions of the three-dimensional vorticity equation with vortex singularities. *The Physics of Fluids*, 31(2810), 1988.

[89] A.J. Chorin. Numerical study of slightly viscous flow. *Journal of Fluid Mechanics*, 57(785), 1973. doi:10.1017/S0022112073002016.

[90] A. Leonard. Vortex methods for flow simulation. *J. Comp. Phys,*, 37(3):289, 1980. doi:10.1016/0021-9991(80)90040-6.

[91] M.M. Hejlesen. *A high order regularisation method for solving the Poisson equation and selected applications using vortex method*. PhD thesis, DTU Mechanical Engineering, 2016. URL: https://orbit.dtu.dk/en/publications/a-high-order-regularisation-method-for-solving-the-poisson-equati.

[92] R.W. Hockney and J.W. Eastwood. *Computer Simulation using Particles*. CRC Press, 1988.

[93] L.H. Thomas. Elliptic problems in linear differential equations over a network. Technical report, Columbia University, 1949. Watson Sci. Comput. Lab Report.

[94] Michael Bader, Hans-Joachim Bungartz, and Tobias Weinzierl, editors. *Advanced Computing*. Number 93 in Lecture Notes in Computational Science and Engineering. Springer-Verlag, Heidelberg, Berlin, 2013. doi:10.1007/978-3-642-38762-3.

[95] W. Hackbusch. *Multi-grid Method and Applications*, volume 4. Springer, 1985. doi:10.1007/978-3-662-02427-0.

[96] Denis-Gabriel Caprace, Thomas Gillis, and Philippe Chatelain. Flups: A fourier-based library of unbounded poisson solvers. *SIAM Journal on Scientific Computing*, 43(1):C31–C60, Jan 2021. URL: http://dx.doi.org/10.1137/19M1303848, doi:10.1137/19m1303848.

[97] R.A. James. The solution of poisson's equation for isolated source distributions. *The name of the journal*, (25):71–93, 1977. doi:10.1016/0021-9991(77)90013-4.

[98] K. Lackner. Computation of ideal mhd equilibria. *Comp. Phys. Comm.*, 12:33–44, 1976. doi:10.1016/0010-4655(76)90008-4.

[99] G.T. Balls and P. Colella. A finite difference domain decomposition method using local corrections for the solution of poissons equation. *J. Comp. Phys.*, (180):25–53, 2002. doi:10.1006/jcph.2002.7068.

[100] P Chatelain and P. Koumoutsakos. A fourier-based elliptic solver for vortical flows with periodic and unbounded directions. *J. Comp. Phys.*, (229):2425–2431, 2010. doi:10.1016/j.jcp.2009.12.035.

[101] J.T. Beale and A. Majda. Vortex methods i: Convergence and three dimensions. *Math. Comp.*, 39(159):1–27, 7 1982. doi:10.2307/2007617.

[102] J.T. Beale and A. Majda. Vortex methods ii: Higher order accuracy in two and three dimensions. *Math. Comp.*, 39(159):29–52, 7 1982. doi:10.2307/2007618.

[103] G. Pedrizzetti. Insight into singular vortex flows. *Fluid Dynamics Research*, 10(2):101–115, 7 1992. doi:10.1016/0169-5983(92)90011-K.

[104] E.A. Novikov. Generalized dynamics of three-dimensional vortical singularities. *Soviet Physics JETP*, 1993.

[105] G.H. Cottet and P Poncet. *Advances in Direct Numerical Simulations of 3D wall-bounded flows by vortex-in-cell methods*, volume 193. J. Comp. Phys-, 2003. doi:10.1016/j.jcp.2003.08.025.

[106] S.B. Pope. Ten questions regarding the large-eddy simulation of turbulent flows. *New J. Phys*, 6(35), 2004. doi:0.1088/1367-2630/6/1/035.

[107] J. Jimenez. Hyperviscous vortices. *J. Fluid Mech*, 279:169–176, 1994. doi:10.1017/S0022112094003861.

[108] A. van Garrel. *Multilevel Panel Method for Wind Turbine Rotor Flow Simulations*. PhD thesis, University of Twente - Enschede, 2016. doi:10.3990/1.9789036542241.

[109] J.P. Berrut and L.N. Trefethen. Barycentric lagrange interpolation. *Siam Review*, 26:501–517, 2004. doi:10.1137/S0036144502417715.

[110] K. Ehrenfriend. *Stroemungsakustik: Skript zur Vorlesung*. Mensch & Buch, 2004.

[111] Tux Family. Eigen, 2021. URL: https://eigen.tuxfamily.org/index.php?title=Main_Page.

[112] C. Nugteren. CLBlast: A Tuned OpenCL BLAS Library. *IWOCL'18: International Workshop on OpenCL*, 53(188):485–507, 2018.

[113] Nvidia. cuBLAS, 2021. URL: https://developer.nvidia.com/cublas.

[114] Khronos. Opencl, 2021. URL: https://www.khronos.org/opencl/.

[115] OpenMP Architecture Review Board. Openmp, 2021. URL: https://www.openmp.org/.

[116] Kitware. Paraview, 2021. URL: https://www.paraview.org/.

[117] H. Samet. An overview of quadtrees, octtrees, and related hierarchical data structures. *NATO ASI Series F: Computer and Systems Sciences*, 40:51–68, 1988. doi:10.1007/978-3-642-83539-1_2.

[118] G.H. Cottet and P.D. Koumoutsakos. *Vortex Methods: Theory and Practice*. Cambridge University Press, 2000. doi:10.1017/CBO9780511526442.

[119] M. Patra and M. Karttunen. Stencils with isotropic discretisation error for differential operators. *Numerical Methods for Particle Differential Equations*, 7 2006. doi:10.1002/num.20129],.

[120] H.K. Moffatt. The degree of knottedness of tangled vortex lines. *Journal of Fluid Mechanics*, 35(1):117–129, 1969. doi:10.1017/S0022112069000991.

[121] P.G. Saffman. The velocity of viscous vortex rings. *Stud. App. Math.*, 49(4):371–380, 12 1970. doi:10.1002/sapm1970494371.

[122] S.K. Stanaway, B.J. Cantwell, and P.R. Spalart. A numerical study of viscous vortex rings using a spectral method. Technical report, NASA, 10 1988. Technilca Memorandum 101041. URL: https://archive.org/details/nasa_techdoc_19890014449.

[123] Le Dizes S. Williamson C.H.K. Leweke, T. Dynamics and instabilities of vortex pairs. *Annual Review of Fluid Dynamics*, 48:507–541, 2016. doi:10.1146/annurev-fluid-122414-034558.

[124] P Chatelain, D. Kivotides, and A. Leonard. Reconnection of colliding vortex rings. *Physical Review Letters*, 3 2003. doi:10.1103/PhysRevLett.90.054501.

[125] S.E. Widnall and J.P. Sullivan. On the stability of vortex rings. *Pril. Trans. R. Soc. Lond.*, 332:335–353, 1973. doi:10.1098/rspa.1973.0029.

[126] P.G. Saffman. The number of waves on unstable vortex rings. *J. Fluid. Mech.*, 84(4):625–639, 1978. doi:10.1017/S0022112078000385.

[127] M. Van Dyke. *An Album of Fluid Motion*. Parabolic Press, 14 edition, 1982. doi:10.1017/S0022112083212888.

[128] K. Shariff, R. Verzicco, and P. Orlandi. A numerical study of three-dimensional vortex ring instabilities: viscous corrections and early nonlinear stage. *J. Fluid Mech.*, 279:351–375, 1994. doi:10.1017/S0022112094003939.

[129] T. Maxworthy. The structure and stability of vortex rings. *J. Fluid Mech.*, 51(1):15–32, 1972. doi:10.1017/S0022112072001041.

[130] J. Norbury. A family of steady vortex rings. *J. Fl. Mech.*, (3):417–431, 1973. doi:10.1017/S0022112073001266.

[131] A. Kolmogorov. The local structure of turbulence in incompressible viscous fluid for very large reynolds' numbers. *Doklady Akademiia Nauk SSSR*, 1941.

[132] G.K. Batchelor and A.A. Townsend. The nature of turbulent motion at large wave numbers. *Proc. Roy. Soc. London*, 199:238–255, 1949. doi:10.1098/rspa.1949.0136.

[133] L. Prandtl. Traglügeltheorie. *Nachrichten v.d. Gesselschaft der Wissenschaften zu Göttingen*, 1918.

[134] J.D. Anderson. *Fundamentals of Aerodynamics*. McGraw-Hill Education, 6 edition, 7 2016. ISBN 1259251349.

[135] Ortega J.M., R.L. Bristol, and Ö. Savas. Experimental study of the instability of unequal-strength counter-rotating vortex pairs. *J. Fluid Mech.*, 2003.

[136] J.G. Schepers, K. Boorsma, T. Cho, S. Gomez-Iradi, P. Schaffarczyk, W.Z. Shen, T. Lutz, K. Meister, B. Stoevesandt, S. Schreck, D. Micallef, R. Pereira, T. Sant, H.A. Madsen, and N. Sørensen. Final report of iea task 29: Mexnext (phase 1). Technical report, ECN, 2012. doi:10.1088/1742-6596/555/1/012089.

[137] J.G. Schepers and H. Snel. Model experiments in controlled conditions-final report. Technical Report ECN-E-07-042, Energy Research Centre of the Netherlands, 2007. doi:10.1088/1742-6596/753/2/022004.

[138] H. Snel, G. Schepers, and N.N. Siccama. Mexico project: The database and results of data processing and interpretation. In *47th AIAA Aerospace Sciences Meeting*. AIAA, 1 2009. doi:10.2514/6.2009-1217.

[139] QBlade Team. Qblade: Wind turbine design and simulation, 2021. URL: http://www.q-blade.org/.

[140] D. Marten. Implementation, optimization and validation of a nonlinear lifting line free vortex wake module withing the wind turbine simulation code qblade. 6 2015. doi:10.1115/1.4031872.

[141] J. Wendler, D. Marten, G. Pechlivanoglou, C.N. Nayeri, and C.O. Paschereit. An unsteady aerodynamics model for lifting line free vortex wake simulations of hawt and vawt in qblade. In *ASME Turbo Expo 2016*. ASME, 6 2016. doi:10.1115/GT2016-57184.

[142] J. Saverin, G. Persico, D. Marten, V. Dossena, G. Pechlivanoglou, and C.O. Paschereit. Advanced medium-order modelling for the prediction of the three-dimensional wake shed by a vertical axis wind turbine. In *Proc. ASME Turbo Expo*, Lillestrom, Norway, 6 2018. ASME. GT2018-76575. doi:10.1088/1742-6596/1037/6/062029.

[143] L.A. Martinez-Tossas, M.J Churchfield, and C. Meneveau. Optimal smoothing length scale for actuator line models of wind turbine blades based on gaussian body force distribution. *Wind Energy*, 4(20):1083–1096, 2017. doi:10.1002/we.2081.

[144] L.J. Vermeer. Measurements on the properties of the tip vortex of a rotor model. In Tsipouridis JL, editor, *European Wind Energy Conference*, pages 805–808, Thessaloniki, Greece, 1994.

[145] S. Gomez-Iradi, R. Steijl, and G.N. Barakos. Development and validation of a cfd technique for the aerodynamic analysis of hawt. *J. Solar Energy Engineering*, 131, 2009. doi:10.1115/1.3139144.

[146] D.G. Caprace, G. Winckelmans, and P. Chatelain. An immersed lifting and dragging line model for the vortex particle method. *Th. Comp. Fl. Dyn.*, 34(1):21–48, 2020. doi:10.1007/s00162-019-00510-1.

