

Time stepping methods for coupled fluid-rigid body simulation

by

Rikin Gurditta

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2026

© Rikin Gurditta 2026

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Interaction between fluids and solid objects is ubiquitous in everyday life, yet the resulting motion is too intricate for visual effects artists and animators to realistically depict by hand. Instead, artists turn to computer graphics applications that employ physics-based animation to simulate these complex phenomena. Some of these applications solve the *incompressible Euler equations* coupled with the *rigid body equations* to compute the motion of an incompressible fluid interacting with undeformable solids. Of particular interest is *two-way coupling*, in which the fluid and solids both affect each other's motion.

Many methods have been developed to improve the realism of fluid simulations, allowing them to simulate more compelling scenarios. There are several *time stepping schemes* for fluid simulation in the literature, presenting ways to evolve the motion of the fluid over time that may generate more energetic or more accurate results. In particular, we focus on the *BDF2* and *Advection-Reflection* families of schemes due to their accuracy and their improved ability to preserve the kinetic energy of the fluid. Our goal in this thesis is to extend these time stepping schemes to two-way coupled fluid-rigid body simulation, to yield more compelling simulations of the interactions between these two types of materials.

We catalogue some of the popular time stepping schemes for fluid simulation, and explain their relations to methods of solving ordinary differential equations. Then, taking as our starting point the popular method of Batty et al. [6], we re-derive the time stepping scheme originally proposed for coupled systems, and derive new schemes for coupled systems corresponding to the previously discussed fluid schemes, along the way comparing to the coupled time stepping scheme proposed by Gibou and Min [19]. We measure the accuracy, energy-preservation, and computational cost properties of each scheme implemented within a 2D simulation, presenting quantitative and qualitative results. We hope our work encourages further investigation into the theoretical basis as well as the qualitative properties of coupled fluid-rigid body simulation.

Acknowledgements

To Professor Christopher Batty, thank you so much for your generous support. It means so much to me that you would offer me not only a position in your lab, but also guidance through the daunting research process, encyclopedic knowledge of an overwhelming field, understanding of my circumstances, kindness toward my struggles, respect for my personhood, and encouragement to achieve past what I thought I was capable of. I could not ask for a better supervisor and I hope to work with you more in the future.

To my readers, Professor Toshiya Hachisuka and Professor Justin Wan, thank you for your invaluable discussion and feedback on my work. I greatly appreciate the provision of your immense expertise to my research.

To my colleagues in Christopher's Computational Motion Group: Brooke, Clara, Daniel, Isaac, Joel, Leo, Liang, Nathan, Rayhaan, Ryusuke, Tina; thank you for being my fellow travellers in the world of computer graphics. Attending group meetings and discussing your brilliant work was as exciting as it was inspiring. To Clara and Brooke in particular, thank you for the extensive discussions, feedback, and camaraderie, without which I would have floundered haplessly. Thank you as well to my colleagues in the Computer Graphics Lab, especially Ben, Brian, Craig, Ege, Matthew, Saba, Silvia, Stephen, Yiwei, Zack, Zane for making the lab an inviting place.

To my friends, which I (so fortunately) have too many to list, thank you for being my fellow travellers in life. Thank you in particular to Justin, Jakob, and Clara for being part of my home.

To my parents Amar and Karishma and my brother Ishaan, thank you for your unyielding support and nurturing that forms the bedrock of everything I do, including and especially this thesis.

To my cat Shadow, I present no gratitude. Your begging for my attention was insatiable, and you went so far as to sabotage me to this end: standing on my computer keyboard to overwrite my code, interrupting my sleep with your desperate meows, keeping me busy cleaning after your hairballs, and more. You were a menace.



Dedication

To my cat, Shadow.

Table of Contents

Author's Declaration	ii
Abstract	iii
Acknowledgements	iv
Dedication	v
List of Figures	ix
List of Tables	xii
1 Introduction	1
2 Related work	3
2.1 Fluid time stepping	3
2.2 Fluid-solid coupling	5
3 Background	8
3.1 Equations of motion	8
3.1.1 Fluid equations	8
3.1.2 Rigid body equations	8
3.1.3 Fluid-rigid body coupling	9

3.2	Solving the coupled fluid-rigid body equations	10
3.2.1	Operator splitting	10
3.2.2	Variational formulation	11
3.2.3	Spatial discretization	13
3.3	Solving differential equations	17
3.3.1	Solving ODEs	17
3.3.2	Method of characteristics	19
3.3.3	Method of characteristics for fluids	21
4	Fluid temporal discretizations	23
4.1	Implicit Euler	23
4.2	BDF2-based methods	25
4.2.1	BDF2 as presented by Xiu and Karniadakis	25
4.2.2	Material acceleration	26
4.2.3	BDF2 as presented by Gibou and Min	29
4.3	Advection-Reflection methods	30
4.3.1	First-order Advection-Reflection	32
4.3.2	Reflection for energy preservation	33
5	Coupled temporal discretization	37
5.1	Coupled Implicit Euler	37
5.2	Coupled BDF2	39
5.2.1	Method of Gibou and Min	39
5.2.2	Our coupled BDF2 method	40
5.3	Coupled Implicit Midpoint	41

6	Results	42
6.1	Pressure projection convergence	43
6.1.1	Taylor-Green vortex	43
6.1.2	Circular domain	45
6.1.3	Ellipse in box	48
6.2	Time stepping convergence	52
6.2.1	Taylor-Green vortex	52
6.2.2	Rigid body force	53
6.2.3	Circle in tube	56
6.3	Energy preservation	59
6.3.1	Dual vortices	59
6.3.2	Ellipse in box	60
6.4	Computational cost	62
7	Conclusion	63
7.1	Limitations	63
7.2	Future work	64
	References	66
	APPENDICES	71
A	2D analogues of 3D operators	72
B	Supplementary video	74

List of Figures

3.1	A given vector field (left) can be decomposed as a sum of a divergence-free vector field (middle) and the gradient of a scalar function (right).	11
3.2	Placement of velocity (u, v) and pressure p samples in a single cell of a staggered grid.	14
3.3	Finite volume-inspired cell edge length computation.	15
4.1	Each point represents a velocity field in $\mathfrak{X}(\Omega)$, with the black line representing $\text{Project}(\mathfrak{X}(\Omega))$, the subspace of divergence-free velocity fields. The blue curves represent advection steps, the green arrows represent the application of $\frac{1}{2}\mathbf{f}^{n+1/2}$, and the red lines represent energy loss due to projection onto $\text{Project}(\mathfrak{X}(\Omega))$. <i>Left</i> : First-order Advection-Reflection applies the midpoint force at the midpoint, but also the end point. <i>Right</i> : Second-order Advection-Reflection applies the midpoint force only at the midpoint, and scales it correctly with reflection. (The red arrow represents $2(\mathbf{u}^{n+1/2} - \mathbf{u}_a^n)$.) This figure is adapted from figure 2 of “A Second-Order Advection-Reflection Solver” [32].	33
4.2	These diagrams are similar to those in figure 4.1, so once again red lines represent kinetic energy loss due to projection onto $\text{Project}(\mathfrak{X}(\Omega))$, the subspace of divergence-free velocity fields. <i>Left</i> : Standard advection-projection incurs a large amount of energy loss. <i>Middle</i> : Two advection-projection steps each advancing by $\frac{1}{2}\Delta t$ require twice as much computation. <i>Right</i> : Advection-Reflection has the same computational cost as two advection-projection steps, but incurs less energy loss. This figure is adapted from figure 2 of “An advection-reflection solver for detail-preserving fluid simulation” [46].	35

6.1	Visualization of Taylor-Green vortex. Coloured particles are passive markers for visualization.	44
6.2	L^1 (left) and L^∞ (right) error vs grid cell spacing Δx for a pressure projection of the Taylor-Green vortex, with a pressure field of $p = x^4 y^3$ added.	45
6.3	L^1 (left) and L^∞ (right) velocity error vs grid cell spacing Δx for a pressure projection of a circular flow in a circular domain, with a pressure field of $p = xy^3$ added.	46
6.4	x (left) and y (right) velocity error over the fluid domain for test from Batty [5] with a grid resolution of 128×128 , i.e. $\Delta x = 0.015625$. The points with highest error are along the boundary of the fluid.	48
6.5	Visualization of the simulation domain for our “ellipse in box” test cases. The ellipse is a rigid body with a prescribed velocity.	49
6.6	L^1 (left) and L^∞ (right) velocity error vs grid cell spacing Δx for a pressure projection with irregular static boundaries.	50
6.7	L^1 (left) and L^∞ (right) velocity error vs grid cell spacing Δx for a pressure projection with irregular moving boundaries.	51
6.8	L^1 (left) and L^∞ velocity error vs time step size Δt for the Taylor-Green vortex at time $T = 5$. Note that the Advection-Reflection data and second-order Advection Reflection data coincide, as well as the two BDF2 methods.	52
6.9	Rigid body velocity error at $T = 5$ for the rigid body component of each coupled time stepping method.	55
6.10	Positions of rigid bodies and marker particles at times $t = 0.01, 0.20, 0.40$ (left to right).	57
6.11	L^1 (left) and L^∞ (right) velocity error of circle-in-tube test case at time $T = 5$	57
6.12	Dual vortex evolution for each fluid-only time stepping scheme. Vortices separate for each scheme except Advection-Projection and Material Acceleration.	59
6.13	Total kinetic energy over time for ellipse spinning within square fluid domain.	60
6.14	Marker particle positions (left) and vorticity (right) of fluid at time $t = 0.30$ for each coupled time stepping method. Schemes other than Advection-Projection result in more intricate vortical motion in the fluid, observable as higher values of vorticity.	61

6.15 Running time of 50 time steps with each coupled scheme, for Taylor-Green vortex and four submerged boxes.	62
---	----

List of Tables

6.1	L^1 velocity error of Taylor-Green vortex at $T = 5$ for various time step sizes, for each fluid-only time stepping scheme.	53
6.2	L^∞ velocity error of Taylor-Green vortex at $T = 5$ for various time step sizes, for each fluid-only time stepping scheme.	53
6.3	Rigid body velocity error at $T = 30$ for the rigid body component of each coupled time stepping method.	56
6.4	L^1 velocity error of circle-in-tube test case at $T = 5$ for various time step sizes, for each coupled time stepping scheme.	58
6.5	L^∞ velocity error of circle-in-tube test case at $T = 5$ for various time step sizes, for each coupled time stepping scheme.	58

Chapter 1

Introduction

The ubiquity of fluid-solid interaction in everyday life makes it a highly desirable subject in visual effects—leaves blowing in the wind, boats sailing through water, and wisps of smoke being fanned away are just some visually stunning examples. However, the high complexity of these motions renders them nigh impossible for visual effects artists and animators to realistically capture by hand. Instead, computer graphics applications turn to physics-based animation of fluid-solid interactions, where the intricate motions are generated by simulation of underlying physical laws. This thesis discusses improvements to the visual fidelity and realism of such animations.

Computer graphics applications typically use the *incompressible Euler equations* to model the motion of a fluid in the absence of compression and viscosity. For many familiar fluids including smoke, fire, and water, this model can convincingly capture the visual behaviours of the fluid despite its idealizations.

Similarly, many solid objects can be idealized as *rigid bodies*, which do not undergo deformations of their geometry. A wide range of objects deform only imperceptibly; for example, a coffee mug may vibrate when tapped, but to the naked eye it looks like it is still. For such objects, we model motion with the *rigid body equations*.

To simulate fluid-solid interaction, we must first devise a strategy to couple the simulation. Coupling the incompressible Euler equations and the rigid body equations can be done in various manners: *One-way coupling* entails simulating the influence of one of them on the other, for example the effect of a solid object with prescribed motion on a simulated fluid. *Two-way coupling* instead entails allowing both the fluid and the solids to influence each other's motion. In this work we consider two-way coupling, as one-way

coupling is only realistic in a small number of scenarios, where the influence of one material on another is negligible. *Weak (partitioned) coupling* entails an independent simulation method of the fluid and the solids with interaction computed by sequential exchange of forces, while *strong (monolithic) coupling* entails simultaneous solution of the interaction forces between the fluid and solids. While weak coupling can be conceptually simpler, this work focuses on strong coupling as it can provide more accurate and stable solutions.

We take as our starting point the popular method of Batty et al. [6], who took a variational approach to solving a strongly two-way coupled system in space, reformulating coupling as an optimization problem at each instant in time. Batty et al. evolve the fluid component of their system over time using the *Implicit Euler* time stepping scheme introduced to fluid simulation for computer graphics by Stam [42], and evolve the solid component over time with a related implicit Euler scheme. Notably, this time stepping scheme takes an *operator splitting* [20] approach, which is ubiquitous in computer graphics.

Since Stam’s time stepping method was first presented, computer graphics researchers have explored many more splitting-based time stepping schemes for fluid simulation, achieving more accurate, energetic, and visually interesting results in the fluid interior, assuming static boundaries. These methods significantly reduce the numerical dissipation observed in Stam’s method, decreasing the “artificial viscosity” in the simulated fluid and allowing for vortices that persist for longer durations. There are two prominent families of such methods: BDF2 schemes [45, 19, 34] and Advection-Reflection schemes [46, 32]. In light of these improvements to fluid-only time stepping, our overall objective is to update the dominant coupling approach presented by Batty et al. to support schemes from each of the two families, in order to pair their visual improvements with accurate fluid-rigid body coupling.

The contributions of this thesis include:

- Deriving time stepping methods for coupled fluid-rigid body systems whose interactions are computed by the method of Batty et al.;
- Cataloguing and discussing various fluid time stepping methods from the literature;
- Comparing convergence and qualitative behaviour of the various schemes discussed.

Chapter 2

Related work

Fluid simulation for computer graphics is a thriving and mature field, with active development into several families of simulation techniques: smoothed-particle hydrodynamics (SPH) [43, 30], the lattice-Boltzmann method (LBM) [44], vorticity-based simulation [16], and more. In this thesis we focus specifically on velocity-based Eulerian grid simulation methods, the most popular of the various techniques. This approach has been well-documented and explicated in *Fluid Simulation for Computer Graphics* [8].

For this particular style of fluid simulation, there has been much work in developing methods for time stepping as well as methods for coupling with rigid bodies. We provide an outline of this related work here.

2.1 Fluid time stepping

The complexity and nonlinearity of the Navier-Stokes equations (and related PDEs such as the incompressible Euler equations) make them challenging to solve monolithically, i.e., it is difficult to formulate techniques to solve the PDE all together in one step. Instead, *operator splitting* is used to break the PDE into constituent parts, which can be solved more easily and whose solutions can be combined into a solution of the original PDE [20].

The now-standard operator splitting method for fluid simulation was first introduced to computer graphics by Stam [42]. Stam used the *method of characteristics* to rewrite the fluid’s governing partial differential equations (PDEs) as ordinary differential equations (ODEs) in the trajectory of each “particle” of fluid, and applied the implicit Euler ODE time discretization method to derive a time stepping scheme. This scheme involves solving

the fluid PDEs step-by-step: first *applying forces*, then *advecting* the fluid, then *diffusing* to incorporate viscosity, then *projecting* the velocity field onto the subspace of divergence-free flows. In this thesis we omit the diffusion step as we are focused on inviscid fluids.

Xiu and Karniadakis [45] improved this operator splitting scheme by providing a method based on the BDF2 discretization for ODEs. Their method makes use of the same fluid operators, or constitutive steps, as Stam’s to create a scheme that achieves second-order accuracy in time where Stam achieved first-order accuracy. Nielsen et al. [34] presented an updated BDF2-based scheme for their work on combustion simulation, called the *material acceleration* scheme. This scheme reduces the computational overhead of BDF2 by reformulating it to make use of an acceleration variable, which can be advected simultaneously with other quantities. The result is a BDF2 discretization that requires only one advection operation, where previously two were required.

Another extension of Stam’s operator splitting scheme is the *advection-reflection* scheme of Zehnder et al. [46]. The pressure projection operator used in previous splitting methods discards part of the velocity field, decreasing the kinetic energy of the fluid. To reduce energy loss, Zehnder et al. make use of an energy-neutral *reflection* step. Narain et al. [32] improved this time stepping algorithm, applying the method of characteristics as Stam did to consider the trajectory of a particle of fluid, and discretizing using the implicit midpoint ODE scheme. This resulted in yet another second-order accurate time stepping scheme which makes use of the same fluid operators as Stam’s.

Jiang et al. [25] introduced a new operator splitting scheme that numerically achieves second-order convergence in time. This scheme explicitly evolves pressure and pressure gradient variables in time, and uses the incremental pressure correction approach outlined by Guermond et al. [23]. However, this scheme was not theoretically proven to achieve second-order accuracy in every scenario.

While Nabizadeh et al. [31] make their main innovations in the advection step of the operator splitting framework, they also present an ostensibly second-order operator splitting scheme. Their method is similar to the implicit Euler time stepping scheme of Stam, albeit computing the advection step using a computed midpoint velocity, resulting in a more accurate estimation of the fluid’s trajectory.

Improvements to time stepping are generally formulated to improve a fluid simulation’s accuracy as well as its qualitative visual fidelity, which generally correlates to numerical dissipation of energy. In pursuit of the latter goal, Mullen et al. [29] abandoned operator splitting in favour of a monolithic time stepping scheme that preserves energy exactly. However, this unconventional approach has some undesirable aspects: it requires a root finding procedure to solve a nonlinear problem for each time step, as opposed to the linear

pressure projection used in operator splitting schemes. It is also a purely Eulerian scheme, making numerical diffusion unavoidable, while operator splitting schemes can be modified using methods such as FLIP [47] to significantly reduce diffusion. Lastly, the order of accuracy of the method was not investigated.

Our contribution in this work is to develop time discretizations for coupled systems of fluid and rigid bodies that are based on the two provably second-order operator splitting schemes discussed, BDF2 and second-order advection-reflection. Adherence to the ubiquitous operator splitting approach makes our work compatible with most existing technology in the field.

2.2 Fluid-solid coupling

Solid objects are typically simulated from a Lagrangian point of view, tracking the positions and velocities of defined material points. Furthermore, as rigid objects undergo no deformation, their state can be fully described by the position, rotation, velocity, and angular velocity about their centre of mass. In contrast, fluids are typically treated from an Eulerian point of view: evolving quantities are tracked at fixed positions in the domain, typically on points of a grid. A principal challenge in fluid-rigid body coupling is making these two representations compatible with each other.