[147] J. Frederik, B.M. Doekemeijer, S.P. Mulder, and J.W. van Wingerden. The helix approach: Using dynamic individual pitch control to enhance wake mixing in wind farms. *Wind Energy*, 5 2020. doi:10.1002/we.2513.

[148] Diego Rossinelli, Babak Hejazialhosseini, Wim van Rees, Mattia Gazzola, Michael Bergdorf, and Petros Koumoutsakos. MRAG-i2d: Multi-resolution adapted grids for remeshed vortex methods on multicore architectures. *J. Comput. Phys.*, 288:1–18, may 2015. URL: `http://www.cse-lab.ethz.ch/wp-content/papercite-data/pdf/rossinelli2015a.pdf`, `doi:10.1016/j.jcp.2015.01.035`.

[149] J.M. Burgers. A mathematical model illustrating the theory of turbulence. *Advances in Applied Mechanics*, (1):171–199, 1948. `doi:10.1016/S0065-2156(08)70100-5`.

[150] T.S. Lundgren. Strained spiral vortex model for turbulent fine structures. *Physics of Fluids*, (25):2193–2203, 1982. `doi:10.1063/1.863957`.

[151] A.A. Townsend. On the fine-scale structure of turbulence. *Proceedings of the Royal Society of London*, pages 534–542, 1951. `doi:10.1098/rspa.1951.0179`.

[152] Pullin D.I. Buntine, J.D. Merger and cancellation of strained vortices. *J. Fl. Mech.*, (205):263–95, 1989. `doi:10.1017/S002211208900203X`.

[153] J.C. Hardin. The velocity field induced by a helical vortex filament. *Physics of Fluids*, 25(11):1949–1952, 7 1982. `doi:10.1063/1.863684`.

[154] A. Gray and G.B. Mathews. *A Treatise on Bessel Functions and Their Applications to Physics*. Macmillan and Co., 1895.

[155] Y. Fukumoto and V.L. Okulov. The velocity field induced by a helical vortex tube. *Physics of Fluids*, (17), 2005. `doi:10.1063/1.2061427`.

[156] M.J. Hill. On a spherical vortex. *Philosophical Transactions of the Royal Society of London*, 185:213–245, 1884. `doi:10.1098/rsta.1894.0006`.

[157] M.H. Hejlesen, J.T. Rasumussen, P. Chatelain, and J.H. Walther. A high order solver for the unbounded poisson equation. *J. Comp. Phys.*, (252):458–467, 2013. `doi:10.1016/j.jcp.2013.05.050`.

[158] J.D.A. Walker and C.R. Smith. The impact of a vortex ring on a wall. *J. Fluid Mech.*, 181:99–140, 1987. `doi:10.1017/S0022112087002027`.

[159] H. Helmholtz and Tait P.G. On integrals of the hydrodynamical equations, which express vortex motion. *Phil. Mag. and J. Sci*, pages 485–512, 1867. `doi:10.1080/14786446708639824`.

[160] Lord Kelvin. Vibrations of a columnar vortex. *Phil. Mag.*, 10:155–168, 1880. doi:10.1080/14786448008626912.

[161] The Old Motor. Blowing smoke rings...., 2011. URL: https://theoldmotor.com/?p=11354.

[162] A.B. Basset. *A treatise on hydrodynamics, with numerous examples.* doi:10.1038/040412a0.

[163] B.J. Cantwell. Viscous starting jets. *J. Fl. Mech*, 175:159–189, 12 1986. doi:10.1017/S002211208600112X.

[164] C.H. Krutsch. Über eine experimentell beobachtete erscheinung an wirbelringen bei ihrere translatorischen bewegung in wirklichen flüßigkeiten. *Ann. Phys*, 1939.

[165] S.E. Widnall and C.Y. Tsai. The instability of the thin vortex ring of constant vorticity. *Pril. Trans. R. Soc. Lond.*, 287(1322):273–305, 1977. doi:10.1098/rsta.1977.0146.

[166] Mads Mølholm Hejlesen, Grégoire Winckelmans, and Jens Honoré Walther. Non-singular green's functions for the unbounded poisson equation in one, two and three dimensions. *Applied Mathematics Letters*, 89:28–34, 2019. doi:10.1016/j.aml.2018.09.012.

Appendix A Validation: 2D Flow Cases

As an addendum to the validation of the solver for 3D cases in Chapter 5, the results of the application of the `GML` and `PML` solvers to 2D cases are demonstrated here. The results for the solver performance have not been included here however relatively comparable results to those for the 3D cases can be assumed. The validation occurs along the same lines as that carried out for 3D flows in Chapter 5.

Description of Validation Procedure

As was carried out for 3D cases, the global and local flow quantities are monitored as the particle set evolves as is dictated by the equations of motion outlined in Chapter 2. This implies observation of global and local flow quantities, along with qualitative observation of the evolution of the particle field.

Simulation Parameters It should be noted that the simulation parameters conceptually are identical to those of the 3D cases (see Section 5.1.1). All parameters are as described there with the following exceptions:

- Vorticity is a scalar quantity: Having only a component in z direction: $\vec{\omega}_{2D} = \omega_z \vec{e}_z$.
- Characteristic volume is calculated with: $dV_{char} = H_{char}^2$
- For a 2D flow the term $\nabla \vec{u} \cdot \vec{\omega}$ vanishes as these vectors are always orthogonal, and hence no vortex stretching occurs. The choice of stretching scheme is therefore irrelevant for 2D simulations.
- The only physical process which gives rise to a rate of change of circulation is viscous diffusion $\nabla^2 \omega_z$.
- Divergence filtering need not be applied here as the particle set remains divergence free.
- No turbulence modelling has been applied here.

Flow Diagnostics For the 2D validation, observation of the three linear diagnostics was made. These are the circulation Γ (5.1), linear momentum L (5.2) and angular momentum A (5.3).

Case 1: Inviscid Elliptical Vortex

This case describes the decay of an inviscid vortex with an initially elliptical distribution of vorticity. Here the ability of the solver to correctly capture induced velocities and convection driven phenomena is shown. Following Koumoutsakos [39] (hereafter referred to as KOUM) the initial vorticity distribution is defined as:

$$\omega(x,y) = \begin{cases} \Lambda\left[1 - \exp\left(-\frac{q}{z}\exp(\frac{1}{z-1})\right)\right] & \text{z} \leq 1 \\ 0 & \text{z} > 1 \end{cases}, \; z = \frac{\sqrt{\left(\frac{x}{2}\right)^2 + y^2}}{R_0}, \tag{A.1}$$

where $\Lambda = 20\,\mathrm{m^2 s^{-1}}$, $q = 2.56085$ and $R_0 = 0.8$, following KOUM.

Simulation parameters Simulation parameters were chosen to mimic those of the simulations carried out in KOUM, with an exception being the time step dT. A different integration scheme was used here, the `AB2LF` method. Particle strengths were not modified as the process is assumed to be inviscid. The time step was taken as $dT = 2.5\text{e}{-4}\,s$. As with 3D simulations, the characteristic core size is chosen such that $\sigma = 1.1\,H$ to ensure particle overlap after remeshing. Here the grid size $H = 6.36\text{e}{-4}\,\mathrm{m}$. The minimum base box size has been chosen such that $H_b = 10\,H$. With these parameters, the initial field has 88760 particles.

Global Flow Diagnostics As the process is assumed inviscid, the total circulation of the field should be conserved. There is in fact no mechanism for changing the circulation of the particles in the implemented code if the simulation is inviscid, however the process of remeshing redistributes the circulation spatially, so monitoring this quantity here is essentially a check on the remeshing scheme. The time evolution of the circulation and linear momentum are displayed in Fig. A.1. The ability of the remeshing scheme to preserve circulation is demonstrated as the value remains practically constant. Furthermore the conservation of momentum is also displayed in that the linear momentum remains constant. As the simulation is symmetric in terms of the planes $x = 0$ and $y = 0$ (also visible in Fig. A.2), it follows that angular momentum is conserved, and this was observed but is not shown for brevity.

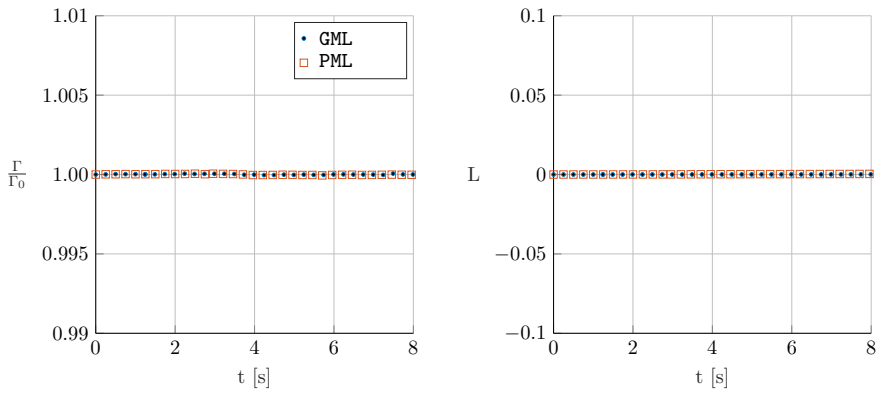

Figure A.1: Evolution of global diagnostics for an inviscid ellipse. Left: Circulation, Right: Linear momentum.

Evolution of Flow Field

Local Flow Diagnostics The results of the evolution of the vorticity field along the x axis are displayed in Fig. A.2. Results are compared to the simulations carried out in KOUM. The evolution of the vorticity field for the times $t = 0$, 1, 1.5, 4, 8 and 12 s are shown. In addition to the graphical plots shown here a visualisation of the vorticity field at the different times is provided in Fig. A.3. This demonstrates just how well the two solvers agree. The results for the evolution of the vorticity agree perfectly, although the methodology between the `GML` and `PML` solvers differs greatly. Furthermore, both solvers agree excellently with the results of KOUM, indicating that the flow field is evolving as expected. The process of *narrowing* of the vortex core section is seen to occur and agrees perfectly in all cases, this is part of the process of axysymmetrisation (phenomenology described in proceeding section). The filamentation occuring at different stages is seen in the *spikes* occurring lateral to the core of the ellipse. Further filamentation with more arms is seen as the process evolves. The dilation of the filaments is also well captured as they tend to spread away from the core. This should not be confused with diffusion, as the process occurring here is inviscid. Results for the vorticity distribution along the axis $x = 0$ were also in the comparative document reported, however the agreement was equally excellent, and hence showing these results in addition was deemed to be superfluous.

Flow Phenomena Two distinct phenomena are seen to occur here for this flow. The first of these is the process of vortex filamentation. The arms thrown away from the vortex in the initial stages of development are the

result of the strong vorticity in the centre of the vortex giving rise to a stronger rotation within the vortex core. This occurs increasingly as the process evolves until the individual arms are no longer discernible from each other. Colour contours represent the simulation with the `GML` solver, and solid contours represent the simulation with the `PML` solver. Attention is drawn here again to the excellent agreement between the two solvers. The strong shearing produced at this interface and the non-monotonic vorticity profile seen at the edges of the main vortex core (see Fig. A.2, $t = 1\,s$) give rise to a Kelvin-Helmholtz style instability which is visible in Fig. A.2 and Fig. A.3 at $t = 1.5\,s$ as jumps in vorticity near the edge of the core domain.

In addition to this process, the axisymmetrisation of the elliptical vortex is seen to occur as the simulation progresses. Vortex core and general vortex arm distribution becomes more radially symmetric. This occurs as the filamentation results in a *shedding-off* of the asymmetries via the filamentation which distributes the asymmetrical vorticity into concentric filaments around the vortex core. This is typical for this style of initial vorticity distribution and the convergence towards a stable symmetric vorticity profile is observed, however for other initial vorticity distributions this is not necessarily the case [42].

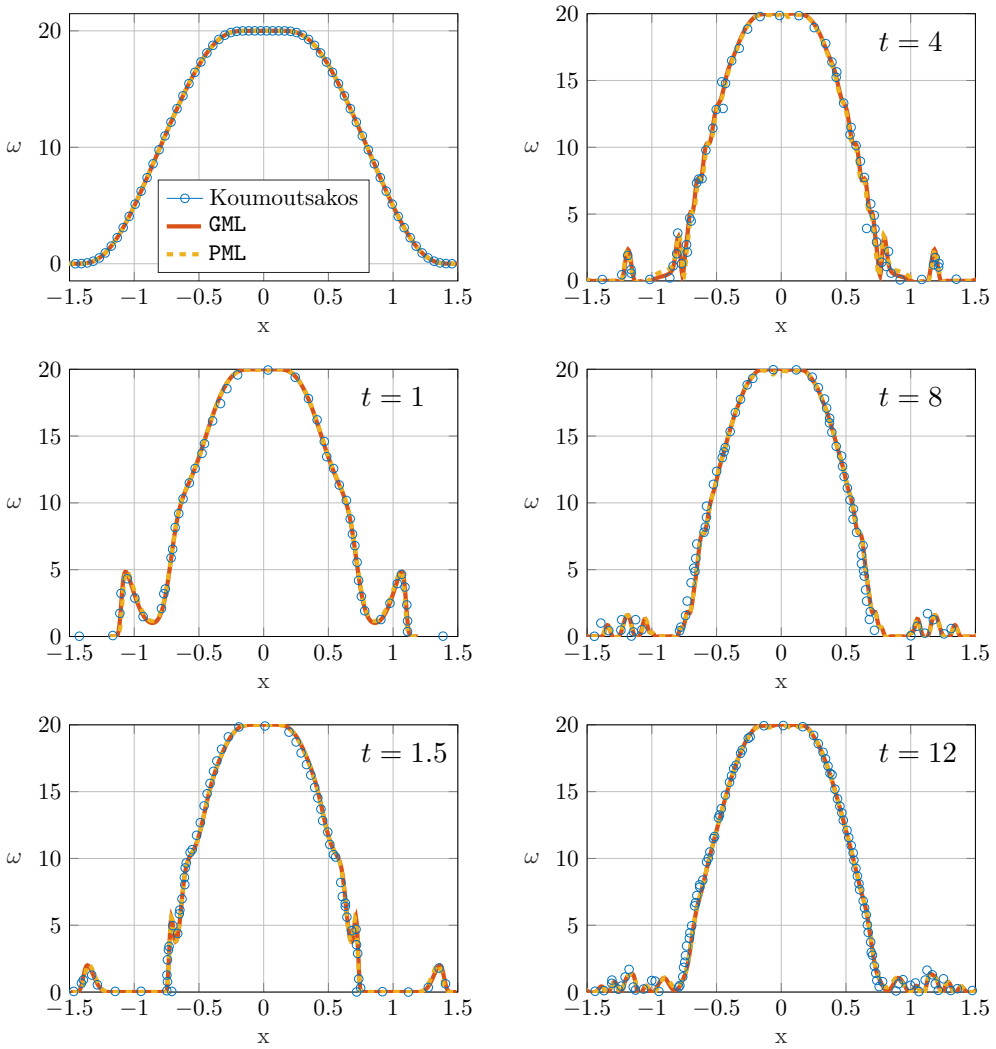

Figure A.2: Distribution of ω along the axis $y = 0$. Comparison to the data of Koumoutsakos [39] shows excellent quantitative agreement in terms of both contraction of the central region of vorticity and filamentation at the vortex perimeter.

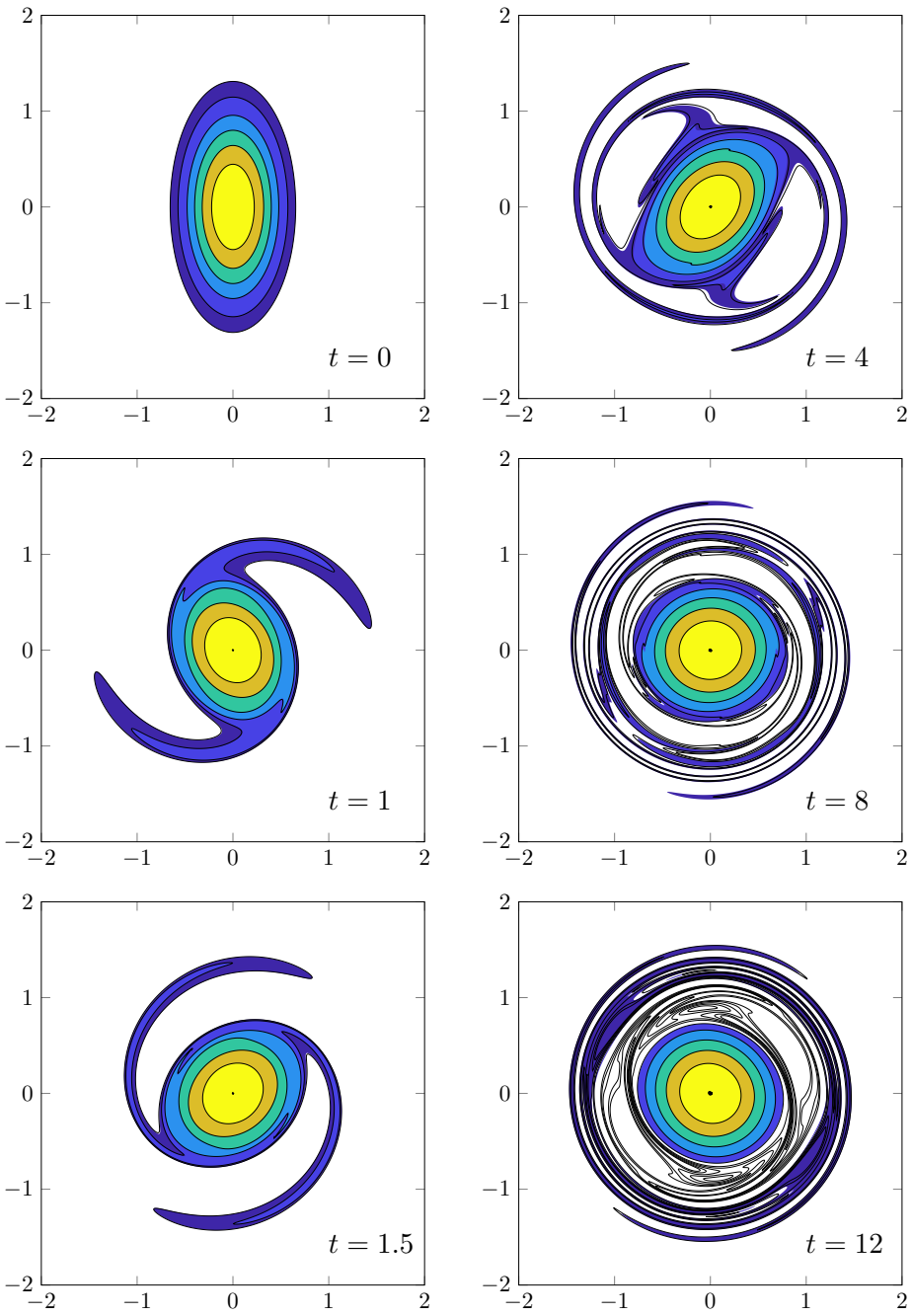

Figure A.3: Evolution of an inviscid elliptical vortex. Colour contours correspond to the GML solver. Solid contours correspond to PML solver. Contours are $|\omega| = 0.25$, 0.5, $1 : 2 : 20$. Agreement between the solvers is seen to be excellent. Qualitative comparison with Koumoutsakos [39] shows vortex evolution to be visually identical. For the plots $t = 8$,12, lower contours of $|\omega|$ have been omitted for clarity of detail.

Case 2: Strained Merging Vortices

Here the ability of the solver to correctly capture viscous interaction and shear-driven phenomena is shown. The case of vortex merging and cancellation under the influence of either an ambient or self-induced shear field is of fundamental importance in turbulence research [149, 150, 151]. A common simulation case is that of a vortex evolving within a straining field $\vec{u}_s$ of the form:

$$\vec{u}_s = -\beta(t)\, x\, \vec{e}_x + [\beta(t) - \gamma(t)]\, y\, \vec{e}_y + \gamma(t)\, z\, \vec{e}_z\,. \tag{A.2}$$

When $\beta = \frac{1}{2}\gamma$, the strain field is axysymmetric. A steady analytical solution for this case was found by Burgers [149]. This solution stands as one of the few fundamental solutions to the vorticity transport equation in 3D. The so-called Burgers' vortex arises when the action of viscous diffusion is balanced by the vorticity intensification which results from the strain field imposed on the flow, and hence acts as an excellent validation case for viscous and shearing effects. The simulation captures not only vortex merging, an important phenomena for coalescence of vortical structures in a wake region, but also the action of diffusion as the simulation converges.

Conversion to 2D field The shear field given above describes a fully 3D flow. Under an appropriate transformation the original vorticity field formulated within a cylindrical coordinate system $\omega(r, \theta, z)$ can be described completely with a 2D solution in a new system Ω satisfying $\Omega(\xi, \theta, \tau) = \omega(r, \theta, t)e^{-A(t)}$ where ξ is a rescaled radial coordinate, τ a stretched time coordinate and A the stretching factor based on $\gamma(t)$. This allows a flow with initial conditions $\omega(r, \theta, t = 0)$ to be simulated for a fully 3D flow. For details the reader is referred to the work of Lundgren [150] and Batchelor [85].

This was simulated for a range of effective Reynolds number Re_ω by Buntine & Pullin [152]. Their solver made use of a spectral solver combined with a finite differences, not entirely dissimilar to the `PML` method developed in this work however with assumed modes of the Fourier coefficients of ψ and ω. The results of this study shall be used as a validation for the modelling of the viscous component of the vorticity transport equation. For the case here $\gamma = 4$, $\beta = 2$ and represents therewith an axysymmetric strain field. The initial vorticity distribution is given by:

$$\omega(x, y) = -\frac{\Gamma}{2\pi}[e^{-(x-x_0)^2-y^2} + e^{-(x+x_0)^2-y^2}]\,. \tag{A.3}$$

This represents physically two Gaussian vortices with equal sign, the total circulation is hence Γ. The initial distribution is shown in Fig. A.6.

Simulation Parameters The kinematic viscosity of the fluid is taken as unity. The only remaining free parameter is the effective Reynolds number $Re_\omega = \Gamma/2\pi\nu$, which for comparative purposes was set here to $Re_\omega = 160$. As described in Chapter 2, the resolution of viscous effects are carried out in two quite different ways depending on the choice of solver. The characteristic core size is specified as described above. Here the grid size has been specified as $H = 7.5\text{e}{-2}\,\text{m}$. The minimum base box size has been chosen such that $H_b = 10\,H$. The integration time step has been set to $dt = 1\text{e}{-3}\,s$. With these settings the initial field has 13080 particles.