Peskin [36] introduced to applied mathematics the highly influential *immersed boundary method*, which reformulates a Lagrangian view of an incompressible viscoelastic material into a system of Eulerian equations, allowing its dynamics to be solved in tandem with a fluid’s dynamics.

In computer graphics, fluid-solid coupling was first introduced by G enevaux et al. [17], concerning mass-spring solids. Their method is a weak coupling scheme, computing viscoelastic forces between the solids and their surrounding interfaces. The interfaces themselves are tracked using marker particles, so that a Lagrangian viewpoint can be taken when computing forces. The total force can be applied to the fluid by interpolating it back to the Eulerian grid. This method is not applicable to rigid solids, as its mass-springs model of solids would require prohibitive stiffness to represent a rigid object. A weak coupling approach was also taken by Guendelman et al. [21] to couple thin shells (both deformable and rigid) to fluids, this time using an intermediate pressure projection step to compute fluid forces on the solid, before taking a last pressure projection step to enforce solid boundary conditions on the fluid.

Carlson et al. [11] take an approach related to that of Peskin, formulating a single

Eulerian discretization of both the fluid and the solids in the scene. In their approach, one field is used to describe the velocity of all material in the scene, whether belonging to the fluid or a rigid body. This velocity field is then made to satisfy constraints of incompressibility of all material, and rigidity within the solid objects. The latter step, modifying the velocity field to be a rigid motion within each solid, leads to a drawback of this method: compatibility between the solid and fluid motion at the interface is not preserved, potentially leading to interpenetration of the fluid into the solid, or compression of the fluid.

An early example of strong fluid-rigid body coupling was presented by Klingner et al. [27]. They took an unconventional approach, representing the fluid domain using a tetrahedral mesh, and remeshing at each step to reflect the changing shape of the domain as rigid bodies move within the fluid. This is advantageous as it allows the fluid mesh to conform closely to the shape of the solid boundaries at each time step. Interaction between the fluid and rigid bodies was computed during the pressure projection step, by finding an appropriate pressure to make the fluid incompressible and to exert forces on the solids.

Bridson and Müller-Fischer [10] presented a variational formulation of the pressure solve for a fluid domain with static boundaries. In other words, they reformulated the pressure PDE into an equivalent optimization problem, simplifying the treatment of irregular boundary conditions. Batty et al. [6] extended this formulation to free-moving rigid solids, resulting in a variational method for not only fluid simulation but also strongly coupled simulation with fluid-solid interaction. This method relies on estimating the volume of the intersection of a grid cell and a solid, to compute a pressure field that applies forces to the surfaces of rigid solids. However, Batty et al. do not provide a derivation, so we present a full derivation of the variational approach to coupling in section 3.2.2. This method is similar to that of Klingner et al., as it implements coupling by enforcing compatibility between the fluid and solid motion through the computation of an appropriate pressure.

Roble et al. [39] used the divergence theorem to derive a simple treatment of a pressure solve for a fluid domain with irregular static boundaries discretized on a static Cartesian grid. They introduced to graphics a discretization that takes into account the *areas of grid faces* that are partially covered by solids, taking a finite volume approach, as opposed to the previously described methods which rely on estimating the *volumes of grid cells* containing solids. Ng et al. [33] extended this finite volume approach to strongly coupled simulations. Like that of Batty et al., their method computes a pressure field that induces compatible forces on the solids. However, it incorporates a more accurate spatial discretization of irregular solid boundaries onto the fluid grid faces, making use of the method of Min and Gibou [28] to estimate integrals over the resulting partial grid faces and partial solid faces.

This allowed their pressure projection to achieve second-order convergence in space, where that of Batty et al. only achieved first-order convergence for pressure.

The work of Gibou and Min [19] presented a new model of the extent of the fluid domain using a Heaviside function, i.e., a function which takes value 1 on the fluid domain and 0 on solids. It also presented a novel BDF2-based time discretization for the coupled system. This discretization, when applied to a fluid system with static boundaries and no free-moving rigid bodies, does not reduce to the BDF2 methods of Xiu and Karniadakis or Nielsen et al. discussed earlier. The difference is in the treatment of pressure: the latter methods treat pressure implicitly, involving it only in the final projection step of the algorithm (non-incremental pressure correction), while the former treats it explicitly, computing intermediate velocities using a pressure variable (incremental pressure correction). These approaches are respectively outlined in sections 3.1 and 3.2 of the work of Guermond et al. [23], where it is also explained that the incremental pressure correction approach is theoretically more accurate than the non-incremental approach when paired with an implicit Euler time discretization, and when periodic boundary conditions are used. However, it is unclear if this improvement persists for BDF2 schemes or for solid boundaries/coupling, and Gibou and Min do not evaluate their choice of the incremental pressure correction approach.

In this thesis we adopt the method of Batty et al. to implement strong rigid body coupling via the pressure projection step. This approach, as opposed to that of Klingner et al., allows us to take the ubiquitous approach of discretizing our simulation domain on a static Cartesian grid. Furthermore, we make partial use of the improved spatial discretizations used by Ng et al., estimating fluxes across partially-occluded grid faces rather than estimating fluid volumes surrounding grid points as Batty et al. do. We do not make use of further improvements to spatial discretization, such as the quadratic interpolants and PDE-based extrapolation methods used by Ng et al., and the Heaviside function-based discretization of Gibou and Min. However, these improvements are not incompatible with our work due to the operator splitting framework they share with the work of Batty et al.

Chapter 3

Background

3.1 Equations of motion

3.1.1 Fluid equations

In graphics, fluid flow is typically modeled by the *incompressible Euler equations* [8]:

$$\begin{aligned}\rho \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \right) &= -\nabla p + \mathbf{f} \\ \nabla \cdot \mathbf{u} &= 0\end{aligned}$$

where $\mathbf{u} \in \mathfrak{X}(\Omega)$ is the velocity field, ρ is the density of the fluid, p is the pressure field, a scalar field over Ω , and $\mathbf{f}$ is the external force on the fluid (typically gravity). We use $\mathfrak{X}(\Omega)$ to denote the vector space of vector fields over Ω .

3.1.2 Rigid body equations

A rigid transformation $\mathbf{T} = (\mathbf{X}, \mathbf{R})$ consists of a translation $\mathbf{X}$ and a rotation $\mathbf{R}$, and we use $\mathbf{T}(\mathbf{x}) = \mathbf{R}\mathbf{x} + \mathbf{X}$ to denote the transformation applied to $\mathbf{x}$. Since a rigid body does not deform, its generalized position can be described by a rigid transformation. For simplicity, we also assume that $\mathbf{X}$ corresponds to the location of the centre of mass of the rigid body, and $\mathbf{R}$ is the rotation around the centre of mass. The generalized velocity of a rigid body can be written as $\boldsymbol{\nu} = (\mathbf{V}, \boldsymbol{\omega})$, consisting of a velocity $\mathbf{V}$ and angular velocity

$\boldsymbol{\omega}$ (also assumed to be about the centre of mass). We can use these quantities to compute the velocity of the material at a point $\mathbf{x}$ on the rigid body:

$$\mathbf{v}(\mathbf{x}) = \mathbf{V} + (\mathbf{x} - \mathbf{X}) \times \boldsymbol{\omega}$$

Note that this computation involves the cross product $\times$. In Appendix A we provide a 2D interpretation of this and related operators. The generalized mass $\mathcal{M} = (\mathbf{M}, \mathcal{I})$ of a rigid body consists of its mass matrix $\mathbf{M}$ and moment of inertia matrix $\mathcal{I}$, and $\mathcal{M}^{-1}$ denotes the component-wise inverse $(\mathbf{M}^{-1}, \mathcal{I}^{-1})$. We use the notation $\mathcal{F} = (\mathbf{F}, \boldsymbol{\tau})$ to denote the generalized body forces, consisting of the body forces $\mathbf{F}$ and body torques $\boldsymbol{\tau}$. Note that in 2 dimensions the rotational quantities $\mathbf{R}, \boldsymbol{\omega}, \mathcal{I}, \boldsymbol{\tau}$ can be represented by scalars.

The equations of motion for a rigid body are:

$$\begin{cases} \frac{d\mathbf{T}}{dt} = \boldsymbol{\nu} \\ \frac{d\boldsymbol{\nu}}{dt} = \mathcal{M}^{-1} \mathcal{F} \end{cases}$$

3.1.3 Fluid-rigid body coupling

Consider a rigid body B fully immersed in a fluid. We assume the material of the rigid body is not porous, so there is no penetration of the fluid into the solid. Let $\hat{\mathbf{n}}$ be the outward-facing unit normal vector to a point on the surface of the rigid body ∂B . At a point on the surface, the fluid velocity in the normal direction must be equal to the surface velocity in this direction, i.e.,

$$\mathbf{u} \cdot \hat{\mathbf{n}} = (\mathbf{V} + (\mathbf{x} - \mathbf{X}) \times \boldsymbol{\omega}) \cdot \hat{\mathbf{n}}.$$

In order to achieve this, there must be force transfer between the fluid and the rigid body. The force exerted by the fluid on the rigid body comes from the fluid pressure applied to its surface in the direction $-\hat{\mathbf{n}}$. Thus, the total force is

$$\mathbf{F}_{f \rightarrow B} = \iint_{\partial B} p(-\hat{\mathbf{n}}).$$

Similarly, the torque applied to the solid is

$$\boldsymbol{\tau}_{f \rightarrow B} = \iint_{\partial B} p(\mathbf{x} - \mathbf{X}) \times (-\hat{\mathbf{n}}).$$

We use notation $\mathcal{J}$ to encapsulate the translational and rotational components of this generalized force as a function of p :

$$\mathcal{J}(p) := \mathcal{F}_{f \rightarrow B} = \begin{pmatrix} \mathbf{J}_{\rightarrow}(p) \\ \mathbf{J}_{\circ}(p) \end{pmatrix} = \begin{pmatrix} -\iint_{\partial B} p \hat{\mathbf{n}} \\ -\iint_{\partial B} p(\mathbf{x} - \mathbf{X}) \times \hat{\mathbf{n}} \end{pmatrix}$$

The force and torque exerted by the solid on the fluid are the negatives of this. In our developments, we consider these coupling forces separate from the rest of our forces and torques $\mathbf{f}$, $\mathbf{F}$, $\boldsymbol{\tau}$, which refer to external forces. Note that $\mathcal{J}$ depends on the rigid body's transformation $\mathbf{T}$ not only in its formula, but also because $\mathbf{T}$ determines the location of the surface ∂B .

Thus for a coupled system with rigid bodies $B_1, \dots, B_N$, and denoting the fluid interior at time t with Ω , we have the equations of motion:

$$\begin{cases} \rho \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \right) = -\nabla p + \mathbf{f} & \text{on } \Omega \\ \nabla \cdot \mathbf{u} = 0 & \text{on } \Omega \\ \mathbf{u} \cdot \hat{\mathbf{n}} = (\mathbf{V}_k + (\mathbf{x} - \mathbf{X}_k) \times \boldsymbol{\omega}_k) \cdot \hat{\mathbf{n}} & \text{on } \partial B_k \text{ for each } k \in \{1, \dots, N\} \\ \frac{d\mathbf{T}_k}{dt} = \boldsymbol{\nu}_k & \text{for each } k \in \{1, \dots, N\} \\ \frac{d\boldsymbol{\nu}_k}{dt} = \mathcal{M}_k^{-1} (\mathcal{F}_k + \mathcal{J}_k(p)) & \text{for each } k \in \{1, \dots, N\} \end{cases}$$

3.2 Solving the coupled fluid-rigid body equations

3.2.1 Operator splitting

The standard method for solving the fluid equations in computer graphics employs a mathematical technique called *operator splitting*, in which the constituent pieces of a PDE are solved sequentially rather than simultaneously [8, 20]. This will be explored in more depth in sections 4 and 5. For now we simply state that after discretizing in time, we encounter

the PDE in space:

$$\begin{cases} \nabla \cdot \left(\mathbf{u} - \frac{\Delta t}{\rho} \nabla p \right) = 0 & \text{on } \Omega \\ \left(\mathbf{u} - \frac{\Delta t}{\rho} \nabla p \right) \cdot \hat{\mathbf{n}} = \left(\mathbf{V}_k + \Delta t \mathbf{M}_k^{-1} \mathbf{J}_{k,\rightarrow}(p) \right) \cdot \hat{\mathbf{n}} \\ \quad + \left((\mathbf{x} - \mathbf{X}_k) \times \left(\boldsymbol{\omega}_k + \Delta t \boldsymbol{\mathcal{I}}_k^{-1} \mathbf{J}_{k,\odot}(p) \right) \right) \cdot \hat{\mathbf{n}} & \text{on } \partial B_k \end{cases} \quad (3.1)$$

This is a Poisson problem in p over the fluid domain Ω , with Neumann boundary conditions depending on rigid body velocities. In effect, this PDE is an *incompressibility* and *compatibility* constraint: its solution yields a velocity field $\mathbf{u}^* = \mathbf{u} - \frac{\Delta t}{\rho} \nabla p$ that is divergence-free on the fluid domain, and which synchronizes the normal velocity of the fluid on the boundary of a rigid body with the rigid body's own velocity.

3.2.2 Variational formulation

In effect, the solution of (3.1) is the Helmholtz decomposition of $\mathbf{u}$, i.e., the splitting of $\mathbf{u}$ into two orthogonal components: the gradient of a scalar function $\frac{\Delta t}{\rho} \nabla p$ and a divergence-free component $\mathbf{u}^* = \mathbf{u} - \frac{\Delta t}{\rho} \nabla p$ [7].

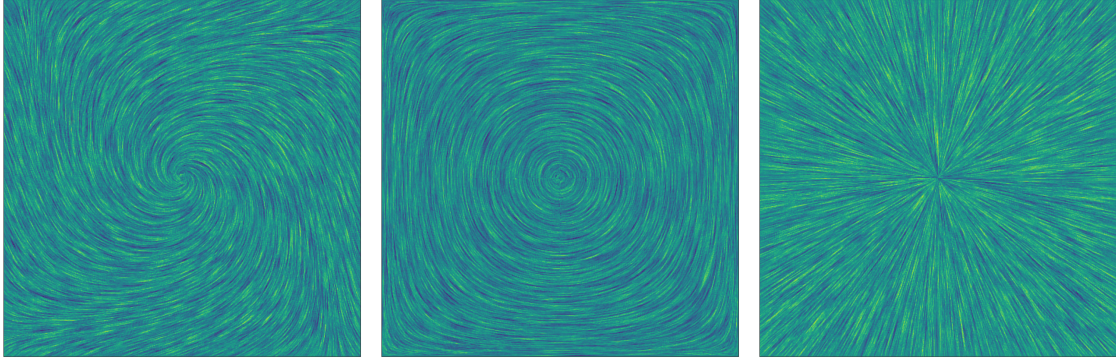


Figure 3.1: A given vector field (left) can be decomposed as a sum of a divergence-free vector field (middle) and the gradient of a scalar function (right).

Bridson and Müller-Fischer [10] show that for static boundaries, (3.1) can be reformulated as a minimization over p of the fluid's kinetic energy, i.e., a removal of the gradient component leaving only the divergence-free component. Batty [5] states without proof

that the minimization approach can be extended to encapsulate the rigid body dynamics as well. We provide the proof here.

Consider the system's total kinetic energy:

$$E(p) = \frac{1}{2} \iiint_{\Omega} \rho \left\| \mathbf{u} - \frac{\Delta t}{\rho} \nabla p \right\|^2 + \sum_{k=1}^N \frac{1}{2} \left(\boldsymbol{\nu}_k + \Delta t \mathbf{M}_k^{-1} \mathcal{J}_k(p) \right)^\top \mathbf{M}_k \left(\boldsymbol{\nu}_k + \Delta t \mathbf{M}_k^{-1} \mathcal{J}_k(p) \right)$$

Suppose p minimizes this energy. Let q be an arbitrary nonzero function on Ω , and consider the perturbation function $g(\epsilon) = E(p + \epsilon q)$:

$$\begin{aligned} g(\epsilon) = & \iiint_{\Omega} \left[\frac{1}{2} \rho \left\| \mathbf{u} - \frac{\Delta t}{\rho} \nabla p \right\|^2 + \epsilon \rho \frac{\Delta t}{\rho} \nabla q \cdot \left(\mathbf{u} - \frac{\Delta t}{\rho} \nabla p \right) + \frac{1}{2} \epsilon^2 \rho \frac{\Delta t^2}{\rho^2} \nabla q \cdot \nabla q \right] \\ & + \sum_{k=1}^N \frac{1}{2} \left(\boldsymbol{\nu}_k + \Delta t \mathbf{M}_k^{-1} \mathcal{J}_k(p) \right)^\top \mathbf{M}_k \left(\boldsymbol{\nu}_k + \Delta t \mathbf{M}_k^{-1} \mathcal{J}_k(p) \right) \\ & + \sum_{k=1}^N \epsilon \Delta t \left(\boldsymbol{\nu}_k + \Delta t \mathbf{M}_k^{-1} \mathcal{J}_k(p) \right)^\top \mathcal{J}_k(q) \\ & + \sum_{k=1}^N \frac{1}{2} \epsilon^2 \Delta t^2 \mathcal{J}_k(q)^\top \mathbf{M}_k^{-1} \mathcal{J}_k(q) \end{aligned}$$

Since $p = p + 0q$ minimizes E , $\epsilon = 0$ must be the minimizer of g , so we must have $g'(0) = 0$:

$$0 = \iiint_{\Omega} \nabla q \cdot \left(\mathbf{u} - \frac{\Delta t}{\rho} \nabla p \right) + \sum_{k=1}^N \left(\boldsymbol{\nu}_k + \Delta t \mathbf{M}_k^{-1} \mathcal{J}_k(p) \right)^\top \mathcal{J}_k(q)$$

Integrating by parts, we can transform the first term:

$$\begin{aligned} 0 = & - \iiint_{\Omega} q \nabla \cdot \left(\mathbf{u} - \frac{\Delta t}{\rho} \nabla p \right) + \iint_{\partial\Omega} q \left(\mathbf{u} - \frac{\Delta t}{\rho} \nabla p \right) \cdot \hat{\mathbf{n}} \\ & + \sum_{k=1}^N \left(\boldsymbol{\nu}_k + \Delta t \mathbf{M}_k^{-1} \mathcal{J}_k(p) \right)^\top \mathcal{J}_k(q) \end{aligned}$$

Letting $\boldsymbol{\nu}_k^* = (\mathbf{V}_k^*, \boldsymbol{\omega}_k^*) = \boldsymbol{\nu}_k + \Delta t \mathbf{M}_k^{-1} \mathcal{J}_k(p)$ and expanding $\mathcal{J}_k(q)$, we get

$$\begin{aligned} 0 = & - \iiint_{\Omega} q \nabla \cdot \left(\mathbf{u} - \frac{\Delta t}{\rho} \nabla p \right) + \iint_{\partial \Omega} q \left(\mathbf{u} - \frac{\Delta t}{\rho} \nabla p \right) \cdot \hat{\mathbf{n}} \\ & - \sum_{k=1}^N \iint_{\partial B_k} \left(q \mathbf{V}_k^* \cdot \hat{\mathbf{n}} + \underbrace{\boldsymbol{\omega}_k^* \cdot (\mathbf{x} - \mathbf{X}_k) \times \hat{\mathbf{n}}}_{=(\mathbf{x} - \mathbf{X}_k) \times \boldsymbol{\omega}_k^* \cdot \hat{\mathbf{n}}} \right) \end{aligned}$$

Now, noticing that $\partial \Omega = \bigcup_k \partial B_k$, we can combine terms

$$\begin{aligned} 0 = & - \iiint_{\Omega} q \nabla \cdot \left(\mathbf{u} - \frac{\Delta t}{\rho} \nabla p \right) \\ & + \sum_{k=1}^N \iint_{\partial B_k} q \left(\mathbf{u} - \frac{\Delta t}{\rho} \nabla p - (\mathbf{V}_k^* + (\mathbf{x} - \mathbf{X}_k) \times \boldsymbol{\omega}_k^*) \right) \cdot \hat{\mathbf{n}} \end{aligned}$$

Since q is arbitrary, this must hold for all q . However, each integral is a product of q with another factor, and the region of each integral is disjoint, so the other factor must be 0 for each integral. Since these other factors are exactly the equations of the coupled PDE, minimizing E results in solving (3.1).

We use the Pressure operator to denote solving this minimization problem for p :

$$\begin{aligned} p = \text{Pressure} \left(\mathbf{u}, \{\boldsymbol{\nu}_k\}_{k=1, \dots, N}, \partial \Omega = \bigcup_k \partial B_k \right) \\ \Updownarrow \\ p \text{ minimizes } E(p) \\ \Updownarrow \\ p \text{ solves (3.1)} \end{aligned}$$

3.2.3 Spatial discretization

Discretizing the PDE (3.1) directly lends itself to straightforward treatment of the interior of the fluid domain. However, with this approach it is unclear how to discretize the fluid-solid boundaries and consistently exchange forces across them. Conversely, by discretizing the variational formulation above, i.e., by discretizing of $E(p)$, Batty et al. [6] derive a natural treatment of boundary conditions and force exchange. In this section we outline their method applied to a 2D simulation.

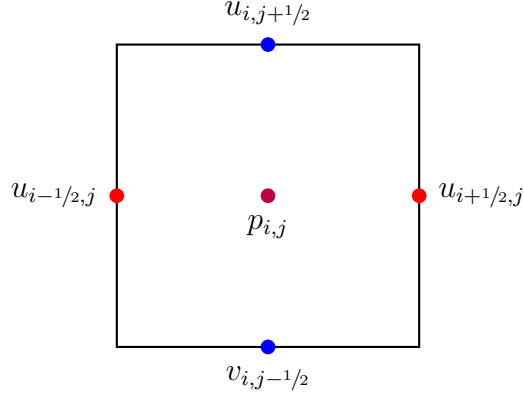


Figure 3.2: Placement of velocity (u, v) and pressure p samples in a single cell of a staggered grid.

We discretize quantities on a staggered (marker-and-cell [24]) grid: pressure samples are located at cell centres, while velocity samples represent fluid fluxes on cell faces (see figure 3.2). We use a square grid, so each cell has length and width Δx .

Since we are discretizing $E(p)$ rather than (3.1) directly, we only need to discretize the operators ∇ and $\mathcal{J}_k$ applied to p . We do so by constructing them as matrices, in particular using the $\mathbf{G}$ matrix as the discrete gradient. In practice, its transpose $\mathbf{G}^\top$ acts like a discrete divergence.

The fluid's kinetic energy must be integrated over the fluid domain, i.e., over the mass of the fluid. To achieve this, we voxelize our arbitrarily-shaped domain by computing the volume of fluid in the cell around each (staggered) velocity sample. Thus the kinetic energy of a fluid with velocity $\mathbf{u} = (u, v)$ discretized on a staggered grid is

$$KE_f = \frac{1}{2}\rho \left(\sum_{i,j} \text{vol}_{i+1/2,j}^f u_{i+1/2,j}^2 + \sum_{i,j} \text{vol}_{i,j+1/2}^f v_{i,j+1/2}^2 \right)$$

The volume samples themselves are estimated as the approximate lengths of the associated cell edges times the grid size as suggested by Batty et al. [6] and presented by Ng et al. [33]. This corresponds to a finite volume perspective, where the exposed cell edges affect the “flux” of fluid through the cell. The cell edge lengths are approximated by estimating

their intersection with the boundary:

$$\text{vol}_{i+1/2,j}^f = \Delta x^2 f_{i+1/2,j}$$

$$\text{where } f_{i+1/2,j} = \begin{cases} \frac{\phi_{i+1/2,j-1/2}}{\phi_{i+1/2,j-1/2} - \phi_{i+1/2,j+1/2}} & \text{if } \phi_{i+1/2,j-1/2} \leq 0, \phi_{i+1/2,j+1/2} > 0 \\ \frac{\phi_{i+1/2,j+1/2}}{\phi_{i+1/2,j+1/2} - \phi_{i+1/2,j-1/2}} & \text{if } \phi_{i+1/2,j-1/2} > 0, \phi_{i+1/2,j+1/2} \leq 0 \\ 1 & \text{if } \phi_{i+1/2,j-1/2} \leq 0, \phi_{i+1/2,j+1/2} \leq 0 \\ 0 & \text{if } \phi_{i+1/2,j-1/2} > 0, \phi_{i+1/2,j+1/2} > 0 \end{cases}$$

and where ϕ is a sample of the fluid domain's signed distance function. An example computation is shown in figure 3.3. We compute a mass matrix $\mathbf{M}_f$ of volume samples multiplied by density for each staggered grid point.

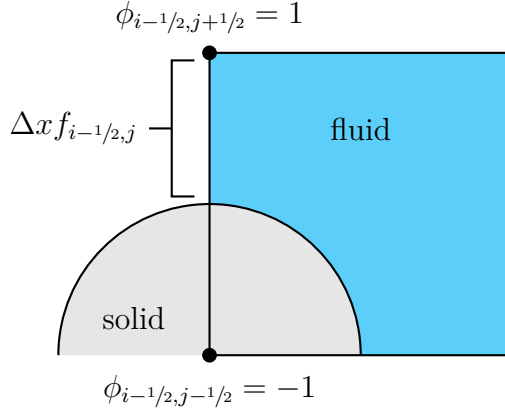


Figure 3.3: Finite volume-inspired cell edge length computation.

The $\mathcal{J}$ operator, which integrates pressure to compute a total force on a rigid body, can be discretized more easily if Stokes' theorem is applied. Starting with the translational component,

$$\begin{aligned} \mathbf{J}_{k,\rightarrow}(p) &= - \iint_{\partial B_k} p \hat{\mathbf{n}} \\ &= \iiint_{B_k} \nabla p. \end{aligned} \quad (\text{Stokes' theorem})$$

This can be discretized by taking finite differences of p over the staggered grid and summing the result weighted by solid volume samples:

$$\iiint_{B_k} \nabla p \approx \left(\sum_{i,j} \text{vol}_{i+1/2,j}^k \frac{p_{i+1,j} - p_{i,j}}{\Delta x} \right)$$

The rotational component can be discretized similarly:

$$\begin{aligned} \mathbf{J}_{k,\odot}(p) &= - \iint_{\partial B_k} p(\mathbf{x} - \mathbf{X}_k) \times \hat{\mathbf{n}} \\ &= \iiint_{B_k} \nabla \times p(\mathbf{x} - \mathbf{X}_k) \\ &= \iiint_{B_k} \frac{\partial}{\partial x} (p(y - Y_k)) - \frac{\partial}{\partial y} (p(x - X_k)) \quad (\text{in 2D}) \\ &= \iiint_{B_k} \frac{\partial p}{\partial x} (y - Y_k) - \frac{\partial p}{\partial y} (x - X_k) \\ &\approx \sum_{i,j} \text{vol}_{i+1/2,j}^k \left(\frac{p_{i+1,j} - p_{i,j}}{\Delta x} \right) (y_{i+1/2,j} - Y_k) \\ &\quad + \sum_{i,j} -\text{vol}_{i,j+1/2}^k \left(\frac{p_{i,j+1} - p_{i,j}}{\Delta x} \right) (x_{i,j+1/2} - X_k) \end{aligned}$$

Thus we may discretize our energy $E(p)$ as

$$\begin{aligned} E(p) &= \frac{1}{2} \left(\mathbf{u} - \frac{\Delta t}{\rho} \mathbf{G} p \right)^\top \mathbf{M}_f \left(\mathbf{u} - \frac{\Delta t}{\rho} \mathbf{G} p \right) \\ &\quad + \frac{1}{2} \sum_{k=1}^N \left(\boldsymbol{\nu}_k + \Delta t \mathcal{M}_k^{-1} \mathcal{J}_k p \right)^\top \mathcal{M}_k \left(\boldsymbol{\nu}_k + \Delta t \mathcal{M}_k^{-1} \mathcal{J}_k p \right). \end{aligned}$$

The discretized $E(p)$ is a positive-semi-definite quadratic form in p , so it is minimized when $\frac{dE}{dp} = 0$. Setting $\frac{dE}{dp} = 0$ and rearranging, we find the following linear system:

$$\Delta t \left(\frac{1}{\rho^2} \mathbf{G}^\top \mathbf{M}_f \mathbf{G} + \sum_{k=1}^N \mathcal{J}_k^\top \mathcal{M}_k^{-1} \mathcal{J}_k \right) p = \frac{1}{\rho} \mathbf{G}^\top \mathbf{M}_f \mathbf{u} - \sum_{k=1}^N \mathcal{J}_k^\top \boldsymbol{\nu}_k$$

This equation is a sparse positive semi-definite linear system in p . The $\frac{1}{\rho^2} \mathbf{G}^\top \mathbf{M}_f \mathbf{G}$ term contributes a standard volume-weighted five-point discrete Laplacian stencil:

$$\begin{aligned} \frac{1}{\rho^2} (\mathbf{G}^\top \mathbf{M}_f \mathbf{G} p)_{i,j} &= \frac{1}{\rho \Delta x^2} (\text{vol}_{i+1/2,j}^f p_{i+1,j} + \text{vol}_{i-1/2,j}^f p_{i-1,j} + \text{vol}_{i,j+1/2}^f p_{i,j+1} + \text{vol}_{i,j-1/2}^f p_{i,j-1} \\ &\quad - (\text{vol}_{i+1/2,j}^f + \text{vol}_{i-1/2,j}^f + \text{vol}_{i,j+1/2}^f + \text{vol}_{i,j-1/2}^f) p_{i,j}) \end{aligned}$$

Each $\mathcal{J}_k^\top \mathcal{M}_k^{-1} \mathcal{J}_k$ term contributes a dense sub-block of pressure samples being integrated over. However, note that only pressure samples near the surface of the solid are relevant, as the contributions of interior pressure variables are cancelled out, and because the solid volume disappears away from the surface. As a result, many pressure degrees of freedom within the solid can be eliminated from the linear system.

A wide range of numerical methods can be used to solve the linear system (3.2.3). Since the system is positive semi-definite we use a Cholesky decomposition, specifically the *SimplicialCholesky* solver from the Eigen linear algebra library [22] (which makes use of OpenMP [14] for parallelization). However, other direct and iterative solvers may be used. In particular, Aanjaneya presents a solver specifically formulated for fluid-rigid body coupling problems that exploits the low dimensionality of the rigid bodies, which have few degrees of freedom in their generalized velocity $\boldsymbol{\nu}$ [1].

3.3 Solving differential equations

In this section we discuss methods of solving ordinary differential equations (ODEs) as well as certain partial differential equations (PDEs). The methods presented here will form the bases of the time stepping schemes for the Incompressible Euler equations and the coupled fluid-rigid body equations in chapters 4 and 5 respectively.

3.3.1 Solving ODEs

An initial value problem (IVP) consists of an ordinary differential equation (ODE) and initial conditions. It is a problem of finding a function y that satisfies equations of the form:

$$\begin{cases} y' = \frac{dy}{dt} = f(t, y) \\ y(t^0) = y^0 \end{cases}$$

Numerical solution of IVPs is done by discretizing the ODE in time, so that a solution of the IVP at one instant in time may lead to a solution at the next instant. We will use superscripts to denote time step, so for a quantity $y(t)$ we use y^n as shorthand for $y(t^n)$.

Implicit Euler

The Implicit Euler method discretizes y' as

$$y'(t^{n+1}) \approx \frac{y^{n+1} - y^n}{\Delta t}$$

(where $\Delta t = t^{n+1} - t^n$) and evaluates f at the future point in time t^{n+1} , resulting in the method

$$\frac{y^{n+1} - y^n}{\Delta t} = f(t^{n+1}, y^{n+1}).$$

A numerical method to solve this equation for y^{n+1} provides a solution to the ODE.

BDF2

Backward Differentiation Formula (BDF) is a family of multi-step implicit time stepping methods. These use data from previous time steps, rather than only the current one, to compute the next one [2]. The second order method in this family, BDF2, is

$$\frac{\frac{3}{2}y^{n+1} - 2y^n + \frac{1}{2}y^{n-1}}{\Delta t} = f(t^{n+1}, y^{n+1}).$$

SBDF2

Ascher et al. [3] discuss a type of ODE time stepping scheme one may call “Implicit-Explicit” (IMEX), where one part of the ODE is discretized implicitly, and the other part is an explicit estimation. In particular, for an ODE

$$y' = f(y) + \nu g(y)$$

the authors suggest the semi-implicit BDF2 (SBDF2) scheme:

$$\frac{\frac{3}{2}y^{n+1} - 2y^n + \frac{1}{2}y^{n-1}}{\Delta t} = 2f(y^n) - f(y^{n-1}) + \nu g(y^{n+1})$$

where the g term is treated implicitly and the f term is approximated using linear extrapolation:

$$f(y^{n+1}) \approx 2f(y^n) - f(y^{n-1}).$$

Implicit Midpoint

The Implicit Midpoint method discretizes y' the same way as the Implicit Euler method, but evaluates f at an approximation of the midpoint rather than the end of the step:

$$\frac{y^{n+1} - y^n}{\Delta t} = f\left(t^{n+1/2}, \frac{1}{2}(y^n + y^{n+1})\right).$$

Define the intermediate variable $y^{n+1/2} = \frac{1}{2}(y^n + y^{n+1})$. Then,

$$\begin{aligned} \frac{y^{n+1} - y^n}{\Delta t} &= f\left(t^{n+1/2}, y^{n+1/2}\right) \\ \frac{y^{n+1} + y^n - y^n - y^n}{\Delta t} &= f\left(t^{n+1/2}, y^{n+1/2}\right) \\ \frac{2y^{n+1/2} - 2y^n}{\Delta t} &= f\left(t^{n+1/2}, y^{n+1/2}\right) \\ \frac{y^{n+1/2} - y^n}{\frac{1}{2}\Delta t} &= f\left(t^{n+1/2}, y^{n+1/2}\right) \end{aligned}$$

This is an Implicit Euler time step solving for $y^{n+1/2}$, so it can be solved as before. We can compute y^{n+1} from the solution for $y^{n+1/2}$, yielding the two-step procedure:

$$\frac{y^{n+1/2} - y^n}{\frac{1}{2}\Delta t} = f(t^{n+1/2}, y^{n+1/2}) \quad (3.2)$$

$$y^{n+1} = 2y^{n+1/2} - y^n \quad (3.3)$$

Thus, we can equivalently see this method as extrapolating an Implicit Euler half-step.

3.3.2 Method of characteristics

Consider a PDE of the form

$$\begin{cases} \frac{\partial q}{\partial t} + \mathbf{v} \cdot \nabla q = f \\ q(t^0, \mathbf{x}) = g(\mathbf{x}) \end{cases}$$

where q, f, g are scalar functions and $\mathbf{v}$ is a vector field. Let $\mathbf{x}(t)$ denote the trajectory of a “particle” flowing through the vector field, so at each point in time $\mathbf{x}$ satisfies

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}(\mathbf{x}).$$

Such a trajectory is called a *characteristic curve* of the PDE [13]. Calculating the time derivative of $q(t, \mathbf{x}(t))$:

$$\begin{aligned}\frac{dq(t, \mathbf{x}(t))}{dt} &= \frac{\partial q}{\partial t}(t, \mathbf{x}(t)) + \frac{d\mathbf{x}}{dt}(t) \cdot \nabla q(t, \mathbf{x}(t)) \\ &= \left(\frac{\partial q}{\partial t} + \mathbf{v} \cdot \nabla q \right)(t, \mathbf{x}(t)) \\ &= f(t, \mathbf{x}(t))\end{aligned}$$

Rewriting our PDE in a Lagrangian perspective following $\mathbf{x}(t)$ has turned it into an ODE in t along the trajectory $\mathbf{x}(t)$.

Advection

Suppose $f = 0$, so the problem reduces to

$$\begin{aligned}\frac{dq(t, \mathbf{x}(t))}{dt} &= 0 \\ q(t^0, \mathbf{x}) &= g(\mathbf{x}).\end{aligned}$$

Suppose furthermore that we would like to solve for q at a fixed point in time t^1 and space $\mathbf{x}(t^1) = \mathbf{x}^1$. Then, observing that q is constant along $\mathbf{x}(t)$:

$$q(t^1, \mathbf{x}^1) = q(t^1, \mathbf{x}(t^1)) = q(t^0, \mathbf{x}(t^0)) = g(\mathbf{x}^0)$$

As a result, finding $q(t^1, \mathbf{x}^1)$ is equivalent to solving the following initial value problem for time t^0 :

$$\begin{cases} \frac{d\mathbf{x}}{dt} = \mathbf{v}(\mathbf{x}) \\ \mathbf{x}(t^1) = \mathbf{x}^1 \end{cases} \quad (3.4)$$

In other words, we must *flow* $\mathbf{x}^1$ back in time through $\mathbf{v}$ to find $\mathbf{x}^0$, and query g there to compute $q(t^1, \mathbf{x}^1)$, thereby *transporting* $q(t^0, \mathbf{x}^0)$ to $q(t^1, \mathbf{x}^1)$.