The output of the solver is valid within the transformed field Ω and must hence be appropriately scaled spatially and temporally with the factors ξ and τ, for details see [150]. It was observed that the remeshing gave rise to spurious regions of negative vorticity on the boundary of the particle set after remeshing due to the choice of the M'_4 scheme which would give rise to instabilities. For this reason the magnitude filtering factor was decreased to $F_{mag} = 10^{-7}$. Furthermore, for the `GML` method it was found that a slightly larger H_{ml} was necessary to capture the PSE correctly directly in the centre of the domain, hence why this was increased to $15 \cdot H_{char}$. This was not necessary for the `PML` solution.

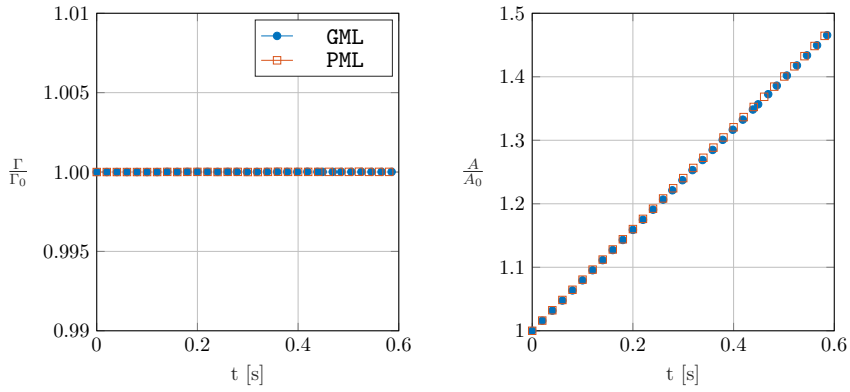

Figure A.4: Evolution of global diagnostics for a strained vortex. Left: Circulation, Right: Angular momentum.

Global Flow Diagnostics The evolution of the circulation and angular momentum for this case is displayed in Fig. A.4. It can be observed that again the circulation is practically constant. This represents two important modelling processes. The diffusion of vorticity occurring is being conserved properly by the diffusion models implemented in both cases. The second

important observation is that the total circulation is not significantly effected by the magnitude filtering being applied.

As this solution represents a transformed 3D flow although the simulation appears to show an increase in angular momentum, the reality is that for the real 3D solution this angular momentum is being stretched out of the plane, and hence a speed up of the core is expected. This is seen as a increase in the angular momentum of the core section and is well observed here. Further amplitude scaling would be required to calculate the actual representative 2D angular momentum. Although this could be employed, this was not followed here as it was deemed unnecessary for the purposes of the 2D validation when the process occurring was intuitive. Despite this, results are shown which display the excellent agreement between the two solvers. As with the case of the inviscid ellipse, the total circulation was seen to be conserved.

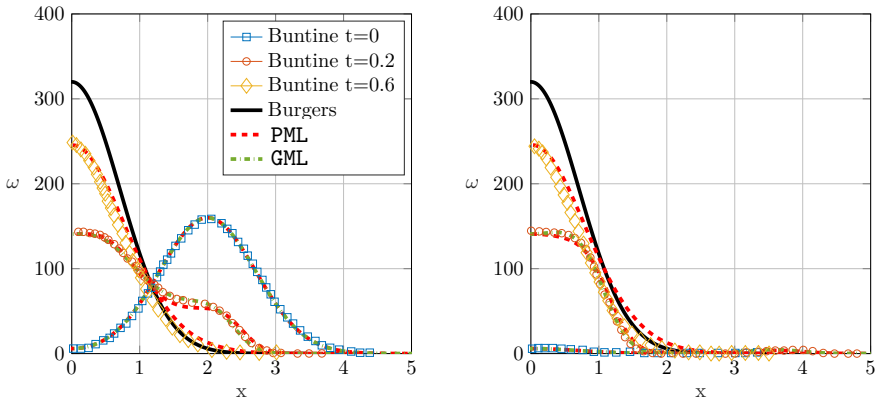

Figure A.5: Vorticity distribution for a set of counter-rotating viscous vortices. Values are compared to those calculated in [152]. The convergence of the solution towards the steady Burgers vortex is seen.

Evolution of flow field

Local Flow Diagnostics In order to demonstrate that viscous effects are correctly being captured, the transformed results for the vorticity distribution along the axes $x = 0$ and $y = 0$ for numerous time values for both solvers are displayed in Fig. A.5 along with the simulated results from [152]. The solution to the steady-state Burgers vortex as $t \to \infty$ is shown. The agreement between the two solvers is again seen to be excellent and the solutions appear to be converging as predicted.

In addition to the values plotted here, the vorticity field is demonstrated for a range of time values for both solvers in Fig. A.6. The merging process is clearly seen to occur and a qualitative comparison to [152] shows practically identical progression of the vorticity field in time.

Flow Phenomena With the introduction of viscous effects a number of important features are captured. The qualitative behaviour agrees well with the results in [152]. Initially the vortex cores rotate about each other with an angular velocity which can be approximated with a point-vortex model. The regions of high vorticity begin to spiral together and eventually merge into a single region of strong vorticity. The core regions rotate increasingly rapidly due to the cumulative effect of the merging vortex regions. The effect of viscosity acts to locally diffuse the vorticity gradients on much shorter time scales than those over which the vortex converges to a single Burgers vortex.

Conclusion

Two test cases have been investigated to validate the `GML` and `PML` solvers for 2D flows. Here the physics of the problem is less complicated than the 3D case as the vorticity exists in a single plane and the effect of stretching automatically vanish. Numerous physical processes were quantitatively and qualitatively captured for the process of an evolving inviscid elliptical vortex. The ability of the 2D solvers to model the action of viscous diffusion was validated against an analytical Burger's vortex and the solvers were both seen to perform well.

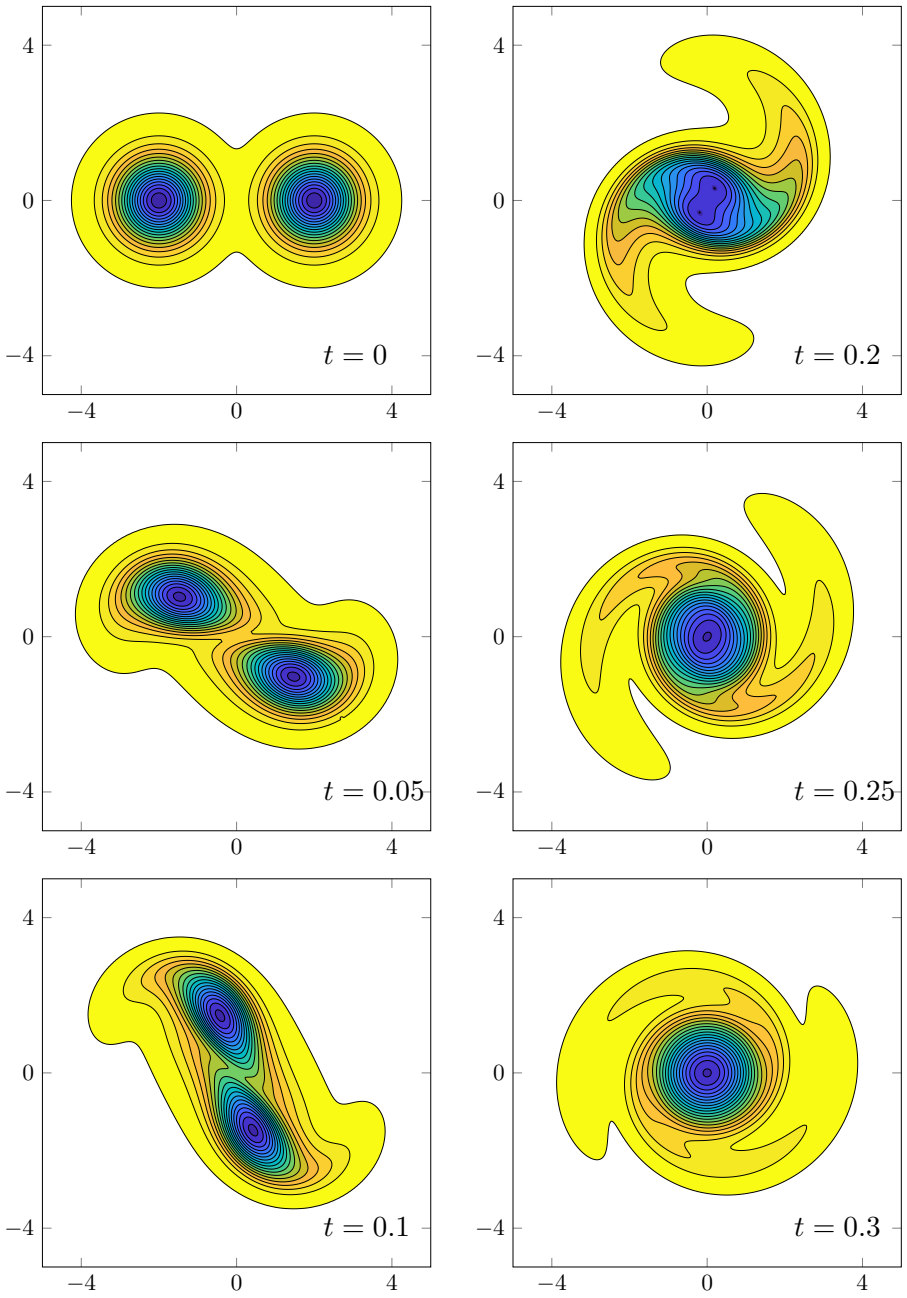

Figure A.6: Evolution of a viscous vortex pair. Colour contours correspond to the `GML` solver. Solid contours correspond to `PML` solver. Contours are $|\omega| = 1 : 9 : \omega_{max}$. Qualitative comparison with Buntine & Pullin [152] shows the vortex merging process to be visually identical.

Appendix B Validation: 3D Vortex Filaments

The `GML` solver will be validated here for the case that the flow field is being represented with vortex filaments. As opposed to scenarios where straight filaments are used, filaments are represented within the VPM as curves discretised with vortex particles. This removes the assumption of a straight vortex line and allows more complicated geometries to be investigated. Vortex tubes are not represented with a continuous distribution, but rather the circulation of the vortex tube is *lumped* onto a particle at the centre of vorticity with an appropriate circulation. This representation is generally unsuitable for treatment with the `PML` solver, as the FP solver relies on a continuous distribution of vorticity. For this reason the results in the following pertain only to the use of the `GML` solver. The case inspected here is the flow field induced by a thin infinitely long helical vortex filament as illustrated in Fig. B.1. The helix is defined with the parametrisation used in Chapter 3, briefly repeated here: radius a, helix pitch h, giving a helix length per winding of $L = 2\pi\, h$.

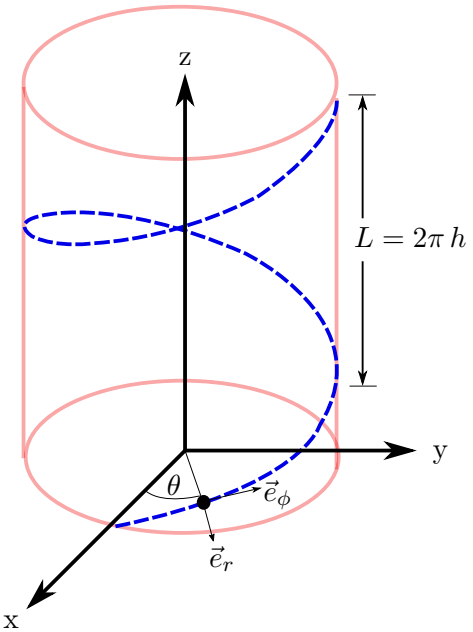

Figure B.1: Coordinate system used for a helical vortex filament.