The ODE (3.4) can be solved using standard methods, and in our work we use a third order Runge-Kutta method. Computing $q(t^1, \mathbf{x}^1)$ by solving (3.4) and transporting the old value $q(t^0, \mathbf{x}^0)$ is referred to as semi-Lagrangian advection, as it may solve for the quantity q at a fixed (Eulerian) point in space $\mathbf{x}^1$, but it does so by tracing the (Lagrangian) trajectory of the “material” at the point. We can compute the entire field $q(t^1, \cdot)$ at the new time by

applying this procedure for each point $\mathbf{x}^1$ in the domain. We denote this by the advection operator, which has notation:

$$q(t^1, \cdot) = \text{Advect} \left(q(t^0, \cdot) \text{ through } \mathbf{v}, \Delta t = t^1 - t^0 \right)$$

We also use the subscript “ a ” to denote that a quantity has been advected:

$$q_a = \text{Advect} (q \text{ through } \mathbf{v}, \Delta t)$$

For a field q stored on a grid, we implement the advection operator by solving the transport problem for each sample point on the grid. We use bicubic interpolation to estimate the values of q and $\mathbf{v}$ at points between the sample points. Since the advected value at each point is independent of the rest, we can easily parallelize this computation over the grid points using OpenMP [14].

Out of scope of this work are other methods of advection such as BFECC [15], MacCormack [41], covector advection [31], and more.

3.3.3 Method of characteristics for fluids

First note that if $\mathbf{x}(t)$ describes the position of a Lagrangian “particle” of fluid at time t , then its velocity is given by

$$\frac{d\mathbf{x}}{dt}(t) = \mathbf{u}(t, \mathbf{x}(t)).$$

This trajectory is a characteristic curve of the momentum equation of the Incompressible Euler equations (a *streamline*), so it reveals a Lagrangian perspective on the motion of the fluid [32].

$$\rho \left(\frac{d(\mathbf{u}(t, \mathbf{x}(t)))}{dt} \right) = -\nabla p(t, \mathbf{x}(t)) + \mathbf{f}(t, \mathbf{x}(t)) \quad (3.5)$$

$$\nabla \cdot \mathbf{u}(t, \cdot) = 0 \quad (3.6)$$

This effectively reframes the PDE as an ODE along the trajectory of the particle, and so ODE time integration methods may be applied.

Advection for fluids

Consider a quantity that is transported along a characteristic curve of a fluid:

$$\frac{dq(t, \mathbf{x}(t))}{dt} = 0$$

where

$$\frac{d\mathbf{x}}{dt}(t) = \mathbf{u}(t, \mathbf{x}(t))$$

and $\mathbf{u}$ is governed by the Incompressible Euler equations.

This situation is somewhat different from the setting in which advection was considered earlier: $\mathbf{u}$ evolves with t , while earlier we considered a static vector field $\mathbf{v}$ governing the trajectory of the characteristic curve. As a result, our advection operator can only approximate a solution by advecting through a *frozen* velocity field:

$$q(t^1, \cdot) = \text{Advect} \left(q(t^0, \cdot) \text{ through } \mathbf{u}(t^*, \cdot), \Delta t \right)$$

where t^* is a particular instant in time at which the velocity field is known.

Chapter 4

Fluid temporal discretizations

In this chapter, we catalogue and discuss temporal discretizations from the literature for the interior of the fluid domain, agnostic of any coupling. We consider three families of time stepping schemes: Implicit Euler (also called “Advection-Projection”), BDF2, and Advection-Reflection, each family corresponding to a different underlying ODE time stepping scheme.

4.1 Implicit Euler

The standard operator splitting method first introduced to computer graphics by Stam can be interpreted as an application of Implicit Euler to a characteristic curve of the Incompressible Euler equations [42, 32]. This entails discretizing (3.5) and (3.6) using Implicit Euler:

$$\begin{aligned}\rho \left(\frac{\mathbf{u}(t^{n+1}, \mathbf{x}(t^{n+1})) - \mathbf{u}(t^n, \mathbf{x}(t^n))}{\Delta t} \right) &= -\nabla p(t^{n+1}, \mathbf{x}(t^{n+1})) + \mathbf{f}(t^{n+1}, \mathbf{x}(t^{n+1})) \\ \nabla \cdot \mathbf{u}(t^{n+1}, \cdot) &= 0\end{aligned}$$

Rewriting using superscripts to denote time step:

$$\begin{aligned}\rho \left(\frac{\mathbf{u}^{n+1}(\mathbf{x}^{n+1}) - \mathbf{u}^n(\mathbf{x}^n)}{\Delta t} \right) &= -\nabla p^{n+1}(\mathbf{x}^{n+1}) + \mathbf{f}^{n+1}(\mathbf{x}^{n+1}) \\ \nabla \cdot \mathbf{u}^{n+1} &= 0\end{aligned}\tag{4.1}$$

Since $\mathbf{x}^n$ and $\mathbf{x}^{n+1}$ are along the same trajectory, we can transport the quantity $\mathbf{u}^n(\mathbf{x}^n)$ to the position $\mathbf{x}^{n+1}$ by advection: $\mathbf{u}_a^n(\mathbf{x}^{n+1}) = \mathbf{u}^n(\mathbf{x}^n)$. To compute this in practice, we approximate $\mathbf{u}$ with the frozen velocity field $\mathbf{u}^n$. Thus we can rewrite (4.1) for a general point $\mathbf{x} \in \Omega$ as:

$$\mathbf{u}_a^n = \text{Advect}(\mathbf{u}^n \text{ through } \mathbf{u}^n, \Delta t) \quad (4.2)$$

$$\rho \left(\frac{\mathbf{u}^{n+1} - \mathbf{u}_a^n}{\Delta t} \right) = -\nabla p^{n+1} + \mathbf{f}^{n+1} \quad (4.3)$$

$$\nabla \cdot \mathbf{u}^{n+1} = 0 \quad (4.4)$$

Now our problem is in a fully Eulerian setting. Combining the last two equations, we find:

$$\nabla \cdot \left(\left(\mathbf{u}_a^n + \frac{\Delta t}{\rho} \mathbf{f}^{n+1} \right) - \frac{\Delta t}{\rho} \nabla p^{n+1} \right) = 0$$

The solution p^{n+1} to this Poisson problem can be found using the pressure solve discussed in section 3.2. Using the solution, we can construct $\mathbf{u}^{n+1}$:

$$p^{n+1} = \text{Pressure} \left(\mathbf{u}_a^n + \frac{\Delta t}{\rho} \mathbf{f}^{n+1} \right)$$

$$\mathbf{u}^{n+1} = \mathbf{u}_a^n + \frac{\Delta t}{\rho} \mathbf{f}^{n+1} - \frac{\Delta t}{\rho} \nabla p^{n+1}$$

We have thus recovered the standard operator splitting “Advection-Projection” time stepping scheme, splitting the solution of the incompressible Euler equations into sequential operations:

	fluid
1	$\mathbf{u}_a^n \leftarrow \text{Advect}(\mathbf{u}^n \text{ through } \mathbf{u}^n, \Delta t)$
2	$\mathbf{u}_f^{n+1} \leftarrow \mathbf{u}_a^n + \frac{\Delta t}{\rho} \mathbf{f}^{n+1}$
3	$p^{n+1} \leftarrow \text{Pressure}(\mathbf{u}_f^{n+1})$
4	$\mathbf{u}^{n+1} \leftarrow \mathbf{u}_f^{n+1} - \frac{\Delta t}{\rho} \nabla p^{n+1}$

We present our fluid time stepping algorithms in a table format, under a single “fluid” column for now. In Chapter 5 we will extend some of these algorithms to support rigid body coupling, and add a “solids” column.

The method we have recovered here is a standard scheme in fluid simulation for computer graphics, but it differs slightly from the one originally proposed by Stam [42]: The original method discretizes the body force $\mathbf{f}$ at the beginning of the time step rather than at the end, deviating from the Implicit Euler strategy used for deriving the overall scheme.

4.2 BDF2-based methods

4.2.1 BDF2 as presented by Xiu and Karniadakis

We can similarly use the method of characteristics to apply the BDF2 discretization to the fluid equations:

$$\rho \left(\frac{\frac{3}{2}\mathbf{u}^{n+1}(\mathbf{x}^{n+1}) - 2\mathbf{u}^n(\mathbf{x}^n) + \frac{1}{2}\mathbf{u}^{n-1}(\mathbf{x}^{n-1})}{\Delta t} \right) = -\nabla p^{n+1}(\mathbf{x}^{n+1}) + \mathbf{f}^{n+1}(\mathbf{x}^{n+1})$$

$$\nabla \cdot \mathbf{u}^{n+1} = 0$$

As before, we can use advection to transport quantities between the different positions along the Lagrangian characteristic trajectory. We would like to transport $\mathbf{u}^{n-1}(\mathbf{x}^{n-1})$ to $\mathbf{x}^{n+1}$, so we compute the advected field $\mathbf{u}_a^{n-1}(\mathbf{x}^{n+1}) = \mathbf{u}^{n-1}(\mathbf{x}^{n-1})$. We use the frozen velocity field $\mathbf{u}^n$, which is the velocity at the midpoint of the trajectory from $\mathbf{x}^{n-1}$ to $\mathbf{x}^{n+1}$:

$$\mathbf{u}_a^{n-1} = \text{Advect} \left(\mathbf{u}^{n-1} \text{ through } \mathbf{u}^n, 2\Delta t \right)$$

Similarly, we would like to solve the advection problem $\mathbf{u}_a^n(\mathbf{x}^{n+1}) = \mathbf{u}^n(\mathbf{x}^n)$, ideally by advecting through the frozen velocity field at the midpoint of the trajectory. We can approximate this midpoint velocity using a backward difference discretization of $\frac{\partial \mathbf{u}}{\partial t}$:

$$\begin{aligned} \mathbf{u}^{n+1/2} &\approx \mathbf{u}^n + \frac{1}{2}\Delta t \frac{\partial \mathbf{u}}{\partial t} \\ &\approx \mathbf{u}^n + \frac{1}{2}\Delta t \left(\frac{\mathbf{u}^n - \mathbf{u}^{n-1}}{\Delta t} \right) \\ &= \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1} \end{aligned}$$

Using this field, we can perform the advection:

$$\mathbf{u}_a^n = \text{Advect} \left(\mathbf{u}^n \text{ through } \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}, \Delta t \right)$$

Substituting the advected fields into our method, we get:

$$\rho \left(\frac{\frac{3}{2}\mathbf{u}^{n+1} - 2\mathbf{u}_a^n + \frac{1}{2}\mathbf{u}_a^{n-1}}{\Delta t} \right) = -\nabla p^{n+1} + \mathbf{f}^{n+1} \quad (4.5)$$

$$\nabla \cdot \mathbf{u}^{n+1} = 0 \quad (4.6)$$

Similarly to our treatment of (4.3), we can combine (4.5) and (4.6) to reduce them to a Poisson problem in p^{n+1} :

$$\nabla \cdot \left(\left(\frac{4}{3}\mathbf{u}_a^n - \frac{1}{3}\mathbf{u}_a^{n-1} + \frac{2\Delta t}{3\rho}\mathbf{f}^{n+1} \right) - \frac{2\Delta t}{3\rho}\nabla p^{n+1} \right) = 0$$

Rewriting the solution to this problem using the Pressure operator, we find:

$$p^{n+1} \leftarrow \frac{3}{2} \text{Pressure} \left(\frac{4}{3}\mathbf{u}_a^n - \frac{1}{3}\mathbf{u}_a^{n-1} + \frac{2\Delta t}{3\rho}\mathbf{f}^{n+1} \right) \quad (4.7)$$

Subtracting the appropriate pressure gradient to make the final velocity field divergence-free yields the final sequential algorithm:

	fluid
1	$\mathbf{u}_a^{n-1} \leftarrow \text{Advect}(\mathbf{u}^{n-1} \text{ through } \mathbf{u}^n, 2\Delta t)$
2	$\mathbf{u}_a^n \leftarrow \text{Advect}(\mathbf{u}^n \text{ through } \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}, \Delta t)$
3	$\mathbf{u}_f^{n+1} \leftarrow \frac{4}{3}\mathbf{u}_a^n - \frac{1}{3}\mathbf{u}_a^{n-1} + \frac{2\Delta t}{3\rho}\mathbf{f}^{n+1}$
4	$p^{n+1} \leftarrow \frac{3}{2} \text{Pressure}(\mathbf{u}_f^{n+1})$
5	$\mathbf{u}^{n+1} \leftarrow \mathbf{u}_f^{n+1} - \frac{2\Delta t}{3\rho}\nabla p^{n+1}$

4.2.2 Material acceleration

Nielsen et al. [34] reformulate BDF2 to incorporate an acceleration variable, in order to provide a BDF2-based method at lower computational cost than the method of Xiu and Karniadakis. Separating BDF2's discretization of y' into two differencing operations across steps of Δt , we can rewrite it as a combination of two derivative terms, one of which is

already known:

$$\begin{aligned}
\frac{\frac{3}{2}y^{n+1} - 2y^n + \frac{1}{2}y^{n-1}}{\Delta t} &= f(t^{n+1}, y^{n+1}) \\
\frac{3}{2} \left(\frac{y^{n+1} - y^n}{\Delta t} \right) - \underbrace{\frac{1}{2} \left(\frac{y^n - y^{n-1}}{\Delta t} \right)}_{s^{n-1/2}} &= f(t^{n+1}, y^{n+1}) \\
\frac{y^{n+1} - y^n}{\Delta t} &= \frac{2}{3} f(t^{n+1}, y^{n+1}) + \frac{1}{3} s^{n-1/2}
\end{aligned}$$

Using the term $\mathbf{a}^{n-1/2} = \frac{1}{\Delta t}(\mathbf{u}^n - \mathbf{u}^{n-1})$ to refer to the fluid's acceleration at departure point $\mathbf{x}^n$, and applying this discretization to a characteristic curve of the fluid, we arrive at the equations:

$$\begin{aligned}
\rho \left(\frac{\mathbf{u}^{n+1}(\mathbf{x}^{n+1}) - \mathbf{u}^n(\mathbf{x}^n)}{\Delta t} \right) &= \frac{2}{3} \left(-\nabla p^{n+1}(\mathbf{x}^{n+1}) + \mathbf{f}^{n+1}(\mathbf{x}^{n+1}) \right) + \frac{1}{3} \rho \mathbf{a}^{n-1/2}(\mathbf{x}^n) \\
\nabla \cdot \mathbf{u}^{n+1} &= 0
\end{aligned}$$

We can evaluate all quantities at $\mathbf{x}^{n+1}$ by advecting our velocity and acceleration through an estimate of $\mathbf{u}^{n+1/2}$ as in the method of Xiu and Karniadakis, resulting in condensed equations

$$\begin{aligned}
\rho \left(\frac{\mathbf{u}^{n+1} - \mathbf{u}_a^n}{\Delta t} \right) &= \frac{2}{3} \left(-\nabla p^{n+1} + \mathbf{f}^{n+1} \right) + \frac{1}{3} \rho \mathbf{a}_a^{n-1/2} \\
\nabla \cdot \mathbf{u}^{n+1} &= 0
\end{aligned}$$

Note that we can advect both $\mathbf{u}^n$ and $\mathbf{a}^{n-1/2}$ at once, as we only need to trace a single trajectory from $\mathbf{x}^n$ to $\mathbf{x}^{n+1}$ to transport both quantities. Meanwhile, the BDF2 method of Xiu and Karniadakis requires two separate advections, as both $\mathbf{x}^n$ and $\mathbf{x}^{n-1}$ must be computed.

Lastly, we perform a pressure solve just as in (4.7), yielding the method (as presented by Nielsen et al.):

	fluid
1	$\mathbf{u}_a^n \leftarrow \text{Advect} \left(\mathbf{u}^n \text{ through } \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}, \Delta t \right)$
2	$\mathbf{a}_a^{n+1/2} \leftarrow \text{Advect} \left(\mathbf{a}^{n-1/2} \text{ through } \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}, \Delta t \right)$
3	$\mathbf{u}_f^{n+1} \leftarrow \mathbf{u}_a^n + \frac{1}{3}\mathbf{a}_a^{n-1/2} + \frac{2\Delta t}{3\rho}\mathbf{f}^{n+1}$
4	$p^{n+1} \leftarrow \frac{3}{2} \text{Pressure} \left(\mathbf{u}_f^{n+1} \right)$
5	$\mathbf{u}^{n+1} \leftarrow \mathbf{u}_f^{n+1} - \frac{2\Delta t}{3\rho}\nabla p^{n+1}$
6	$\mathbf{a}^{n+1/2} \leftarrow \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t}$

Improved material acceleration

The presented update formula for the acceleration variable is

$$\mathbf{a}^{n+1/2} \leftarrow \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t}$$

However, the condensed notation hides a mistake: $\mathbf{u}^n$ and $\mathbf{u}^{n+1}$ are not discretized at the same point in a semi-Lagrangian particle's trajectory $\mathbf{x}(t)$. Sampling $\mathbf{u}^n$ at $\mathbf{x}^n$ would provide a better finite difference estimation of the particle's acceleration:

$$\begin{aligned}
\mathbf{a}^{n+1/2}(\mathbf{x}^{n+1}) &\leftarrow \frac{d\mathbf{u}(t, \mathbf{x}(t))}{dt} \\
&\approx \frac{\mathbf{u}^{n+1}(\mathbf{x}^{n+1}) - \mathbf{u}^n(\mathbf{x}^n)}{\Delta t} \\
&= \frac{\mathbf{u}^{n+1}(\mathbf{x}^{n+1}) - \mathbf{u}_a^n(\mathbf{x}^{n+1})}{\Delta t} \\
\mathbf{a}^{n+1/2} &\leftarrow \frac{\mathbf{u}^{n+1} - \mathbf{u}_a^n}{\Delta t}
\end{aligned}$$

This yields the following final method, which differs from the original only in the acceleration update on line 6:

	fluid
1	$\mathbf{u}_a^n \leftarrow \text{Advect} \left(\mathbf{u}^n \text{ through } \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}, \Delta t \right)$
2	$\mathbf{a}_a^{n-1/2} \leftarrow \text{Advect} \left(\mathbf{a}^{n-1/2} \text{ through } \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}, \Delta t \right)$
3	$\mathbf{u}_f^{n+1} \leftarrow \mathbf{u}_a^n + \frac{1}{3}\mathbf{a}_a^{n-1/2} + \frac{2\Delta t}{3\rho}\mathbf{f}^{n+1}$
4	$p^{n+1} \leftarrow \frac{3}{2} \text{Pressure} \left(\mathbf{u}_f^{n+1} \right)$
5	$\mathbf{u}^{n+1} \leftarrow \mathbf{u}_f^{n+1} - \frac{2\Delta t}{3\rho}\nabla p^{n+1}$
6	$\mathbf{a}^{n+1/2} \leftarrow \frac{\mathbf{u}^{n+1} - \mathbf{u}_a^n}{\Delta t}$

4.2.3 BDF2 as presented by Gibou and Min

The approach Gibou and Min [19] take differs slightly from that of Xiu and Karniadakis as it involves tracking an explicit time-evolving pressure variable p rather than only solving for pressure at the end of the time step. This is referred to as an “incremental” pressure approach [23].