Description of flow The analytical solution for the induced velocity field and stream function due to an infinitely thin vortex filament was derived by Hardin [153][1]. Here the solution for the velocity $\vec{u}$ is expressed in cylindrical coordinates (r, ϕ, z):

$$u_r = \frac{\Gamma}{\pi k^2} H^{1,1}\,, \quad u_\phi = \frac{\Gamma}{2\pi r} + \frac{\Gamma a}{r\pi k} H^{1,0}\,, \quad u_z = \frac{\Gamma}{2\pi k} - \frac{\Gamma a}{\pi k^2} H^{1,0}\,, \tag{B.1}$$

where H is a function closely related to the Kapteyn series (a product of Bessel functions). This is defined as:

$$H^{i,j} = \sum_m^\infty m K_m^i \left(\frac{am}{k}\right) I_m^j \left(\frac{rm}{k}\right)\,, \tag{B.2}$$

where I and K are the first and second modified Bessel functions [154] and the superscript index implies differentiation with respect to the argument. This expression is valid only for $r < a$. A separate solution is obtained for values $r > a$ which is relatively similar. An advancement upon this was achieved by Fukumoto and Okulov [155], where an asymptotic expansion was used to increase the accuracy of the prediction. Considering the terms in the Hardin solution as a vortex monopole, the Fukumoto solution added a dipole term which allows for a distribution of vorticity in the vortex core. This higher order correction is not accounted for here as the focus is made on the far field term, where the dipole effects vanish. The validation is carried out by initially demonstrating that direct evaluation of the VP method accurately predicts the velocity field. Following this, the `GML` solver will be investigated and the solution compared to direct evaluation.

Direct evaluation For the comparisons made here a vortex with 40 windings in $\pm z$ directions has been constructed. Trial and error demonstrated that this was sufficient to produce a pseudo-infinite influence (an observation of practical interest in the consideration of wind turbine wake modelling). Furthermore, the Kapteyn series was truncated to $m = 50$. The results are shown in Fig. B.2. The vortex particle is treated as having a singular distribution (modelling option `REG=SINGULAR`) for consistency with the assumption of an infinitely thin vortex in the analytical solution. The grid size has been chosen as $H = 0.01\,\mathrm{m}$ which discretises each helix winding with approximately 10^3 particles.

The induced velocities as a function of azimuthal angle ϕ are first compared for $r = 0.25\,a$, $0.75\,a$, and $1.25\,a$, the results for $\phi > \pi$ are not shown

[1] It is in fact stated in Hardin that the solution is practically used for validation of Biot-Savart routines

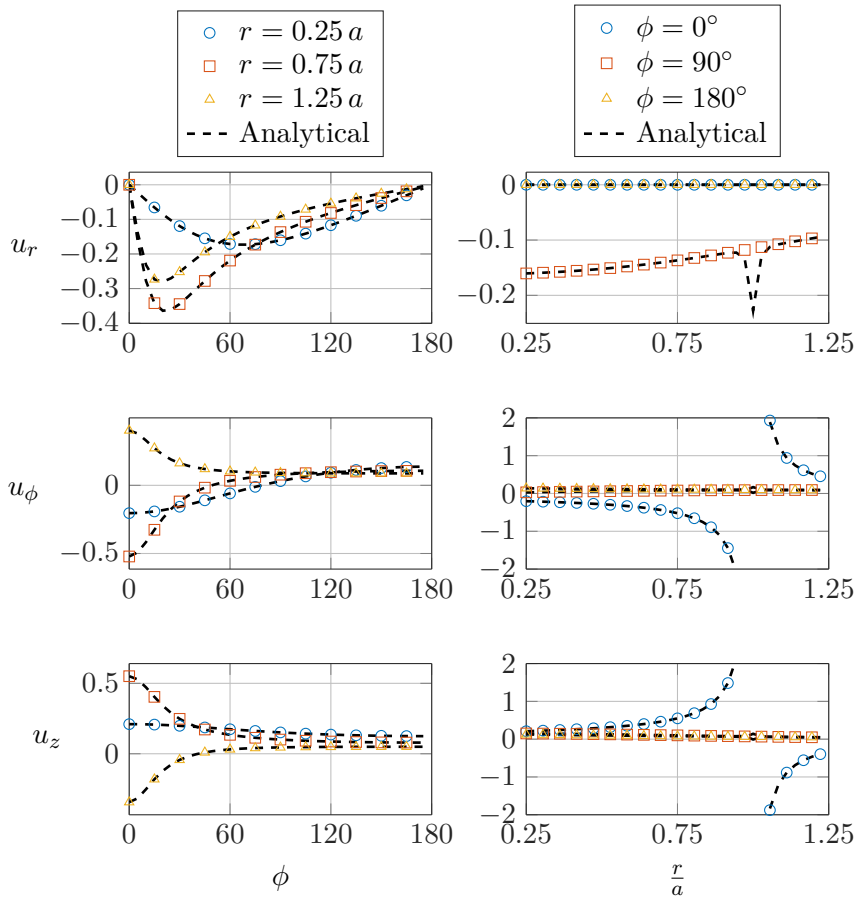

Figure B.2: Velocity components in a polar coordinate system induced by a single infinite helical vortex filament. Analytical results are shown with dashed lines. Left: Along lines $r = const.$ Right: Along lines $\phi = const.$

due to the symmetry of the solution. The position $r = a$ is not inspected as this leads to divergence in the Kapteyn series, where the analytical solution is undefined, an issue avoided in the treatment of Fukumoto & Okulov [155]. It is seen that the results are essentially perfect and no discrepancy can be visibly observed. As is physically intuitive, the velocity components u_ϕ and u_z due to the vortex filament peak as the evaluation point nears the vortex filament at $\phi = 0°$. Velocity predictions along the lines $r = const.$ demonstrate essentially similar results, however in this case the singularity at $r = a$ is clearly captured. As expected at positions $\phi = 0°$ and $\phi = 180°$ the radial velocity vanishes due to the antisymmetry of the vortex filament about the plane $x = 0$. As with the previous results, the induced velocity asymptotes like r^{-1} as the vortex filament is radially

approached and antisymmetric velocities are predicted as expected.

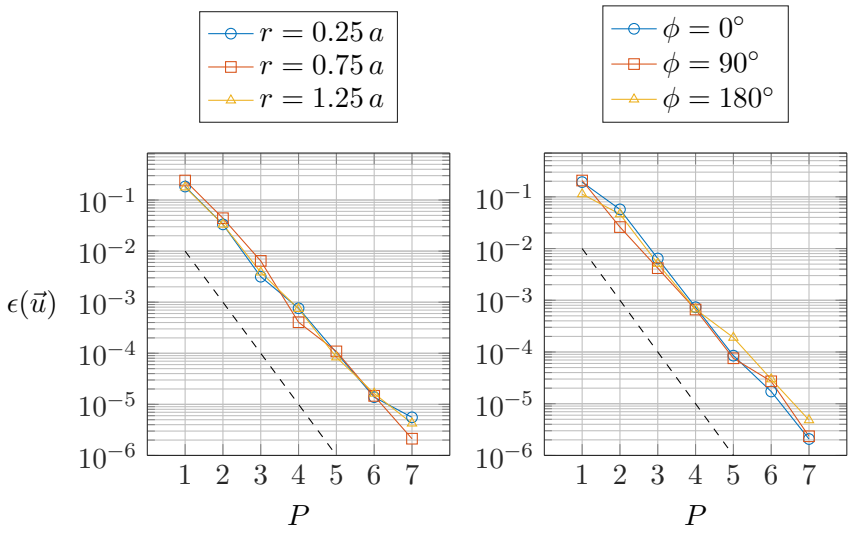

Figure B.3: Relative error induced by the GML solver as compared to direct evaluation.

Accuracy of GML method: singular filaments The results predicted by the GML shall now be validated against those from direct evaluation to ensure that the velocity prediction behaves as demonstrated in the multilevel chapter 3. In this case again singular regularisation of the vortex core has been used. The grid size has been set for consistency with that from the previous demonstration, $H = 0.01\,\text{m}$. The minimum grid box size has been set to $H_b = 8\,\sigma$ where $\sigma = 1.1\,H$ to ensure particle overlap. Displayed in Fig. B.3 is the L_2-norm error of the velocity prediction as described in Chapter 3. The error of the integration along $r = const$ is also seen. In this case the singular position $r = a$ is avoided by only considering positions $0.25\,a < r < 0.9\,a$. It is seen that for essentially all cases the error logarithm scales exactly as predicted by the MLMIC method with polynomial order P. The near field evaluations of the VPM are hence accurately being captured and the far field singular particle approximation inherent to the method employed here appears to be working satisfactorily.

Accuracy of GML method: regularised vortex ring Although the previous sections demonstrate the accuracy for a singular vortex filament, in practice regularized vortex filaments must be applied in order to avoid singularities in the numerical solution. A logical progression from the

previous case is hence to inspect the accuracy of the solution when a regularisation is being applied. An ideal comparison case here is the vortex ring, as described more fully in Appendix E. As described there, for a Gaussian core distribution (most commonly applied in the work here) β takes the value $\frac{1}{2}\log 2 + 1 - \frac{\gamma}{2}$ [22], this is used here to validate the `GML` method for regularized filaments. The particle set is generated as particles along the vortex ring circumference with vorticity tangent to the ring and proportional to the angular segment and filament core parameter a. In

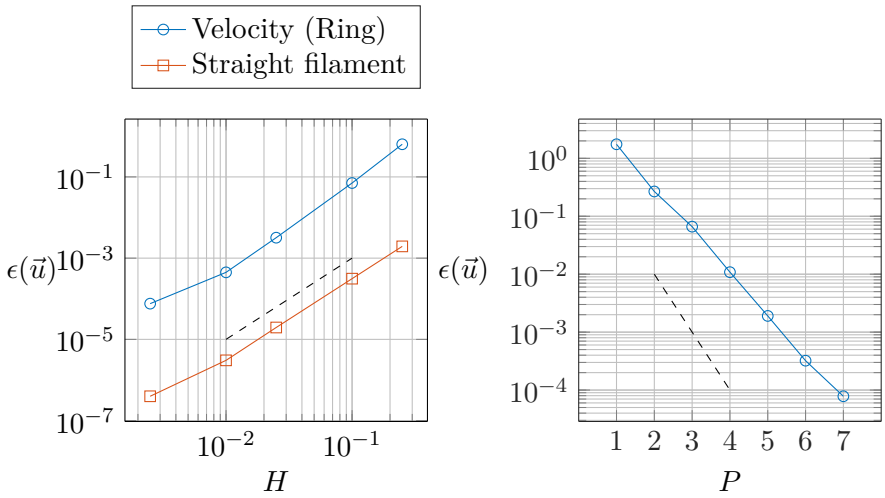

Figure B.4: Error of VP treatment of a regularised vortex ring as compared to the analytical value of translational velocity. Left: Accuracy as a function of grid size H. Right: Accuracy of `GML` solver as a function of polynomial order P.

Fig. B.4 the error as a function of grid size H is shown and as expected behaves as $\mathcal{O}(H_c^2)$. For purposes of comparison the error of prediction of a single vortex filament is also shown. Also displayed is the error scaling of the regularized ring as a function of polynomial order P. The grid size has been chosen here as $H = 0.001$, which discretises the ring into approximately 600 elements. Here the behaviour is practically the same as for the regularised tests, demonstrating that the `GML` method is applicable to regularised vortex filaments. In Appendix D an unsteady vortex filament case will be inspected where the accuracy of the stretching term is validated.