Jiang et al. [25] state that the evolution of the pressure field follows the equation:

$$\frac{\partial p}{\partial t} + \mathbf{u} \cdot \nabla p = 0$$

i.e., pressure is simply advected in time. Thus, if p^n is known, $p^{n+1}(\mathbf{x}^{n+1}) = p^n(\mathbf{x}^n)$, so p^{n+1} can be approximated as:

$$p^{n+1} \approx p_a^n = \text{Advect} \left(p^n \text{ through } \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}, \Delta t \right)$$

However, an update based on this advected pressure is not guaranteed to make the flow divergence-free and satisfy the compatibility constraint, so a final pressure projection is still needed. This final pressure projection is used to correct the advected pressure variable. This yields a final method:

	fluid
1	$\mathbf{u}_a^{n-1} \leftarrow \text{Advect}(\mathbf{u}^{n-1} \text{ through } \mathbf{u}^n, 2\Delta t)$
2	$\mathbf{u}_a^n \leftarrow \text{Advect}(\mathbf{u}^n \text{ through } \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}, \Delta t)$
3	$p_a^n \leftarrow \text{Advect}(p^n \text{ through } \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}, \Delta t)$
4	$\mathbf{u}_f^{n+1} \leftarrow \frac{4}{3}\mathbf{u}_a^0 - \frac{1}{3}\mathbf{u}_a^{n-1} + \frac{2\Delta t}{3\rho}(\mathbf{f}^{n+1} - \nabla p_a^n)$
5	$q^{n+1} \leftarrow \frac{3}{2} \text{Pressure}(\mathbf{u}_f^{n+1})$
6	$\mathbf{u}^{n+1} \leftarrow \mathbf{u}_f^{n+1} - \frac{2\Delta t}{3\rho} \nabla q^{n+1}$
7	$p^{n+1} \leftarrow p_a^n + q^{n+1}$

4.3 Advection-Reflection methods

The final family of time stepping schemes we consider are Advection-Reflection methods.

The second order Advection-Reflection method is based on the Implicit Midpoint method:

$$\frac{y^{n+1} - y^n}{\frac{1}{2}\Delta t} = f(t^{n+1/2}, y^{n+1/2}) \quad (4.8)$$

$$y^{n+1} = 2y^{n+1/2} - y^n \quad (4.9)$$

The first step of Implicit Midpoint is identical to a half-step using Implicit Euler. Accordingly, (4.8) is implemented as such:

	fluid
1	$\mathbf{u}_a^n \leftarrow \text{Advect}(\mathbf{u}^n \text{ through } \mathbf{u}^n, \frac{1}{2}\Delta t)$
2	$\mathbf{u}_f^{n+1/2} \leftarrow \mathbf{u}_a^n + \frac{\Delta t}{2\rho} \mathbf{f}^{n+1/2}$
3	$p^{n+1/2} \leftarrow 2 \text{Pressure}(\mathbf{u}_f^{n+1/2})$
4	$\mathbf{u}^{n+1/2} \leftarrow \mathbf{u}_f^{n+1/2} - \frac{\Delta t}{2\rho} \nabla p^{n+1/2}$

The second step, (4.9), can be applied to our fluid velocity using the method of characteristics:

$$\mathbf{u}^{n+1}(\mathbf{x}^{n+1}) = 2\mathbf{u}^{n+1/2}(\mathbf{x}^{n+1/2}) - \mathbf{u}^n(\mathbf{x}^n) \quad (4.10)$$

By advecting $\mathbf{u}^n$, we can evaluate all quantities on the right hand side of (4.10) at $\mathbf{x}^{n+1/2}$. We introduce a new vector field for the right hand side:

$$\hat{\mathbf{u}}^{n+1/2} := 2\mathbf{u}^{n+1/2} - \mathbf{u}_a^n$$

The operation to compute $\hat{\mathbf{u}}^{n+1/2}$ is referred to as *Reflection*, because the midpoint between it and the advected velocity is the pressure-projected velocity:

$$\begin{aligned} \frac{\mathbf{u}_a^n + \hat{\mathbf{u}}^{n+1/2}}{2} &= \frac{\mathbf{u}_a^n + 2\mathbf{u}^{n+1/2} - \mathbf{u}_a^n}{2} \\ &= \mathbf{u}^{n+1/2} \end{aligned}$$

Thus, $\hat{\mathbf{u}}^{n+1/2}$ is $\mathbf{u}_a^n$ *reflected* across the hyperplane of divergence-free flows (ignoring body forces) within the space of all vector fields $\mathfrak{X}(\Omega)$.

Rewriting (4.10) using $\hat{\mathbf{u}}^{n+1/2}$, we get:

$$\mathbf{u}^{n+1}(\mathbf{x}^{n+1}) = \hat{\mathbf{u}}^{n+1/2}(\mathbf{x}^{n+1/2})$$

This is a transport problem that can be solved using advection. Narain et al. solve it by advecting $\hat{\mathbf{u}}^{n+1/2}$ through an approximation of $\mathbf{u}^{n+1}$:

$$\begin{aligned} \mathbf{u}^{n+1} &\approx \mathbf{u}^{n+1/2} + \frac{1}{2}\Delta t \left. \frac{\partial \mathbf{u}}{\partial t} \right|_{t_{n+1/2}} && \text{(Taylor's theorem)} \\ &\approx \mathbf{u}^{n+1/2} + \frac{1}{2}\Delta t \left(\frac{\mathbf{u}^{n+1/2} - \mathbf{u}^n}{\frac{1}{2}\Delta t} \right) \\ &= 2\mathbf{u}^{n+1/2} - \mathbf{u}^n \end{aligned}$$

They show that this approximation leads to cancelling of errors, resulting in second-order accuracy [32].

The result of advection is not necessarily divergence-free, so a last pressure projection is needed to satisfy the incompressibility constraint. All together, this results in a final method:

	fluid
1	$\mathbf{u}_a^n \leftarrow \text{Advect} \left(\mathbf{u}^n \text{ through } \mathbf{u}^n, \frac{1}{2}\Delta t \right)$
2	$\mathbf{u}_f^{n+1/2} \leftarrow \mathbf{u}_a^n + \frac{\Delta t}{2\rho} \mathbf{f}^{n+1/2}$
3	$p^{n+1} \leftarrow 2 \text{Pressure} \left(\mathbf{u}_f^{n+1/2} \right)$
4	$\mathbf{u}^{n+1/2} \leftarrow \mathbf{u}_f^{n+1/2} - \frac{\Delta t}{2\rho} \nabla p^{n+1/2}$
5	$\hat{\mathbf{u}}^{n+1/2} \leftarrow 2\mathbf{u}^{n+1/2} - \mathbf{u}_a^n$
6	$\hat{\mathbf{u}}_a^{n+1/2} \leftarrow \text{Advect} \left(\hat{\mathbf{u}}^{n+1/2} \text{ through } 2\mathbf{u}^{n+1/2} - \mathbf{u}^n, \frac{1}{2}\Delta t \right)$
7	$p^{n+1} \leftarrow \text{Pressure} \left(\hat{\mathbf{u}}_a^{n+1/2} \right)$
8	$\mathbf{u}^{n+1} \leftarrow \hat{\mathbf{u}}_a^{n+1/2} - \frac{\Delta t}{\rho} \nabla p^{n+1}$

4.3.1 First-order Advection-Reflection

The second-order scheme described above was devised as a correction to another scheme, the first-order accurate Advection-Reflection scheme of Zehnder et al. [46]. We discuss in section 4.3.2 the energy preservation qualities that motivated this original scheme. The method presented is:

	fluid
1	$\mathbf{u}_a^n \leftarrow \text{Advect} \left(\mathbf{u}^n \text{ through } \mathbf{u}^n, \frac{1}{2}\Delta t \right)$
2	$\mathbf{u}_f^{n+1/2} \leftarrow \mathbf{u}_a^n + \frac{\Delta t}{2\rho} \mathbf{f}^{n+1/2}$
3	$p^{n+1} \leftarrow 2 \text{Pressure} \left(\mathbf{u}_f^{n+1/2} \right)$
4	$\mathbf{u}^{n+1/2} \leftarrow \mathbf{u}_f^{n+1/2} - \frac{\Delta t}{2\rho} \nabla p^{n+1/2}$
5	$\hat{\mathbf{u}}^{n+1/2} \leftarrow 2\mathbf{u}^{n+1/2} - \mathbf{u}_f^n$
6	$\hat{\mathbf{u}}_a^{n+1/2} \leftarrow \text{Advect} \left(\hat{\mathbf{u}}^{n+1/2} \text{ through } \mathbf{u}^{n+1/2}, \frac{1}{2}\Delta t \right)$
7	$\mathbf{u}_f^{n+1} \leftarrow \hat{\mathbf{u}}_a^{n+1/2} + \frac{\Delta t}{2\rho} \mathbf{f}^{n+1/2}$
8	$p^{n+1} \leftarrow \text{Pressure} \left(\hat{\mathbf{u}}_f^{n+1} \right)$
9	$\mathbf{u}^{n+1} \leftarrow \hat{\mathbf{u}}_f^{n+1} - \frac{\Delta t}{\rho} \nabla p^{n+1}$

This scheme incorrectly applies the midpoint force at the end of the material's trajectory, in step 7 of the above algorithm. Furthermore, it does not trace characteristic curves as accurately, as in step 6 it advects $\hat{\mathbf{u}}^{n+1/2}$ through $\mathbf{u}^{n+1/2}$ rather than through $2\mathbf{u}^{n+1/2} - \mathbf{u}^n$, missing the cancelling of errors that Narain et al. exploit in their second-order version. These differences between the two algorithms are visualized in figure 4.1.

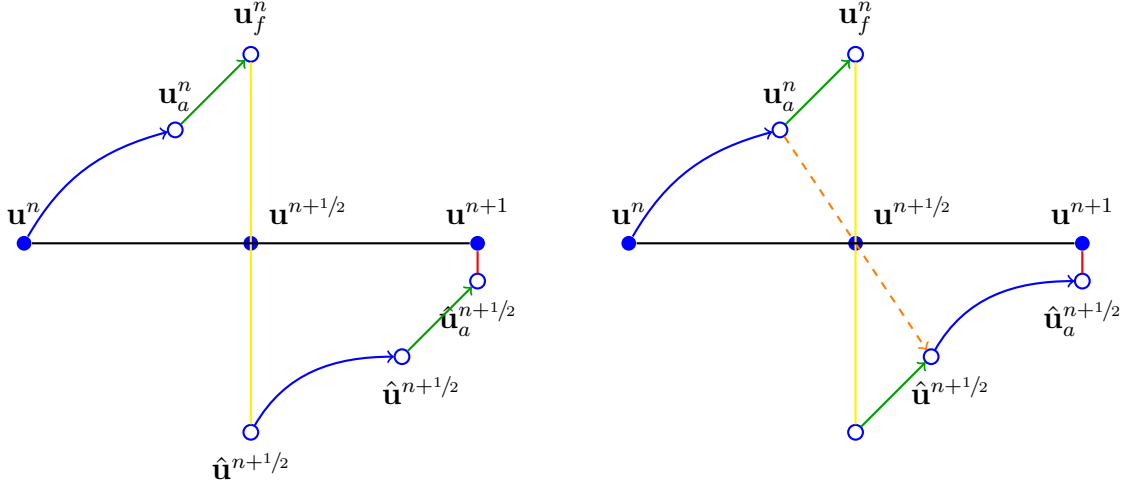


Figure 4.1: Each point represents a velocity field in $\mathfrak{X}(\Omega)$, with the black line representing $\text{Project}(\mathfrak{X}(\Omega))$, the subspace of divergence-free velocity fields. The blue curves represent advection steps, the green arrows represent the application of $\frac{1}{2}\mathbf{f}^{n+1/2}$, and the red lines represent energy loss due to projection onto $\text{Project}(\mathfrak{X}(\Omega))$. *Left*: First-order Advection-Reflection applies the midpoint force at the midpoint, but also the end point. *Right*: Second-order Advection-Reflection applies the midpoint force only at the midpoint, and scales it correctly with reflection. (The red arrow represents $2(\mathbf{u}^{n+1/2} - \mathbf{u}_a^n)$.) This figure is adapted from figure 2 of “A Second-Order Advection-Reflection Solver” [32].

4.3.2 Reflection for energy preservation

The original (first-order) Advection-Reflection method was devised as a strategy to more effectively preserve kinetic energy through a time step. The key part is the titular “Reflection” step in the algorithm. We discuss in this section the motivation for this step and the analysis presented by Zehnder et al. in the original paper [46].

We define the projection operator as the one removing the divergent component of a

velocity field:

$$\begin{aligned}\mathbf{v} &= \text{Project}(\mathbf{u}) \\ &\Updownarrow \\ \mathbf{v} &= \mathbf{u} - \frac{\Delta t}{\rho} \nabla p \text{ where } p = \text{Pressure}(\mathbf{u})\end{aligned}$$

Notice that $\text{Project}(\cdot)$ is linear, and that $\text{Project}(\text{Project}(\mathbf{u})) = \text{Project}(\mathbf{u}) = \mathbf{v}$, since $\mathbf{v}$ is divergence-free. These facts together mean that Project is an orthogonal projection [4]. As a result, $\mathbf{u} - \mathbf{v} \in \text{Project}(\mathfrak{X}(\Omega))^\perp$, the orthogonal complement of the subspace $\text{Project}(\mathfrak{X}(\Omega))$ of divergence-free flows. We can bound $\|\mathbf{v}\|^2$ using this orthogonality:

$$\begin{aligned}\|\mathbf{u}\|^2 &= \|\mathbf{v} + (\mathbf{u} - \mathbf{v})\|^2 \\ &= \|\mathbf{v}\|^2 + \|\mathbf{u} - \mathbf{v}\|^2 && \text{(orthogonality)} \\ \|\mathbf{u}\|^2 &\geq \|\mathbf{v}\|^2\end{aligned}$$

This is significant, as the kinetic energy of a fluid flow $\mathbf{w}$ is $\frac{1}{2}\rho\|\mathbf{w}\|^2$. What this shows us is that the projection operator decreases the kinetic energy of a fluid flow.

Next, we define the reflection operator as the one reflecting a velocity field across the hyperplane $\text{Project}(\mathfrak{X}(\Omega))$ of divergence-free flows:

$$\mathbf{w} = \text{Reflect}(\mathbf{u}) \tag{4.11}$$

$$\Updownarrow \tag{4.12}$$

$$\mathbf{w} = 2\text{Project}(\mathbf{u}) - \mathbf{u} = 2\mathbf{v} - \mathbf{u} \tag{4.13}$$

Taking the squared norm:

$$\begin{aligned}\|\mathbf{w}\|^2 &= \|2\mathbf{v} - \mathbf{u}\|^2 \\ &= \|\mathbf{v} + (\mathbf{v} - \mathbf{u})\|^2 \\ &= \|\mathbf{v}\|^2 + \|\mathbf{v} - \mathbf{u}\|^2 && \text{(orthogonality)} \\ &= \|\mathbf{v}\|^2 + \|\mathbf{u} - \mathbf{v}\|^2 \\ &= \|\mathbf{u}\|^2\end{aligned}$$

Thus, $\|\text{Reflect}(\mathbf{u})\| = \|\mathbf{u}\|$, so analogously to before, the reflection operator preserves the kinetic energy of a fluid flow. This illustrates why using Advection-Reflection preserves more energy than simply taking smaller advection-projection steps: No energy is lost at the midpoint. A visual representation of this property is shown in figure 4.2.

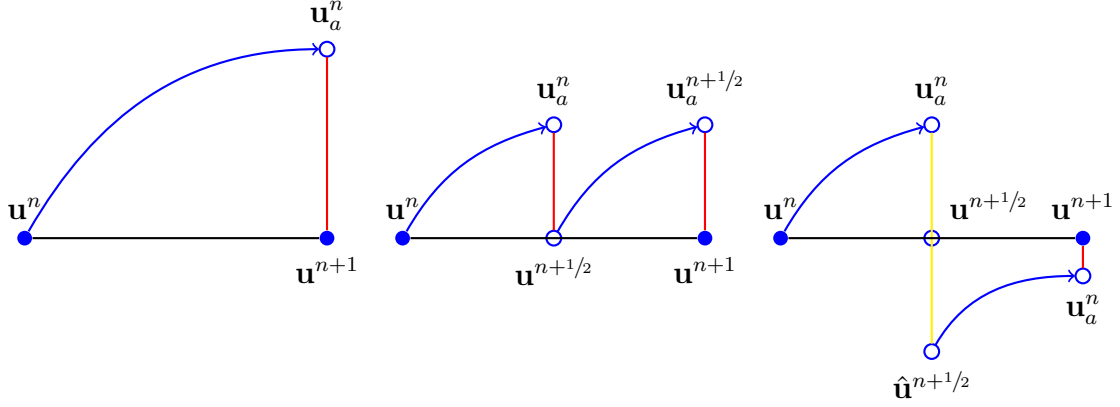


Figure 4.2: These diagrams are similar to those in figure 4.1, so once again red lines represent kinetic energy loss due to projection onto $\text{Project}(\mathfrak{X}(\Omega))$, the subspace of divergence-free velocity fields. *Left:* Standard advection-projection incurs a large amount of energy loss. *Middle:* Two advection-projection steps each advancing by $\frac{1}{2}\Delta t$ require twice as much computation. *Right:* Advection-Reflection has the same computational cost as two advection-projection steps, but incurs less energy loss. This figure is adapted from figure 2 of “An advection-reflection solver for detail-preserving fluid simulation” [46].