Appendix C Validation: Hill's Spherical Vortex

An ideal case for validation of the flow solver under steady conditions is Hill's spherical vortex. This represents a flow for which a sphere is a stream surface. Originally described by Hill in 1884 [156], the case offers a unique opportunity to inspect the accuracy of the solvers as it represents one of the few cases where analytical solutions are known for all of the important flow quantities: Vorticity $\vec{\omega}$, stream function $\vec{\psi}$ and velocity $\vec{u}$. A plot of the streamlines is given in Fig. C.1

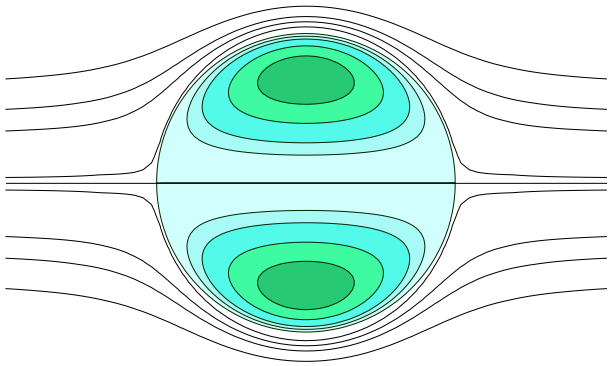

Figure C.1: Streamlines around a Hill's spherical vortex.

Description of flow The flow is described in a polar coordinate system (r, ϕ, z). The vortex translates along the axial coordinate z with translation velocity U_0. Vorticity is confined within a sphere of radius a aligned purely in azimuthal direction $\vec{\omega} = \omega \vec{e}_\phi$ with magnitude:

$$\omega = \begin{cases} -\frac{15}{2}\frac{U_0 r}{a^2} & r \leq a \\ 0 & r > a \end{cases} . \tag{C.1}$$

It is clear that the flow outside of the sphere is is irrotational and hence a potential flow. The surfaces of $\omega = const.$ are cylinders around the axis of translation.

Comparison of stream function It appears a logical choice to initially check the ability of the solver to calculate the stream function. The `GML`

solver can be utilised to directly output the stream function using a Green's function expression, as is done with the BS kernel. The stream function furthermore is a direct output of the FP routine when using the `PML` solver. The analytical value of the Stokes stream function Ψ is given as:

$$\Psi = \begin{cases} -\frac{3}{4}U_0 r^2 \left[1 - \frac{r^2+z^2}{a^2}\right] & r \leq a \\ -\frac{1}{2}U_0 r^2 \left[1 - \frac{a^3}{(r^2+z^2)^{3/2}}\right] & r > a \end{cases}. \tag{C.2}$$

It should be noted that the Stokes stream function is formulated for axysymmetric incompressible flows of an inviscid fluid and for a cylindrical coordinate system $r\vec{\psi} = \Psi\vec{e}_\phi$. Similar to the stream function, values of $\Psi = const$ enclose a streamtube, everywhere tangential to the flow velocity vector. Both solvers are configured for Cartesian coordinate systems. For comparison the analytical solution to Ψ given above is converted to the equivalent Cartesian solution. The relative L_2-norm error of the stream function $\epsilon(\vec{\psi})$ is shown Fig. C.2 for both the `GML` and `PML` solvers. For comparison two Gaussian

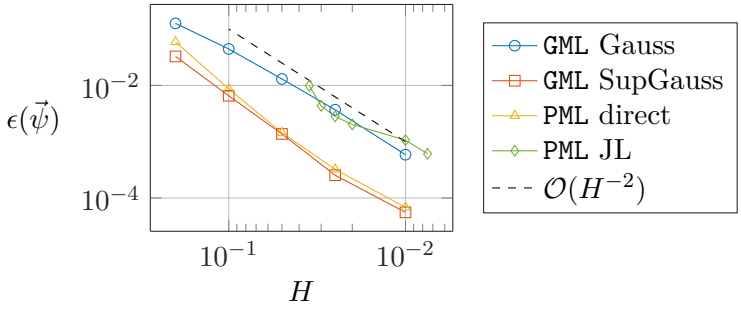

Figure C.2: Relative error in the calculation of the stream function as a function of grid resolution.

kernels have been investigated when using the `GML` solver, the standard Gaussian and the super Gaussian kernel, corresponding to the case $m = 2$ in Hejlesen [157] and to the super Gaussian in Winckelmans & Leonard [23]. The error reduction is seen scale as $\mathcal{O}(H^{-2})$ for all solvers. The accuracy of the Super Gaussian is also as expected almost an order or magnitude lower than the standard Gaussian. The Hejlesen-style spectral kernel [91] was also tested, however this was found to have no better accuracy than the Super Gaussian, furthermore the computational expense of evaluating the sine integral was significantly higher than that for the Gaussian regularisations, this kernel has application in e.g. Lattice Green's functions representations for FP solver– see Chapter 4.

In addition, two forms of the `PML` solver have been investigated as described in Chapter 4. The first makes use of the James-Lackner (JL) algorithm for the

boundary evaluations, the second makes use of a direct boundary condition using the SF kernel (PML direct). This was carried out for verification of the PML JL BC calculation. Two points are observed: Both solvers scale as $\mathcal{O}(H^2)$ as expected, and the boundary evaluation with the JL algorithm incurs an error most likely due to the approximation of the boundary with the singular SF kernel. For the PML -JL configuration some grid sizes were impractical due to overlap requirements and hence a narrower band of grid sizes could be investigated. The results generally are as expected and the convergence with H is observed.

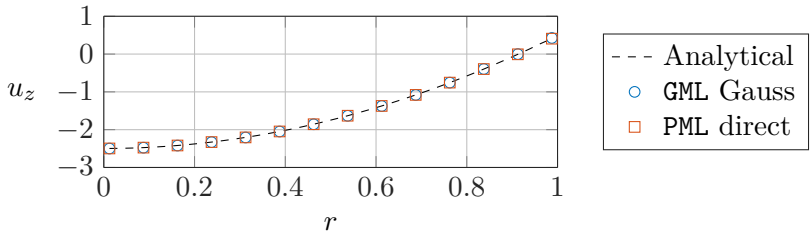

Figure C.3: Axial velocity as a function of radial position.

Comparison of velocity The analytical value for velocity is given as:

$$u_z = -\frac{3}{2}U_0\left(1 - \frac{2\rho^2 + z^2}{a^2}\right) \quad , \qquad u_\rho = -\frac{3}{2}\frac{z\rho}{a^2}\,. \tag{C.3}$$

This again must be converted to the Cartesian frame of reference for comparison with the solvers. This expression along with the predictions made by the solver is displayed in Fig. C.3. The GML solver in this case calculates the velocity based on the expression for the BS Kernel. The PML solver calculates this using FD as described in Chapter 4.

Two solvers have been plotted in Fig. C.3, the GML and PML -direct solvers as described above. The results for GML -SupGauss and PML -JL were visually identical and were hence not included. The relative L_2-norm error of the z-velocity $\epsilon(\vec{u})$ is shown Fig. C.4 for the considered solvers. Here essentially similar behaviour is seen, as the prediction for the stream function. The $\mathcal{O}(H^{-2})$ error is again seen and the error induced by the FD calculation of the velocity is predicted as expected. As a second-order method FD scheme is used, a $\mathcal{O}(H^{-2})$ scaling of error is predicted and this is indeed observed.

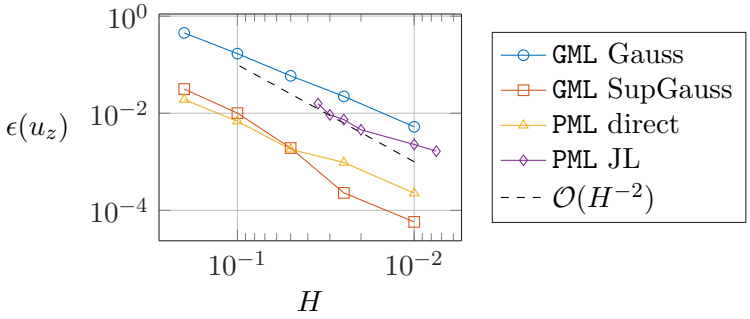

Figure C.4: L_2-norm error in the calculation of the axial velocity as a function of grid resolution.

Appendix D Validation: Vortex Ring Collision

An ideal test case to validate vortex stretching is the collision of two inviscid vortex rings. Here two vortex rings are aligned along the same axis of symmetry and collide head-on. The symmetry of the problem implies that the problem is identical to the case of a vortex colliding against an inviscid wall, as illustrated in Fig. D.1. The theory of the inviscid vortex ring of Helmholtz and Kelvin as described in Section E shall be exploited. This test demonstrates that the process of vortex stretching is proceeding correctly for both filament or *sparse* particles and for dense particle representations. This tests acts as a precursor to a viscous tests where both stretching and diffusion terms act together.

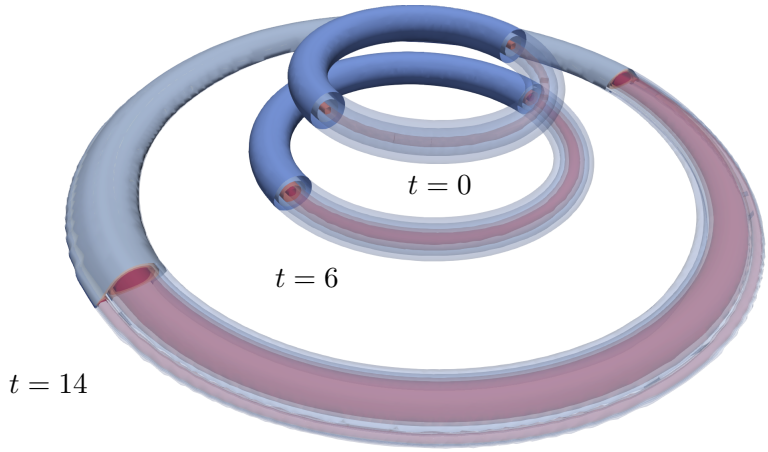

Figure D.1: The impingement of an inviscid vortex ring against a slip wall. The time is normalized with respect to $T_0 = R_0^2/\Gamma_0$. Here $R_0 = 1\,\mathrm{m}$ and $\Gamma_0 = 10\,\mathrm{m}^2\mathrm{s}^{-1}$.

Vortex filament In the case that a single vortex filament is used, the particle are aligned along the curve $R = 1$ and the particle strength and volumes are appropriately defined such that the circulation of the equivalent infinitely thin vortex filament is specified. The efficacy of the `GML` solver is demonstrated here, as the `PML` solver is unsuited to the treatment of singular filaments as the finite difference process of the calculation requires continuous field quantities. As described in Walker [158], a solution to this system is found by considering that each vortex ring is influenced only

by itself u_s and the mirrored ring u_m. The ring does not cause itself to expand, hence $u_{r,s} = 0$ and the self-induced axial velocity $u_{z,s}$ is given by Eq. (E.3). The influence of the mirrored ring can be derived from Eq. (E.2)– see Appendix E.

$$\frac{dz}{dt} = u_{z,s} + \frac{\bar{k}}{4\pi z}[2F(\bar{k}) - E(\bar{k})]\,, \quad \frac{dr}{dt} = \frac{\bar{k}^3}{4\pi r} D(\bar{k})\,, \tag{D.1}$$

where $k^2 F(k) = E(k) - (1-k^2)K(k)$ and $k^2 D = K(k) - E(k)$. When appended with the additional constraint that $ra^2 = const$, the differential equations above can be numerically integrated to give the time evolution of ring radius $r(t)$ and distance $z(t)$.

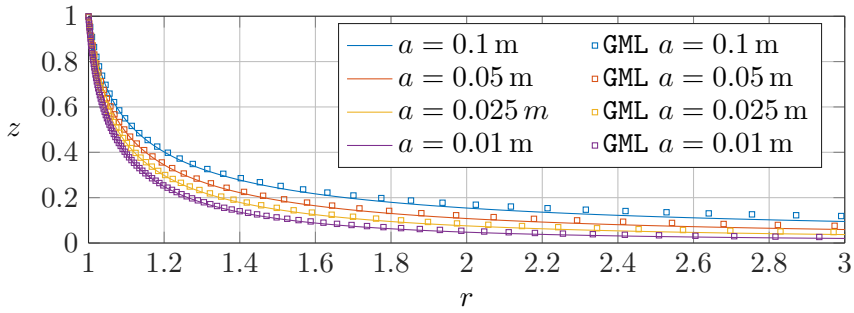

Figure D.2: Motion of a colliding vortex ring pair. Lines indicate the numerically integrated analytical solution as given in Eq. (D.1) and those predicted by the GML solver.

This was carried out here with a simple forward Eulerian scheme and the results for a range of vortex core sizes a is shown in Fig. D.2. Also visible in this plot are the results of the GML solver with the assumption of a Gaussian vortex smoothing and using the transpose stretching scheme. It is seen that the prediction improves as $a \to 0$. This is to be expected as the analytical expression are valid for an infinitely thin vortex ring. This implies an accurate resolution of the stretching term of the GML solver, as the vortex position and strength are coupled.

Dense vortex ring In order to test both the GML and PML solvers, the vorticity field is now described with a continuous field using the distribution as described in Eg. (5.8). Visualisations are identical for the GML model. It should be noted that in reality two vortex rings have been created in order to simulate the effect of a slip wall by placing a second vortex mirrored about the plane $z = 0$.

Diagnostics In order to demonstrate that flow diagnostics are being conserved these have been monitored for both the `GML` solver and the `PML` solver using the James algorithm for calculation of boundary terms. The results are shown in Fig. D.3.

Circulation Γ For every particle in the vortex ring, there is a second particle at the azimuthally opposite position with exactly the opposite circulation, hence by construction the system has net zero circulation $\Gamma = \left\|\vec{\Gamma}\right\|$. This is observed at the beginning of the beginning of the simulation and it seen to be conserved during the process of ring collision and expansion. This demonstrates that for the `GML` method the stretching scheme is effective, and for the `PML` solver the FD scheme for the stretching terms is working effectively. The expansion of the ring is seen to occur and the reduction in radial velocity is observed in Fig. D.2. It should be noted here that remeshing and divergence filtering is being applied to the field every 10 time steps, so the results here further demonstrate the these filtering procedures are operating effectively.

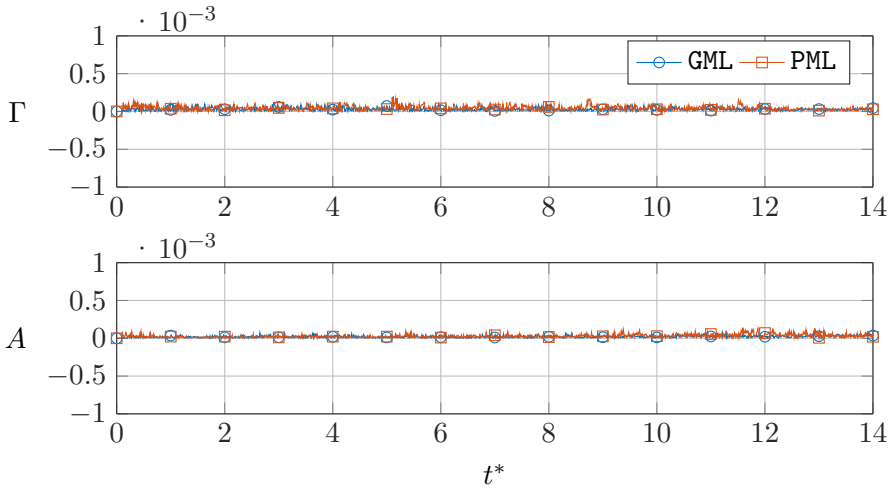

Figure D.3: Conservation of flow invariants during the collision process.

Angular Impulse A The net angular impulse $A = \left\|\vec{A}\right\|$ given explicitly by Eq. (5.3) should also by argument of symmetry vanish for this problem, as seen at the beginning of the simulation. This quantity should also be conserved, which is seen to occur through the expansion of the ring. This suggests, in addition to the conservation of circulation above, that particle convection is correctly being carried out for both solvers.

When compared with the plots in Chapter 5, one sees what appears to be unsteadiness in the flow invariants. This results from the use of single precision floating point variables in the tests here rather than double precision as used in Chapter 5. This inevitably gives rise to lower absolute accuracy.

Appendix E Theory of the Vortex Ring

The validation cases described in Chapter 5 are based upon vortex rings. For the avid reader of fluid dynamics the vortex ring presents a highly insightful yet not overly complex flow case which permits investigation with numerous techniques. The theory is abundant with analytical solutions which are not only mathematically elegant, but which furthermore illustrate important physical processes which occur in a vortex-driven flow. The theory has a rich history spanning back almost two centuries and almost every fluid dynamicist of notable achievement (including Helmholtz [84, 159], Kelvin [160], and Lamb [1]) has left their mark on the field in some form. Furthermore, the phenomena has not been unnoticed by the general public. An enduring example was the *camel man* of Times Square in New York– see Fig. E.1. A large billboard advertising camel cigarettes was constructed which periodically generated a vortex ring imitating a joyous smoking gentleman.

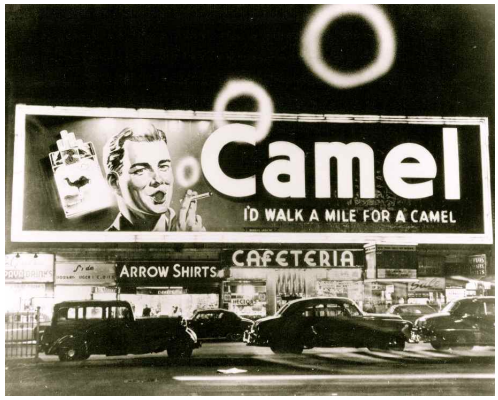

Figure E.1: The camel man, mounted at Times Square from the beginning of the 1940's until 1966. In this time he puffed away approximately 200 million rings. Image taken from [161].

The theory of the vortex ring is described here initially for inviscid rings, the effect of including viscosity terms is then described. Following this the instability of a vortex ring is described for reference to the unsteady turbulent cases investigated.

Inviscid Vortex Ring

The theory is greatly simplified by assuming the fluid to be inviscid, this furthermore allows a number of important results to be derived.

Stream function representation Assuming the vortex ring to be composed of a single, thin vortex filament of radius R and cross-sectional circulation Γ, Helmholtz was able to express the flow with the use of the Stokes stream function Ψ [84]. This is valid for an axysymmetric flow $\vec{u} = u_r\vec{e}_r + u_z\vec{e}_z$ and allows reduction of the problem to a single potential Ψ:

$$u_r = -\frac{1}{r}\frac{\partial\Psi}{\partial z}\,, \quad u_z = \frac{1}{r}\frac{\partial\Psi}{\partial\rho}\,. \tag{E.1}$$

By exploiting the symmetry of the problem Helmholtz was able to express the stream function at a radial position r as:

$$\Psi(r,z,t) = \frac{\sqrt{Rr}}{2\pi k}[(2-k^2)K(k) - 2E(k)]\ , \quad k^2 = \frac{4Rr}{z^2+(r+R)^2}\,, \tag{E.2}$$

where K and E are the complete first and second elliptical integrals, respectively.

Translational velocity of a vortex ring Each element of the vortex filament induces a velocity on the remaining segments of the ring as described by the Biot-Savart law [1]. Due to the axysymmetric geometry of the ring, all components of induced velocity not aligned with the axis of symmetry z of the vortex cancel and there remains only an induced velocity in the axial direction $U = u_z$ which causes the vortex to translate under its own self-influence. In a translation of Helmholtz' seminal paper, Lord Kelvin communicated the solution to the translation velocity of the thin vortex ring of core size a as [159]:

$$U = \frac{\Gamma}{4\pi R}\left\{\log\frac{8R}{a} - \beta + \mathcal{O}\left(\frac{a}{R}\right)\right\}\,. \tag{E.3}$$

The parameter β depends on the type of core description used. The expression due to Kelvin himself assumed a Rankine-type core-distribution (uniform vorticity) which gives $\beta = \frac{1}{4}$. The general case however allows determination of β based upon the core description used [22]. In the case that a Gaussian vorticity distribution is used, β takes the value $\frac{1}{2}\log 2 + 1 - \frac{\gamma}{2}$ where γ is the Euler-Mascheroni constant.

Expanding vortex rings Thus far only the case of a vortex ring with constant radius has been discussed. The behaviour of the vortex ring in the case that the radius is somehow changed due to an external field. Two cases of interest present themselves:

A vortex ring approaching an impermeable wall In this case the ring, moving under its own self influence (velocity U) approaches a wall at a distance z_w. An accurate treatment of the effects at the wall would account for the no-slip condition there and the therewith evoked shear layer in the limit as $z_w \to 0$. In order to remain within the scope of inviscid flows however this shall be avoided. A kinematically equivalent case applying the method of images is to mirror the vortex ring about the plane $z_w = 0$ [162]. The two rings approach each other along the same axis of symmetry with separation distance $2z_w$. The self-induced velocity U of each vortex as given by Eq. (E.3) causes the rings to approach each other.

The effect of the each ring is to induce a dilation in the opposing ring. This effect magnifies as the distance between the rings decreases. The linear momentum in axial direction is converted to the radial direction as the two rings expand outwards, their linear momentum (outward expansion) decreasing with increasing radius– see Fig. D.1. The vorticity distribution being axysymmetric, conservation of vorticity implies that the circulation Γ of the ring remains constant and hence the vortex core must correspondingly decrease as ring radius R increases and geometrically one observes that $Ra^2 = const.$ Neglecting higher order processes, eventually the vortex ring becomes so large that the radial velocity asymptotes to zero, as does the axial velocity. The process is now inspected with the help of the Helmholtz's expression- Eq. (E.2). By superimposing the second mirrored ring, one arrives at an expression for the total stream function: $\Psi_{tot} = \Psi(r, z_+, t) + \Psi(r, z_-, t)$. This is illustrated in Fig. E.2.