Furthermore, Zehnder et al. provide a proof that their first-order method loses energy with $\mathcal{O}(\Delta t^4)$ while the standard Advection-Projection method loses energy with $\mathcal{O}(\Delta t^2)$. However, this proof does not conclusively show that the energy preservation properties of Advection-Reflection are superior. We can do a similar analysis for BDF2. Consider the pre-projection velocity in BDF2 (without external forces, as in the work of Zehnder et al.):

$$\tilde{\mathbf{u}}^{n+1} = \frac{4}{3}\mathbf{u}_a^n - \frac{1}{3}\mathbf{u}_a^{n-1}$$

Using the first-order Taylor expansion of the advection operators, we find:

$$\begin{aligned} \tilde{\mathbf{u}}^{n+1} &= \frac{4}{3} \left(\mathbf{u}^n - \Delta t \left(\left(\frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1} \right) \cdot \nabla \right) \mathbf{u}^n + \mathcal{O}(\Delta t^2) \right) \\ &\quad - \frac{1}{3} \left(\mathbf{u}^{n-1} - 2\Delta t (\mathbf{u}^n \cdot \nabla) \mathbf{u}^{n-1} + \mathcal{O}(\Delta t^2) \right) \end{aligned}$$

Taking the divergence of both sides and recalling that $\mathbf{u}^n$ and $\mathbf{u}^{n-1}$ are divergence-free, we find:

$$\nabla \cdot \tilde{\mathbf{u}}^{n+1} = \Delta t \nabla \cdot \left(-(2\mathbf{u}^n \cdot \nabla) \mathbf{u}^n + \frac{2}{3}(\mathbf{u}^{n-1} \cdot \nabla) \mathbf{u}^n + \frac{2}{3}(\mathbf{u}^n \cdot \nabla) \mathbf{u}^{n-1} \right) + \mathcal{O}(\Delta t^2)$$

Thus, the divergence of $\tilde{\mathbf{u}}^{n+1}$ is $\mathcal{O}(\Delta t)$, so by Lemma 1 of [46], BDF2 loses energy with $\mathcal{O}(\Delta t^2)$, just as standard Advection-Projection does. However, Narain et al. [32] show experimentally that BDF2 has similar energy preservation qualities to Advection-Reflection. Thus, the asymptotic order of energy loss is an inconclusive measure of energy preservation for this family of fluid time stepping methods.

Chapter 5

Coupled temporal discretization

In this chapter we discuss temporal discretizations of the entire coupled fluid-rigid body system.

In general, the interior of the fluid domain can be treated as a fluid system with moving boundaries. Thus the fluid equations can be discretized for a semi-Lagrangian “particle” in the interior just as before, and so the same temporal discretizations of the fluid system carry over. In this chapter we discuss how these discretizations can be *augmented* to simultaneously discretize the motion of coupled rigid bodies.

In this chapter we propose our own time stepping schemes for coupled systems:

- In section 5.2.2 we present a coupled BDF2 method generalizing that of Xiu and Karniadakis [45] which simplifies the BDF2 method of Gibou and Min [19] discussed in section 5.2,
- In section 5.3 we present a coupled implicit midpoint method generalizing the second-order Advection-Reflection method of Narain et al. [32].

5.1 Coupled Implicit Euler

We would like to apply the Implicit Euler time discretization to the entire coupled fluid-rigid body system. To do this, we apply the discretization to the fluid interior, using the scheme developed in section 4.1, and construct a rigid body update scheme to match.

Discretizing the solid equations using Implicit Euler, we get:

$$\frac{\mathbf{T}_k^{n+1} - \mathbf{T}_k^n}{\Delta t} = \boldsymbol{\nu}_k^{n+1} \quad (5.1)$$

$$\frac{\boldsymbol{\nu}_k^{n+1} - \boldsymbol{\nu}_k^n}{\Delta t} = \mathcal{M}_k^{-1} \left(\mathcal{F}_k^{n+1} + \mathcal{J}_k^{n+1}(p^{n+1}) \right) \quad (5.2)$$

Note that the change in $\boldsymbol{\nu}_k$ relies on $\mathcal{J}_k^{n+1}$, which depends on the rigid transformation $\mathbf{T}_k = (\mathbf{X}_k, \mathbf{R}_k)$ at time t^{n+1} . However, unlike the fluid, there is no semi-Lagrangian viewpoint we can take to choose a final transformation $\mathbf{T}_k^{n+1}$ and backtrace dynamics through $\boldsymbol{\nu}_k^n$ to discover our known $\mathbf{T}_k^n$ (without an expensive and non-convex optimization procedure). Therefore, we must instead make our scheme semi-implicit, computing $\mathbf{T}_k^{n+1}$ using an explicit Euler step:

$$\frac{\mathbf{T}_k^{n+1} - \mathbf{T}_k^n}{\Delta t} = \boldsymbol{\nu}_k^n.$$

This is analogous to advecting the fluid velocity $\mathbf{u}^n$ through the frozen velocity field $\mathbf{u}^n$, as in both cases material quantities are being transported by the beginning-of-step velocity.

The new rigid body velocity itself must be compatible with the fluid velocity, in that the total system must solve the coupling PDE (3.1) which requires non-penetration. As discussed in section 5, this can be solved using a pressure projection. This yields a final method of:

	fluid	solids
1	$\mathbf{u}_a^n \leftarrow \text{Advect}(\mathbf{u}^n \text{ through } \mathbf{u}^n, \Delta t)$	$\mathbf{T}_k^{n+1} \leftarrow \mathbf{T}_k^n + \Delta t \boldsymbol{\nu}_k^n$
2	$\mathbf{u}_f^{n+1} \leftarrow \mathbf{u}_a^n + \frac{\Delta t}{\rho} \mathbf{f}^{n+1}$	$(\boldsymbol{\nu}_k^{n+1})_f \leftarrow \boldsymbol{\nu}_k^n + \Delta t \mathcal{M}_k^{-1} \mathcal{F}_k^{n+1}$
3	$p^{n+1} \leftarrow \text{Pressure}(\mathbf{u}_f^{n+1}, \{(\boldsymbol{\nu}_k^{n+1})_f\}_k)$	
4	$\mathbf{u}^{n+1} \leftarrow \mathbf{u}_f^{n+1} - \frac{\Delta t}{\rho} \nabla p^{n+1}$	$\boldsymbol{\nu}_k^{n+1} \leftarrow (\boldsymbol{\nu}_k^{n+1})_f + \Delta t \mathcal{M}_k^{-1} \mathcal{J}_k^{n+1}(p^{n+1})$

We have recovered the time stepping method suggested by Batty et al. [6] when originally presenting this coupling framework.

5.2 Coupled BDF2

5.2.1 Method of Gibou and Min

We discretize the interior of the fluid using the method of Gibou and Min [19] described earlier. For the solids, the BDF2 ODE scheme suggests the update:

$$\frac{\frac{3}{2}\mathbf{T}_k^{n+1} - 2\mathbf{T}_k^n + \frac{1}{2}\mathbf{T}_k^{n-1}}{\Delta t} = \boldsymbol{\nu}_k^{n+1} \quad (5.3)$$

$$\frac{\frac{3}{2}\boldsymbol{\nu}_k^{n+1} - 2\boldsymbol{\nu}_k^n + \frac{1}{2}\boldsymbol{\nu}_k^{n-1}}{\Delta t} = \mathcal{M}_k^{-1} \left(\mathcal{F}_k^{n+1} + \mathcal{J}_k^{n+1}(p^{n+1}) \right) \quad (5.4)$$

Gibou and Min [19] solve the problem of interdependency between $\mathbf{T}_k^{n+1}$ and $\boldsymbol{\nu}_k^{n+1}$ by implementing the implicit-explicit SBDF2 scheme discussed by Ascher et al. [3]. This involves computing an estimate of $\boldsymbol{\nu}_k^{n+1}$ to use in updating the rigid body transformation:

$$\begin{aligned} \boldsymbol{\nu}_k^{n+1} &\approx \boldsymbol{\nu}_k^n + \Delta t \boldsymbol{\nu}_k' \\ &\approx \boldsymbol{\nu}_k^n + \Delta t \frac{\boldsymbol{\nu}_k^n - \boldsymbol{\nu}_k^{n-1}}{\Delta t} \\ &= 2\boldsymbol{\nu}_k^n - \boldsymbol{\nu}_k^{n-1} \end{aligned}$$

Using this estimate, we can rewrite the (5.3) as:

$$\frac{\frac{3}{2}\mathbf{T}_k^{n+1} - 2\mathbf{T}_k^n + \frac{1}{2}\mathbf{T}_k^{n-1}}{\Delta t} = 2\boldsymbol{\nu}_k^n - \boldsymbol{\nu}_k^{n-1}$$

Similarly, in the velocity update (5.4), $\mathcal{F}_k^{n+1}$ is approximated as:

$$\mathcal{F}_k^{n+1} \approx 2\mathcal{F}_k^n - \mathcal{F}_k^{n-1}$$

We do not need to apply this treatment to the fluid-solid interaction force $\mathcal{J}_k^{n+1}(p^{n+1})$, because, as described in section 4.2.3, an explicit estimate of p^{n+1} is computed.

Combining these solid updates in the appropriate sequence yields the following final method:

	fluid	solids
1	$\mathbf{u}_a^{n-1} \leftarrow \text{Advect}(\mathbf{u}^{n-1} \text{ through } \mathbf{u}^n, 2\Delta t)$	$\mathbf{T}_k^{n+1} \leftarrow \frac{4}{3}\mathbf{T}_k^n - \frac{1}{3}\mathbf{T}_k^{n-1} + \frac{2\Delta t}{3}(2\boldsymbol{\nu}_k^n - \boldsymbol{\nu}_k^{n-1})$
2	$\mathbf{u}_a^n \leftarrow \text{Advect}(\mathbf{u}^n \text{ through } \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}, \Delta t)$	
3	$p_a^n \leftarrow \text{Advect}(p^n \text{ through } \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}, \Delta t)$	
4	$\mathbf{u}_f^{n+1} \leftarrow \frac{4}{3}\mathbf{u}_a^0 - \frac{1}{3}\mathbf{u}_a^{n-1} + \frac{2\Delta t}{3\rho}(\mathbf{f}^{n+1} - \nabla p_a^n)$	$(\boldsymbol{\nu}_k^{n+1})_f \leftarrow \frac{4}{3}\boldsymbol{\nu}_k^n - \frac{1}{3}\boldsymbol{\nu}_k^{n-1} + \frac{2\Delta t}{3}\mathcal{M}_k^{-1}(2\mathcal{F}_k^n - \mathcal{F}_k^{n-1} + \mathcal{J}_k^{n+1}(p^n))$
5	$q^{n+1} \leftarrow \frac{3}{2} \text{Pressure}(\mathbf{u}_f^{n+1}, \{(\boldsymbol{\nu}_k^{n+1})_f\}_k)$	
6	$\mathbf{u}^{n+1} \leftarrow \mathbf{u}_f^{n+1} - \frac{2\Delta t}{3\rho}\nabla q^{n+1}$	$\boldsymbol{\nu}_k^{n+1} \leftarrow (\boldsymbol{\nu}_k^{n+1})_f + \frac{2\Delta t}{3}\mathcal{M}_k^{-1}\mathcal{J}_k^{n+1}(q^{n+1})$
7	$p^{n+1} \leftarrow p_a^n + q^{n+1}$	

5.2.2 Our coupled BDF2 method

As discussed by Jiang et al. [25], accuracy and stability issues may arise from the advection and updating of a p variable, requiring treatment of a higher-order spatial discretization of p , as well as enforcement of extra boundary conditions [25]. To avoid these difficulties, we propose a method *without* the explicit pressure variable. In the interior of the fluid, we employ the BDF2 method of Xiu and Karniadakis [45] discussed in section 4.2.1. For rigid bodies, we use the method of Gibou and Min discussed above, but without the explicit pressure variable update, rather only with the end-of-step computed pressure. Our resulting method is:

	fluid	solids
1	$\mathbf{u}_a^{n-1} \leftarrow \text{Advect}(\mathbf{u}^{n-1} \text{ through } \mathbf{u}^n, 2\Delta t)$	$\mathbf{T}_k^{n+1} \leftarrow \frac{4}{3}\mathbf{T}_k^n - \frac{1}{3}\mathbf{T}_k^{n-1} + \frac{2\Delta t}{3}(2\boldsymbol{\nu}_k^n - \boldsymbol{\nu}_k^{n-1})$
2	$\mathbf{u}_a^n \leftarrow \text{Advect}(\mathbf{u}^n \text{ through } \frac{3}{2}\mathbf{u}^n - \frac{1}{2}\mathbf{u}^{n-1}, \Delta t)$	
3	$\mathbf{u}_f^{n+1} \leftarrow \frac{4}{3}\mathbf{u}_a^n - \frac{1}{3}\mathbf{u}_a^{n-1} + \frac{2\Delta t}{3\rho}\mathbf{f}^{n+1}$	$(\boldsymbol{\nu}_k^{n+1})_f \leftarrow \frac{4}{3}\boldsymbol{\nu}_k^n - \frac{1}{3}\boldsymbol{\nu}_k^{n-1} + \frac{2\Delta t}{3}\mathcal{M}_k^{-1}(2\mathcal{F}_k^n - \mathcal{F}_k^{n-1})$
4	$p^{n+1} \leftarrow \frac{3}{2} \text{Pressure}(\mathbf{u}_f^{n+1}, \{(\boldsymbol{\nu}_k^{n+1})_f\}_k)$	
5	$\mathbf{u}^{n+1} \leftarrow \mathbf{u}_f^{n+1} - \frac{2\Delta t}{3\rho}\nabla p^{n+1}$	$\boldsymbol{\nu}_k^{n+1} \leftarrow (\boldsymbol{\nu}_k^{n+1})_f + \frac{2\Delta t}{3}\mathcal{M}_k^{-1}\mathcal{J}_k^{n+1}(p^{n+1})$

5.3 Coupled Implicit Midpoint

We can analogously derive an implicit midpoint update for our coupled system by treating the interior of the fluid separately, and incorporating coupling as forcing for rigid bodies. Thus, the interior of the fluid can inherit the same second-order Advection-Reflection scheme that results from the Implicit Midpoint discretization. For rigid bodies, applying Implicit Midpoint yields the method:

$$\frac{\mathbf{T}_k^{n+1/2} - \mathbf{T}_k^n}{\frac{1}{2}\Delta t} = \boldsymbol{\nu}_k^{n+1/2} \quad (5.5)$$

$$\frac{\boldsymbol{\nu}_k^{n+1/2} - \boldsymbol{\nu}_k^n}{\frac{1}{2}\Delta t} = \mathcal{M}_k^{-1} \left(\mathcal{F}_k^{n+1/2} + \mathcal{J}_k^{n+1/2}(p^{n+1/2}) \right) \quad (5.6)$$

$$\mathbf{T}_k^{n+1} = \mathbf{T}_k^n + \Delta t \boldsymbol{\nu}_k^{n+1/2} \quad (5.7)$$

$$\boldsymbol{\nu}_k^{n+1} = 2\boldsymbol{\nu}_k^{n+1/2} - \boldsymbol{\nu}_k^n \quad (5.8)$$

However, this requires a few modifications. (5.5) and (5.6) are identical to an Implicit Euler step, so just as in section 5.1 we must replace (5.5) with an explicit step to estimate $\mathbf{T}_k^{n+1/2}$. Furthermore, just as in the fluid case, we must apply the influence of pressure at the end of the step to enforce our compatibility constraint, (3.1) The resulting rigid body updates, combined with second-order Advection-Reflection for the fluid interior, yield the time stepping method:

	fluid	solid
1	$\mathbf{u}_a^n \leftarrow \text{Advect}(\mathbf{u}^n \text{ through } \mathbf{u}^n, \frac{1}{2}\Delta t)$	$\mathbf{T}_k^{n+1/2} = \mathbf{T}_k^n + \frac{1}{2}\Delta t \boldsymbol{\nu}_k^n$
2	$\mathbf{u}_f^{n+1/2} \leftarrow \mathbf{u}_a^n + \frac{\Delta t}{2\rho} \mathbf{f}^{n+1/2}$	$(\boldsymbol{\nu}_k^{n+1/2})_f \leftarrow \boldsymbol{\nu}_k^n + \frac{1}{2}\Delta t \mathcal{M}_k^{-1} \mathcal{F}_k^{n+1/2}$
3	$p^{n+1} \leftarrow 2 \text{ Pressure}(\mathbf{u}_f^{n+1/2}, \{(\boldsymbol{\nu}_k^{n+1/2})_f\}_k)$	
4	$\mathbf{u}^{n+1/2} \leftarrow \mathbf{u}_f^{n+1/2} - \frac{\Delta t}{2\rho} \nabla p^{n+1/2}$	$\boldsymbol{\nu}_k^{n+1/2} = (\boldsymbol{\nu}_k^{n+1/2})_f + \frac{1}{2}\Delta t \mathcal{M}_k^{-1} \mathcal{J}_k^{n+1/2}(p^{n+1/2})$
5	$\hat{\mathbf{u}}^{n+1/2} \leftarrow 2\mathbf{u}^{n+1/2} - \mathbf{u}_a^n$	$\mathbf{T}_k^{n+1} = \mathbf{T}_k^n + \Delta t \boldsymbol{\nu}_k^{n+1/2}$
6	$\hat{\mathbf{u}}_a^{n+1/2} \leftarrow \text{Advect}(\hat{\mathbf{u}}^{n+1/2} \text{ through } 2\mathbf{u}^{n+1/2} - \mathbf{u}^n, \frac{1}{2}\Delta t)$	$\tilde{\boldsymbol{\nu}}_k^{n+1} = 2\boldsymbol{\nu}_k^{n+1/2} - \boldsymbol{\nu}_k^n$
7	$p^{n+1} \leftarrow \text{Pressure}(\hat{\mathbf{u}}_a^{n+1/2}, \{\tilde{\boldsymbol{\nu}}_k^{n+1}\}_k)$	
8	$\mathbf{u}^{n+1} \leftarrow \hat{\mathbf{u}}_a^{n+1/2} - \frac{\Delta t}{\rho} \nabla p^{n+1}$	$\boldsymbol{\nu}_k^{n+1} = \tilde{\boldsymbol{\nu}}_k^{n+1} + \Delta t \mathcal{M}_k^{-1} \mathcal{J}_k^{n+1}(p^{n+1})$

Chapter 6

Results

Throughout this section, we use the following abbreviations for our time stepping methods in our tables:

Abbreviation	Time stepping scheme
AP	Advection-Projection, aka Implicit Euler
AR	Advection-Reflection (first-order) [46]
AR2	Second-order Advection-Reflection [32], aka Implicit Midpoint
BDF2_XK	BDF2 scheme of Xiu and Karniadakis [45]
BDF2_GM	BDF2 scheme of Gibou and Min [19]
MA	Material Acceleration [35]
MA_imp	Our improved Material Acceleration

All results in this section are for 2D simulations. Correspondingly, some input quantities are computed using the 2D cross product and curls detailed in section ??.

In tables which list error E_i vs discretization size s_i , we show convergence order for each subsequent test. We compute this order as the slope of the log-log plot between the current point and the previous point:

$$\text{order}_i = \frac{\log E_i - \log E_{i-1}}{\log s_i - \log s_{i-1}} = \frac{\log \left(\frac{E_i}{E_{i-1}} \right)}{\log \left(\frac{s_i}{s_{i-1}} \right)}$$

6.1 Pressure projection convergence

To verify our implementation of the method of Batty et al. [6], we conduct convergence studies of the pressure projection step, measuring how its error changes with resolution. For each study we set the fluid density to $\rho = 1$.

6.1.1 Taylor-Green vortex

We start with static boundaries. Our first example is a box $\Omega = [0, \pi]^2$ of fluid with velocity field according to the Taylor-Green vortex, with a pressure gradient added over time $\Delta t = 1$ to make the field divergent. The pressure projection step should remove the divergent component of an input velocity field, so our expected result is the Taylor-Green vortex *without* the added pressure gradient.

$$\begin{aligned}\mathbf{u}_{\text{final}} &= \begin{pmatrix} \sin(x) \cos(y) \\ -\cos(x) \sin(y) \end{pmatrix} = \nabla \times (\sin(x) \sin(y)) \\ p &= x^4 y^3 \\ \mathbf{u}_{\text{input}} &= \mathbf{u}_{\text{final}} + \frac{\Delta t}{\rho} \nabla p \\ &= \begin{pmatrix} \sin(x) \cos(y) + 4x^3 y^3 \\ -\cos(x) \sin(y) + 3x^4 y^2 \end{pmatrix}\end{aligned}\tag{6.1}$$

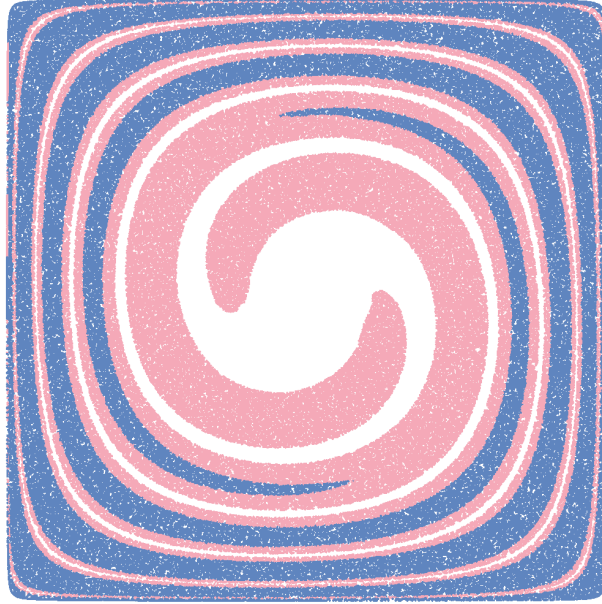


Figure 6.1: Visualization of Taylor-Green vortex. Coloured particles are passive markers for visualization.

Refining spatially, we observe second-order convergence of the velocity field in both the L^1 and L^∞ norms. This is consistent with our expectations, as all boundaries are axis-aligned, and all derivative approximations use ideal second-order centred differencing.

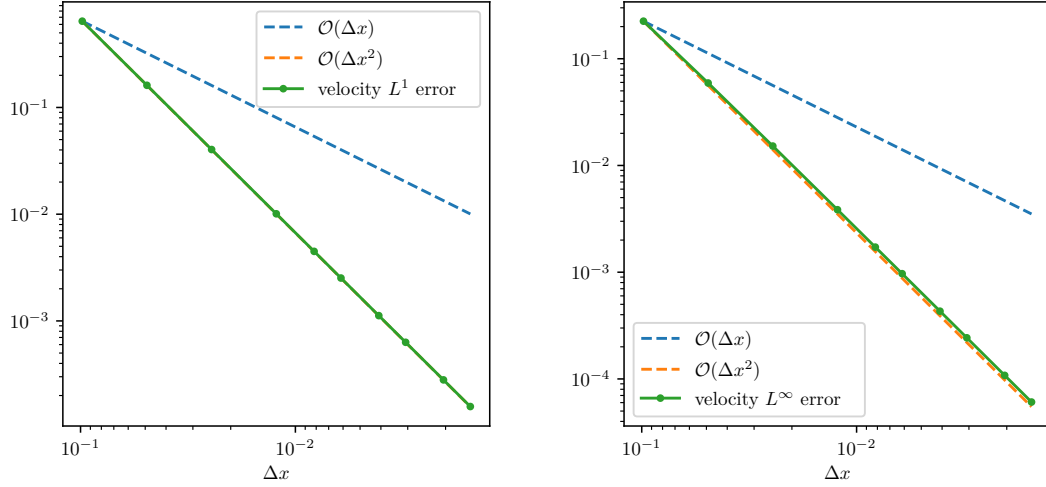


Figure 6.2: L^1 (left) and L^∞ (right) error vs grid cell spacing Δx for a pressure projection of the Taylor-Green vortex, with a pressure field of $p = x^4 y^3$ added.

Δx	L^1 error	order	L^∞ error	order
0.0981748	0.643		0.225	
0.0490874	0.161283	2.0	0.0592574	1.92
0.0245437	0.0403498	2.0	0.0152001	1.96
0.0122718	0.0100893	2.0	0.00384898	1.98
0.00818123	0.00448426	2.0	0.00171794	1.99
0.00613592	0.00252243	2.0	0.00096839	1.99
0.00409062	0.0011211	2.0	0.000431303	1.99
0.00306796	0.000630629	2.0	0.000242856	2.0
0.00204531	0.000280312	2.0	0.000108025	2.0
0.00153398	0.000157723	2.0	6.07581e-05	2.0

6.1.2 Circular domain

Next we borrow a test case from Batty [5], using a circular domain of fluid centred at the origin with radius $r = 1$. We add a pressure gradient over time $\Delta t = 1$ to a circular flow (i.e., a rigid rotation about the origin) as our input conditions, and expect the circular flow

as our result.

$$\begin{aligned}
 p &= xy^3 \\
 \mathbf{u}_{\text{final}} &= \begin{pmatrix} y \\ -x \end{pmatrix} \\
 \mathbf{u}_{\text{input}} &= \mathbf{u}_{\text{final}} + \frac{\Delta t}{\rho} \nabla p \\
 &= \begin{pmatrix} y + y^3 \\ -x + 3xy^2 \end{pmatrix}
 \end{aligned}$$

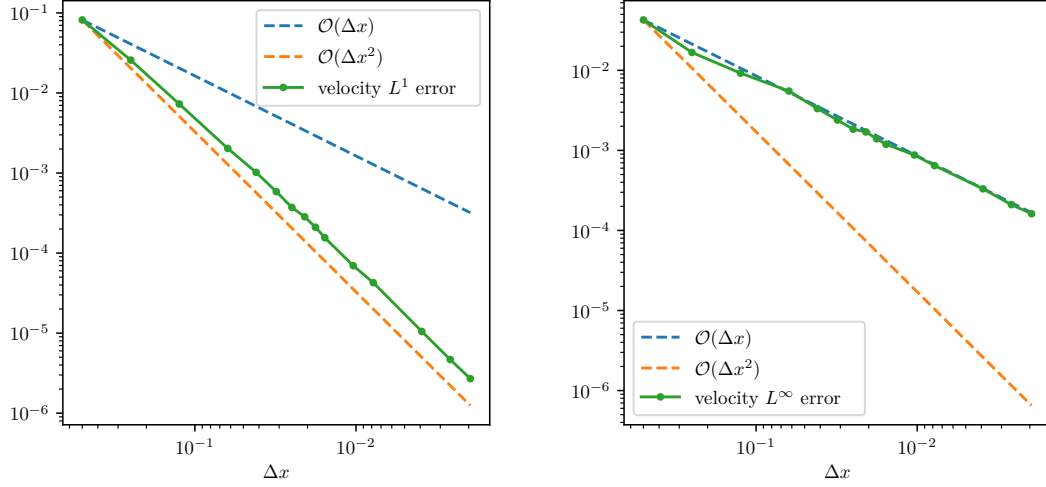


Figure 6.3: L^1 (left) and L^∞ (right) velocity error vs grid cell spacing Δx for a pressure projection of a circular flow in a circular domain, with a pressure field of $p = xy^3$ added.

Δx	L^1 error	order	L^∞ error	order
0.5	0.0817		0.0426	
0.25	0.0257208	1.67	0.0167238	1.35
0.125	0.00728152	1.82	0.0092444	0.855
0.0625	0.00202841	1.84	0.00550821	0.747
0.0416667	0.00101822	1.7	0.00332914	1.24
0.03125	0.000585956	1.92	0.00239453	1.15
0.025	0.000371342	2.04	0.00184037	1.18
0.0208333	0.000283868	1.47	0.00170165	0.43
0.0178571	0.000210361	1.94	0.00139661	1.28
0.015625	0.000155554	2.26	0.00119441	1.17
0.0104167	6.96028e-05	1.98	0.000876442	0.763
0.0078125	4.26807e-05	1.7	0.000647065	1.05
0.00390625	1.04953e-05	2.02	0.000331062	0.967
0.00260417	4.6776e-06	1.99	0.00021042	1.12
0.00195312	2.7005e-06	1.91	0.000161775	0.914

We measure the L^1 and L^∞ error as Δx decreases. Where Batty finds first-order convergence in the L^1 norm and no convergence (but bounded error) in L^∞ , we instead find approximately second-order and first-order convergence, respectively. This improvement is likely due to our adoption of the finite-volume inspired cell face “fluxes” used in discretizing the non-grid aligned domain. Discretization of the circular domain is also likely why we observe first-order convergence in the L^∞ error, where the previous example (with grid-aligned boundaries) showed second-order.

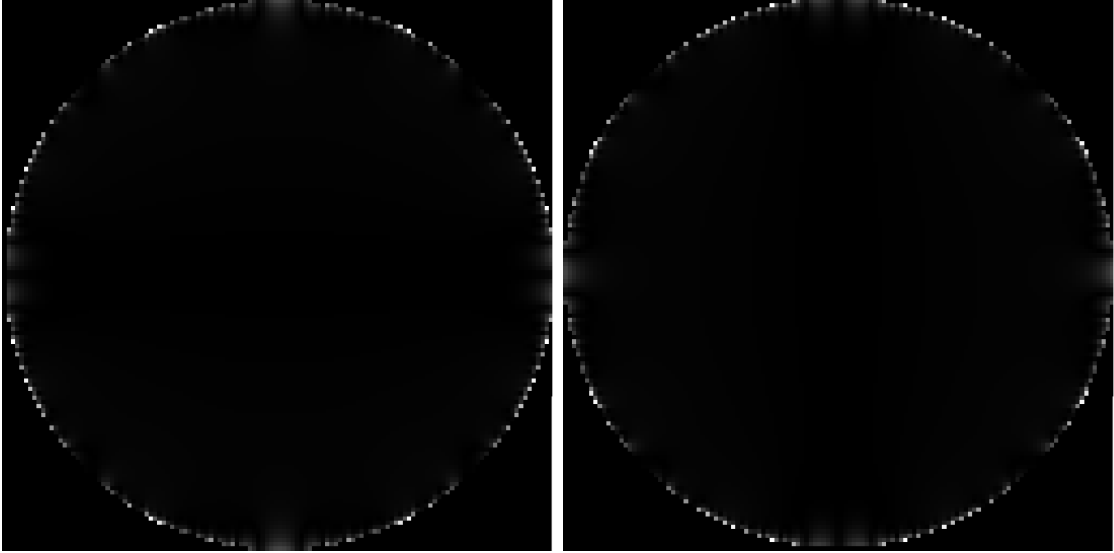


Figure 6.4: x (left) and y (right) velocity error over the fluid domain for test from Batty [5] with a grid resolution of 128×128 , i.e. $\Delta x = 0.015625$. The points with highest error are along the boundary of the fluid.

6.1.3 Ellipse in box

We test the pressure projection step for an asymmetric domain that is not grid-aligned. Our domain is a 1×1 box containing a solid ellipse E defined by the boundary

$$\partial E = \left\{ \mathbf{X} + 0.1 \begin{pmatrix} \cos(\theta) \\ 1.5 \sin(\theta) \end{pmatrix} : \theta \in [0, 2\pi] \right\}$$

with density ∞ , position $\mathbf{X} = (0.3, 0.4)$ and rotation $\mathbf{R} = \mathbf{I}$, and with velocities $\mathbf{V}, \omega$ to be chosen later. Thus the fluid domain is $\Omega = [0, 1]^2 \setminus E$. We construct a solution velocity that satisfies our constraints using the method proposed by Bridson et al. [9] with the modifications of Chang et al. [12]. This is a method to construct a divergence-free flow that respects no-normal flow boundary conditions for arbitrary boundary geometry, by first constructing an appropriate stream function ϕ . We take as our base the Taylor-Green vortex, which is generated by the stream function $\phi_f(x, y) = \sin(\pi x) \sin(\pi y)$, and use the above method to modify it to flow “around” E , using a modification radius constant of

$d_E = 0.01$:

$$\phi_f(x, y) = \sin(\pi x) \sin(\pi y)$$

$$\phi_g(\mathbf{x}) = \mathbf{V} \times (\mathbf{x} - \mathbf{X}) + \frac{0.1^2 - \|\mathbf{x} - \mathbf{X}\|^2}{2} \omega + \phi_f(\mathbf{X})$$

$$\phi(\mathbf{x}) = \phi_f(\mathbf{x}) + (\phi_g(\mathbf{x}) - \phi_f(\text{cp}_E(\mathbf{x}))) \left(1 - \text{ramp} \left(\frac{\text{sdf}_E(\mathbf{x})}{d_E} \right) \right)$$

$$\mathbf{u}_{\text{final}} = \nabla \times \phi$$

$$p(x, y) = x^2 y^2$$

$$\mathbf{u}_{\text{input}} = \mathbf{u}_{\text{final}} + \nabla p$$

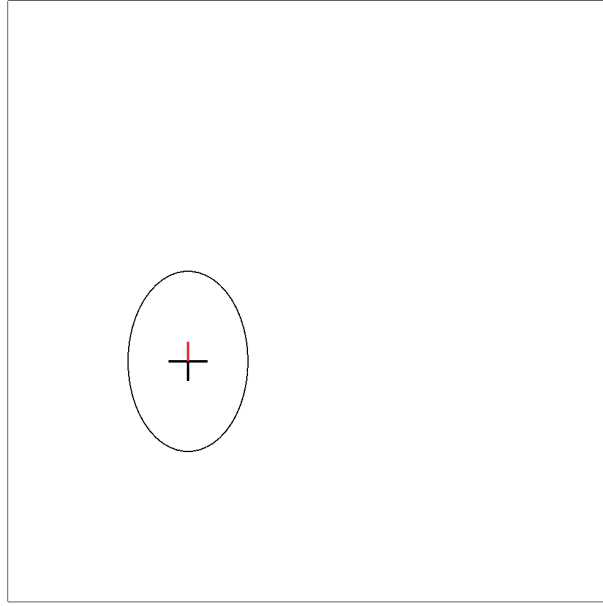


Figure 6.5: Visualization of the simulation domain for our “ellipse in box” test cases. The ellipse is a rigid body with a prescribed velocity.

We first use $\mathbf{V} = \mathbf{0}$ and $\omega = 0$, observing convergence with static boundaries.

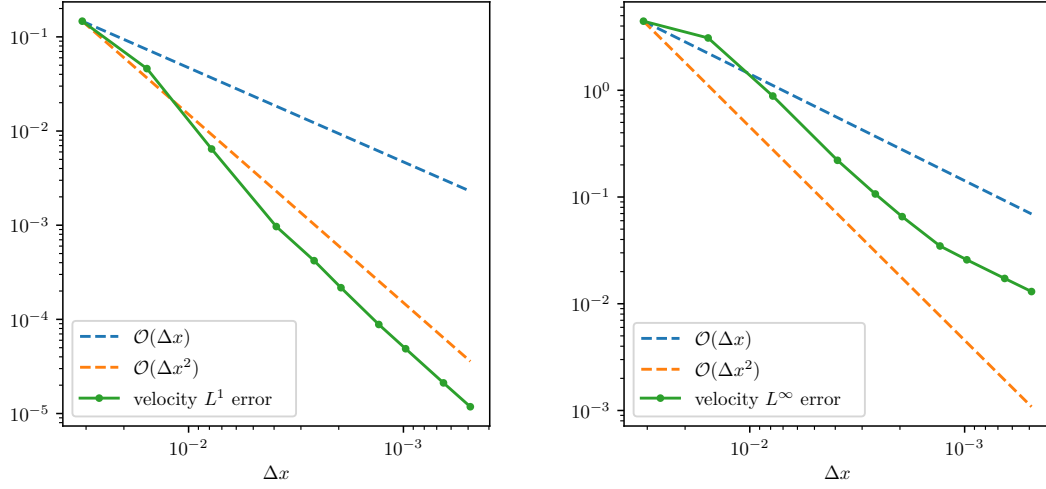


Figure 6.6: L^1 (left) and L^∞ (right) velocity error vs grid cell spacing Δx for a pressure projection with irregular static boundaries.

Δx	L^1 error	order	L^∞ error	order
0.03125	0.147		4.44	
0.015625	0.046035	1.67	3.09176	0.522
0.0078125	0.00644761	2.84	0.883573	1.81
0.00390625	0.000967578	2.74	0.220737	2.0
0.00260417	0.000420072	2.06	0.106778	1.79
0.00195312	0.000216953	2.3	0.0655021	1.7
0.00130208	8.82303e-05	2.22	0.0347666	1.56
0.000976562	4.87611e-05	2.06	0.0258049	1.04
0.000651042	2.1155e-05	2.06	0.0172603	0.992
0.000488281	1.18066e-05	2.03	0.0130096	0.983

We then give the ellipse motion, using $\mathbf{V} = (1, 1)$ and $\omega = 10$.