Leapfrogging vortices In this case two (or potentially more) vortex rings are again aligned along the same axis of symmetry. They are initially within the same plane, have however distinct radii such that $R_1 < R_2$. The velocity field induced by both rings causes the inner ring to accelerate and the outer ring to decelerate, inducing a displacement between the two rings. The velocity field now causes the inner ring to expand and the outer to contract, both proportionally to the difference in radii $R_2 - R_1$. This process proceed, both vortices convecting, until the radii are equivalent and the inverse process occurs, where the (initially) outer ring is *pulled* inside the (initially) inner ring. The process continues and the inner and outer ring,

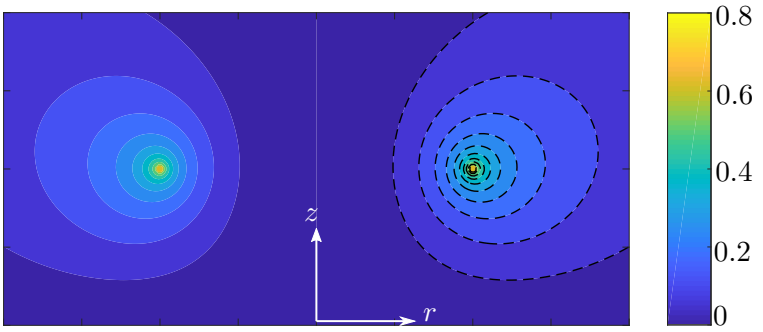

Figure E.2: Contours of the Stokes stream function Ψ of a thin vortex ring. The ring is mirrored about the plane $z = 0$, explaining the deflection of the stream contours upwards and outwards. The contour lines shown dashed are those predicted by the `GML` method under the assumption of a singular smoothing function, corresponding to the infinitely thin filament in the analytical theory. The agreement is seen to be perfect.

continuously convecting, *leapfrog* through each other, visualised in Fig. E.3. The calculation of the motion occurs in a similar manner to that above for the colliding vortices however the inner $[r_i, z_i]$ and outer $[r_o, z_o]$ velocity contributions are added $\Psi_{tot} = \Psi(r_i, z_i, t) + \Psi(r_o, z_o, t)$. Accounting for the change in vortex cross section for each ring under expansion and contraction $Ra^2 = const$ is again crucial for accurate results.

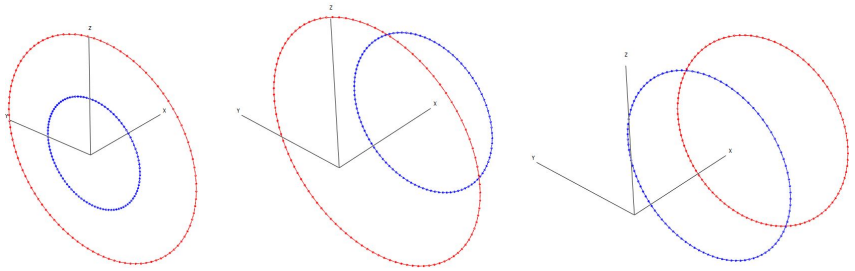

Figure E.3: Two initially parallel vortex rings undergoing leapfrogging.

Viscous Vortex Ring

The results were extended to the viscous case in the classical work by Saffman [121]. There Saffman observed that the flow is unsteady in any reference frame, hence making a specification of global velocity for U ambiguous. He

introduced a new quantity to represent the vortical centroid of impulse elements:

$$\vec{X} = \iint_S \frac{(\vec{x} \times \vec{\omega}) \cdot \vec{I}}{I^2} \vec{x} \, dV \tag{E.4}$$

where $\vec{I}$ is the linear impulse (5.2), $\vec{x}$ and $\vec{\omega}$ the position and vorticity vectors, respectively. For many simple geometries, this definition simplifies to the classical vorticity centroid, however the additional specification enabled Saffman to apply earlier results to the viscous case. Using this he arrived at the following expression:

$$U = \frac{dX_1}{dt} = \frac{\Gamma}{4\pi R} \left\{ \log \frac{8R}{\sqrt{4\nu t}} - 0.558 + \mathcal{O}\left[\left(\frac{\nu t}{R^2} \right)^{\frac{1}{2}} \log \left(\frac{\nu t}{R^2} \right) \right] \right\} \tag{E.5}$$

One observes that the core size a is replaced by a viscous quantity. This expression is valid in the limit $\frac{\nu t}{R^2} = S \to 0$. The effect of the viscosity is hence to slow down the ring like $-\log \nu t$. This expression assumes that the vorticity is, to first order, Gaussian in the limit of a small core. The assumption of a locally Gaussian vortex is influenced by the curvature of the vortex ring. The validity of this expression was numerically investigated in the work of Stanaway et al. [122], where a dedicated spectral solver based on the axisymmetric Navier-Stokes equation was applied to investigate vortex ring geometries. There the error estimate due to Saffman was found to be conservative and was improved to $\mathcal{O}(S \log S^{1/2})$.

Asymptotically large time As described in Stanaway [122], a bubble of vorticity surrounds the vortex core and travels with it. Viscous diffusion entrains irrotational fluid into the bubble as the vortex ring translates causes it to slow down as the ring grows. The ring asymptotically comes to rest as vorticity spreads to the far field with viscous length scale $\sqrt{\nu t}$. Eventually the ring takes on the velocity profile of a Stokes vortex ring and drafts with characteristic time $t^{-3/2}$ [163].

Stability of a Vortex Ring

The topic of the stability of a vortex ring has attracted much attention due a number of fascinating experimental investigations into the topic. Experimentally the generation of such a ring is accomplished simply by impulsively pushing fluid through a circular orifice. This was carried out by Krutsch [164] and tracer dye was injected in order to visualise the core of the vortex ring. Here it was demonstrated that frequently instabilities

occur which are manifested as azimuthal perturbations of the vortex core. These have wavenumber of k– see Fig. 5.7. Similar experimental results were reported in Windnall & Sullivan [125], where a heuristic inviscid model was also proposed. In Widnall & Tsai [79] a convincing explanation is given based on a rectilinear vortex filament. This was confirmed in greater detail in a follow-up paper [165], where a rigorous mathematical basis allows for general calculation of instability growth rates for a given k.

As described in Saffman [126], the underlying concept is that a rectilinear vortex filament, deformed into a sinusoid with wavelength $2\pi/k$ is under self-influence steady. When the filament however is placed into a plane strain field parallel to the axis of the undisturbed vortex, the sinusoidal peaks convect faster than the undisturbed vortex centreline and hence the wave grows. The velocity field induced by a vortex ring (in an azimuthal cut) appears as a plane strain field. The vortex core, small compared to the ring radius R hence locally behaves as a rectilinear vortex. It was seen that the wave number appeared to be coupled to the Re of the ring. An explanation for this was proposed by Saffman [126] who furthermore showed how the wave number is sensitive to vorticity distribution inside the vortex core. There quantitative estimates are given which allow for the prediction of the most unstable wavenumber for a representative Reynolds number which includes a curvature.induced strain rate. This was investigated numerically by Shariff et al. [128] which demonstrated the validity of the theory outlined in Saffman.

Appendix F VPM Expressions

Introducing a core size σ allows specification of a smoothing function for the vorticity field $\zeta(\rho)$ where $\rho = r/\sigma$ is the normalised distance from the particle. This smoothing influences the stream function $g(\rho)$ (2.15) and velocity $q(\rho)$ (2.16) field induced by each particle. If PSE is used to calculate viscous diffusion, the integral Laplace operator $\eta(\rho)$ is also calculated from this.

Regularisation Options

Numerous options for $\zeta(\rho)$ exist depending on application case. Four options are given in Table F.1. These are the low-order algebraic (LOA), high-order algebraic (HOA) [22], 1st order and 2nd order Gaussian [157]. Numerous further options are available for e.g. 2D smoothings [23] and higher-order Gaussian [157] and spectral smoothings, the latter being suitable for lattice-Greens functions for FP solvers [166]. These have been omitted for brevity. For the work presented here the Gaussian kernel has been applied. For Gaussian expressions $\mathrm{E} = (2/\pi)^{1/2}\exp\{-\rho^2/2\}$ and $\mathrm{F} = \mathrm{erf}\{\rho^2/2^{1/2}\}$, where erf is the error function, defined as $\mathrm{erf}(x) = \int_0^x \exp\{-w^2\}dw$ [86].

Table F.1: Regularisation functions for the field induced by a vortex particle.

Reg.	$4\pi\zeta(\rho)$	$4\pi g(\rho)$	$4\pi q(\rho)$	$4\pi\eta(\rho)$
LOA	$\dfrac{3}{(\rho^2+1)^{5/2}}$	$\dfrac{1}{(\rho^2+1)^{1/2}}$	$\dfrac{\rho^3}{(\rho^2+1)^{3/2}}$	$\dfrac{15}{(\rho^2+1)^{7/2}}$
HOA	$\dfrac{15/2}{(\rho^2+1)^{7/2}}$	$\dfrac{\rho^2+3/2}{(\rho^2+1)^{3/2}}$	$\rho^3\dfrac{\rho^2+5/2}{(\rho^2+1)^{5/2}}$	$\dfrac{105/2}{(\rho^2+1)^{9/2}}$
Gauss (1)	E	$\dfrac{1}{\rho}\mathrm{F}$	$\mathrm{F}-\rho\mathrm{E}$	E
Gauss (2)	$\left(\dfrac{5}{2}-\dfrac{\rho^2}{2}\right)\mathrm{E}$	$\dfrac{1}{\rho}\left(\mathrm{F}+\dfrac{1}{2}\rho\mathrm{E}\right)$	$\mathrm{F}-\left(1-\dfrac{\rho^2}{2}\right)\rho\mathrm{E}$	$\left(\dfrac{7}{2}-\dfrac{\rho^2}{2}\right)\mathrm{E}$

Expressions for the Stretching Term

Keeping in mind the expression for the rate of change of the particle circulation from Eq. (2.9): $(\vec{\omega}\cdot\nabla)\vec{u}$, one observes that the stretching is only nonzero in positions with nonzero vorticity. As such, the local vorticity (and therewith circulation) influences the stretching term. Adopting the notation that source particle quantities have the subscript $(\cdot)_q$ and local quantities have the subscript $(\cdot)_p$, the stretching terms as a function of regularisation function are given for each stretching scheme by:

$$\frac{d\vec{\alpha}_p}{dt} = \frac{d\vec{\omega}_p}{dt}dV_p = \begin{cases} -\dfrac{q(\rho)}{r^3}(\vec{\alpha}_p \times \vec{\alpha}_q)) + \dfrac{1}{\sigma^5}[(\vec{\alpha}_p \cdot \vec{r})(\vec{r} \times \vec{\alpha}_q)]S & \text{Classic} \\ \dfrac{q(\rho)}{r^3}(\vec{\alpha}_p \times \vec{\alpha}_q)) + \dfrac{1}{\sigma^5}[\vec{\alpha}_p \cdot (\vec{r} \times \vec{\alpha}_q)\vec{r}]S & \text{Transpose} \\ \dfrac{1}{2\sigma^5}[(\vec{\alpha}_p \cdot \vec{r})(\vec{r} \times \vec{\alpha}_q) + \vec{\alpha}_p \cdot (\vec{r} \times \vec{\alpha}_q)\vec{r}]S & \text{Mixed} \end{cases} \tag{F.1}$$

where $\vec{r} = \vec{x}_p - \vec{x}_q$ and $r = \|\vec{r}\|$. The expression S depends again upon the regularisation chosen and is given by:

$$S = -\frac{1}{\rho}\frac{d}{d\rho}\left(\frac{q(\rho)}{\rho^3}\right) = \frac{1}{\rho^2}\left(3\frac{q(\rho)}{\rho^3} - \zeta(\rho)\right), \tag{F.2}$$

where the latter expression allows for straightforward implementation. The factor dV_p is the volume of the particle, which in practice has a finite value. It is seen that for evaluations of the velocity, stretching and viscous diffusion terms, numerous expressions repeatedly occur. This can be exploited to greatly reduce the number of floating point operations required for each interaction calculation.