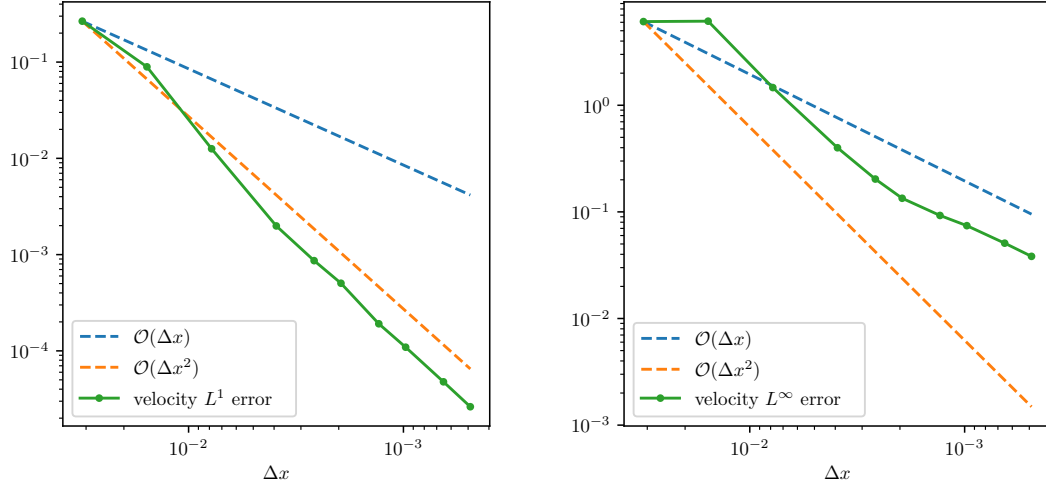


Figure 6.7: L^1 (left) and L^∞ (right) velocity error vs grid cell spacing Δx for a pressure projection with irregular moving boundaries.

Δx	L^1 error	order	L^∞ error	order
0.03125	0.266		6.09	
0.015625	0.0892659	1.58	6.14723	-0.0128
0.0078125	0.0125774	2.83	1.46518	2.07
0.00390625	0.00198613	2.66	0.399469	1.87
0.00260417	0.000865558	2.05	0.203205	1.67
0.00195312	0.000504157	1.88	0.134195	1.44
0.00130208	0.000191211	2.39	0.0925628	0.916
0.000976562	0.000109174	1.95	0.0742487	0.766
0.000651042	4.7702e-05	2.04	0.0508769	0.932
0.000488281	2.62832e-05	2.07	0.0382702	0.99

For both cases we observe near-second-order convergence in the L^1 norm and approximately first-order convergence in the L^∞ norm. This is consistent with our previous example borrowed from Batty, which has symmetric static boundaries and a simpler velocity field.

6.2 Time stepping convergence

Now we measure error in the velocity field after several time steps have been taken. For each study we use fluid with density $\rho = 1$.

6.2.1 Taylor-Green vortex

We first test a case with static square boundaries to verify our implementations of the various fluid time stepping schemes, and to evaluate our improved material acceleration scheme. Our initial velocity is the Taylor-Green vortex as described in (6.1), whose velocity field should remain constant over time. We measure the error in the velocity field at $t = 5$ for various time step sizes.

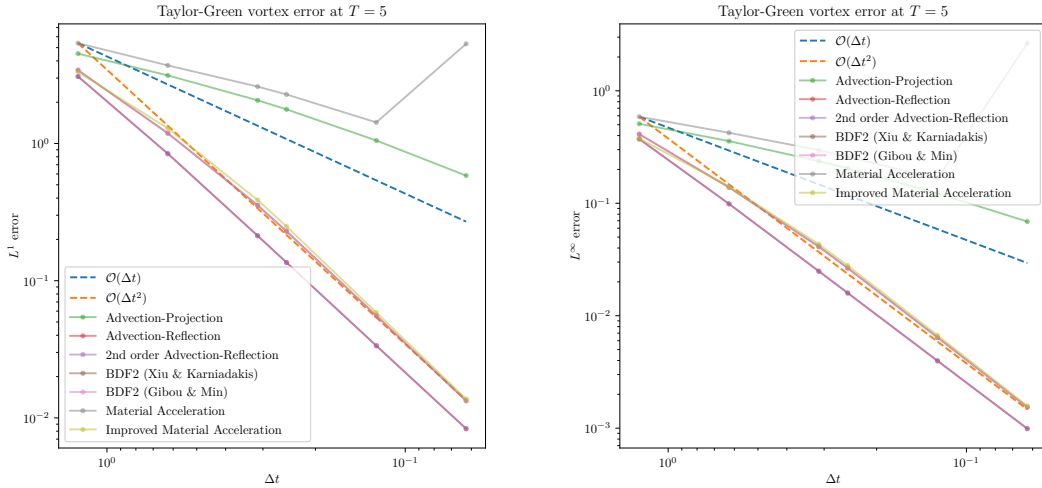


Figure 6.8: L^1 (left) and L^∞ velocity error vs time step size Δt for the Taylor-Green vortex at time $T = 5$. Note that the Advection-Reflection data and second-order Advection Reflection data coincide, as well as the two BDF2 methods.

Δt	AP	order	AR	order	AR2	order	BDF2_XK	order	BDF2_GM	order	MA	order	MA_imp	order
1.25	4.52		3.08		3.08		3.42		3.42		5.39		3.32	
0.625	3.14	0.528	0.843	1.87	0.843	1.87	1.19	1.53	1.19	1.53	3.71	0.539	1.29	1.36
0.3125	2.07	0.602	0.213	1.99	0.213	1.99	0.355	1.74	0.355	1.74	2.6	0.512	0.388	1.74
0.25	1.77	0.68	0.136	2.01	0.136	2.01	0.229	1.96	0.229	1.96	2.28	0.591	0.249	1.99
0.125	1.05	0.755	0.0336	2.01	0.0336	2.01	0.0556	2.04	0.0556	2.04	1.43	0.676	0.0587	2.08
0.0625	0.584	0.85	0.00833	2.01	0.00834	2.01	0.0133	2.06	0.0133	2.06	5.33	-1.9	0.0137	2.1

Table 6.1: L^1 velocity error of Taylor-Green vortex at $T = 5$ for various time step sizes, for each fluid-only time stepping scheme.

Δt	AP	order	AR	order	AR2	order	BDF2_XK	order	BDF2_GM	order	MA	order	MA_imp	order
1.25	0.509		0.373		0.373		0.412		0.412		0.588		0.377	
0.625	0.357	0.512	0.0987	1.92	0.0988	1.92	0.138	1.57	0.138	1.57	0.424	0.475	0.141	1.42
0.3125	0.237	0.592	0.0249	1.99	0.0248	1.99	0.0412	1.75	0.0412	1.75	0.297	0.511	0.0432	1.7
0.25	0.204	0.664	0.016	2.0	0.0159	2.0	0.0265	1.97	0.0265	1.97	0.259	0.62	0.0279	1.96
0.125	0.123	0.737	0.00398	2.0	0.00396	2.0	0.00641	2.05	0.00641	2.05	0.165	0.648	0.00668	2.06
0.0625	0.0687	0.833	0.000991	2.0	0.000989	2.0	0.00154	2.06	0.00154	2.06	2.65	-4.01	0.00158	2.08

Table 6.2: L^∞ velocity error of Taylor-Green vortex at $T = 5$ for various time step sizes, for each fluid-only time stepping scheme.

For both norms, we observe our expected convergence orders for each time stepping method: near-first-order for Advection-Projection (Stable Fluids implicit Euler), and second-order for the rest, except for Material Acceleration, which is on par with Advection-Projection. However, our improved Material Acceleration scheme meets the second-order accuracy claimed by Nielsen et al. [35] and closely matches the error of the Xiu and Karniadakis BDF2 scheme, suggesting that our update is indeed a correction. For this case in particular, we suspect that MA performs similarly to AP due to the nature of the example: the Taylor-Green vortex should (ideally) maintain a constant velocity field, so we expect MA's incorrect acceleration term $\mathbf{a}^{n+1/2} = \frac{1}{\Delta t}(\mathbf{u}^{n+1} - \mathbf{u}^n)$ to remain near 0. Without this term, the MA scheme is identical to AP. Conversely, our improved Material Acceleration scheme correctly estimates the acceleration of the material as $\mathbf{a}^{n+1/2} = \frac{1}{\Delta t}(\mathbf{u}^{n+1} - \mathbf{u}_a^n)$, and is thus able to more accurately simulate the curved motion of each point of the fluid.

6.2.2 Rigid body force

Continuing to investigate the components of our method separately, we next test the time integration of a rigid body not immersed in any fluid, with a time-varying external force. Thus we limit ourselves to simulating the rigid body equations, which amount to an ODE.

We use a circular rigid body with density $\rho = 1$, and we choose a simple external force

so that we have an analytical solution to our ODE:

$$\begin{aligned}
\mathbf{X}_{\text{input}}^0 &= \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\
\mathbf{V}_{\text{input}}^0 &= \begin{pmatrix} 1 \\ 1 \end{pmatrix} \\
\mathbf{f}_{\text{input}}(t) &= \mathbf{M} \begin{pmatrix} -\sin(t) \\ -Y(t) \end{pmatrix} \quad \left(\text{where } \mathbf{X}(t) = \begin{pmatrix} X(t) \\ Y(t) \end{pmatrix} \right) \\
\mathbf{X}_{\text{final}}(t) &= \begin{pmatrix} \sin(t) \\ \sin(t) \end{pmatrix} \\
\mathbf{V}_{\text{final}}(t) &= \begin{pmatrix} \cos(t) \\ \cos(t) \end{pmatrix}
\end{aligned}$$

We compute the L^2 error in the rigid body's final velocity, $\|\mathbf{V}(T) - \mathbf{V}_{\text{final}}(T)\|$ at $T = 30$ for each time stepping scheme. Note that in the absence of a fluid with a pressure field, the methods of Xiu and Karniadakis [45] and Gibou and Min [19] are identical, so we represent them by a single BDF2 scheme.

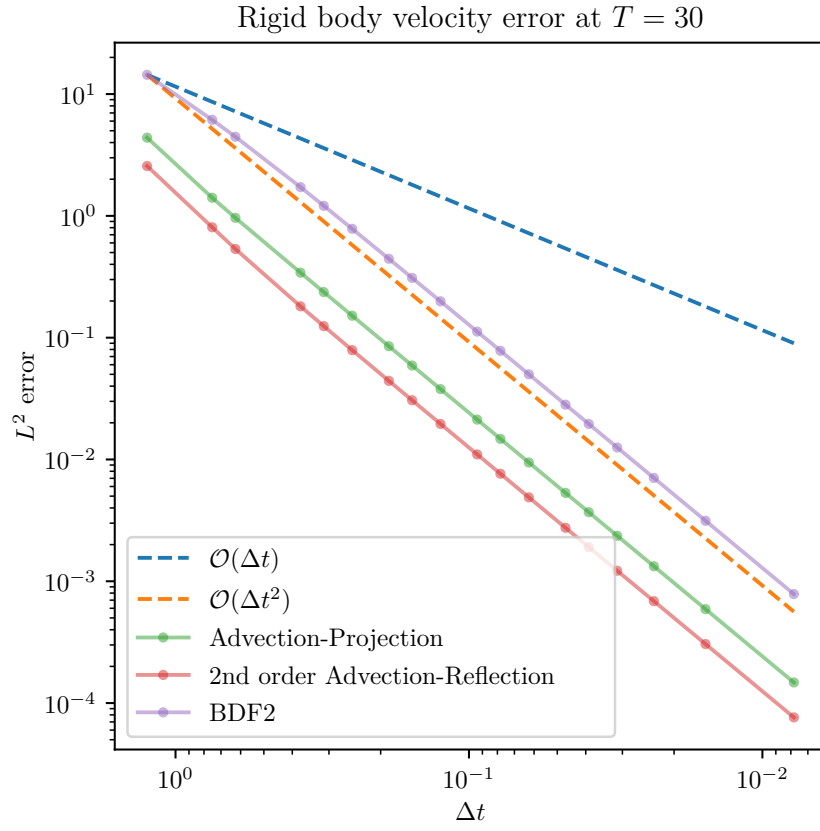


Figure 6.9: Rigid body velocity error at $T = 5$ for the rigid body component of each coupled time stepping method.

Δt	AP	order	AR2	order	BDF2	order
1.25	4.39		2.57		14.4	
0.75	1.41	2.22	0.806	2.27	6.14	1.67
0.625	0.964	2.08	0.535	2.25	4.46	1.76
0.375	0.342	2.03	0.181	2.12	1.72	1.87
0.3125	0.237	2.01	0.124	2.05	1.21	1.93
0.25	0.151	2.01	0.079	2.03	0.783	1.95
0.1875	0.0851	2.0	0.0442	2.02	0.444	1.97
0.15625	0.0591	2.0	0.0307	2.01	0.31	1.98
0.125	0.0378	2.0	0.0196	2.01	0.199	1.98
0.09375	0.0213	2.0	0.011	2.0	0.112	1.99
0.078125	0.0148	2.0	0.00764	2.0	0.078	1.99
0.0625	0.00945	2.0	0.00488	2.0	0.05	1.99
0.046875	0.00532	2.0	0.00275	2.0	0.0282	2.0
0.0390625	0.00369	2.0	0.00191	2.0	0.0196	2.0
0.03125	0.00236	2.0	0.00122	2.0	0.0125	2.0
0.0234375	0.00133	2.0	0.000687	2.0	0.00706	2.0
0.015625	0.00059	2.0	0.000305	2.0	0.00314	2.0
0.0078125	0.000148	2.0	7.63e-05	2.0	0.000785	2.0

Table 6.3: Rigid body velocity error at $T = 30$ for the rigid body component of each coupled time stepping method.

We find approximately first-order convergence for implicit Euler and second-order convergence for implicit midpoint and BDF2, as expected.

6.2.3 Circle in tube

For this example, we have a circular solid wall of radius 0.4 on the outside and a free-moving solid disk of radius 0.2 and density 1 in the centre. The initial velocity of the fluid is circular, and the disk has zero initial velocity. Both velocities are expected to remain as such. An animation of this scene is provided in the supplementary material.

$$\begin{aligned}
r_{\text{wall}} &= 0.4 \\
r_{\text{disk}} &= 0.2 \\
\mathbf{u} &= \begin{pmatrix} y \\ -x \end{pmatrix}
\end{aligned}$$

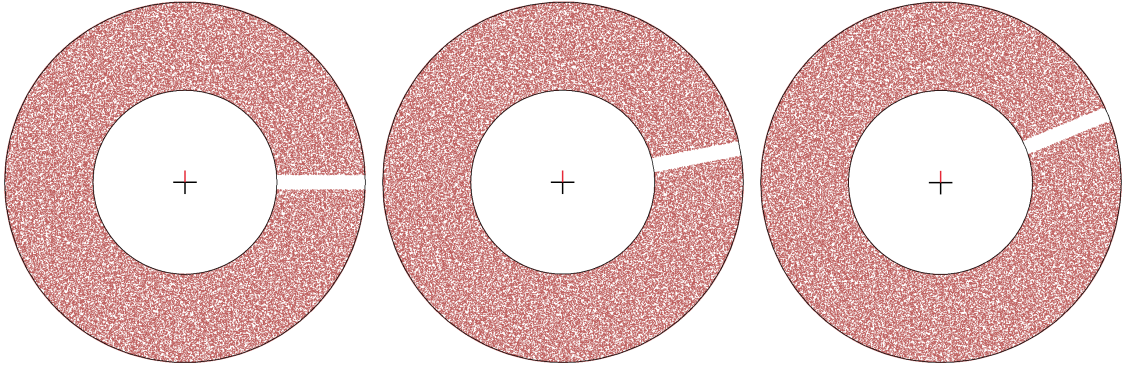


Figure 6.10: Positions of rigid bodies and marker particles at times $t = 0.01, 0.20, 0.40$ (left to right).

The error in the solid velocity was observed to remain around the order of 10^{-15} , aka numerically zero. We measure the error in the fluid velocity field at $t = 5$ for various time step sizes.

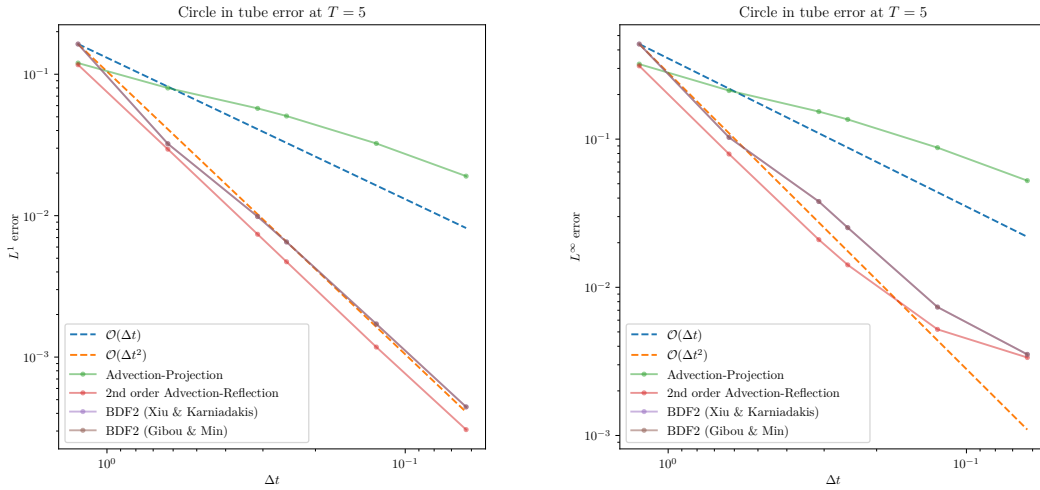


Figure 6.11: L^1 (left) and L^∞ (right) velocity error of circle-in-tube test case at time $T = 5$.

Δt	AP	order	AR2	order	BDF2_XK	order	BDF2_GM	order
1.25	0.12		0.117		0.163		0.163	
0.625	0.0799	0.586	0.0295	1.98	0.0323	2.34	0.0323	2.34
0.3125	0.0574	0.478	0.00739	2.0	0.00986	1.71	0.00986	1.71
0.25	0.0507	0.557	0.00472	2.01	0.00652	1.85	0.00652	1.85
0.125	0.0324	0.645	0.00118	2.0	0.00171	1.93	0.00171	1.93
0.0625	0.019	0.768	0.000307	1.94	0.000444	1.95	0.000444	1.95

Table 6.4: L^1 velocity error of circle-in-tube test case at $T = 5$ for various time step sizes, for each coupled time stepping scheme.

Δt	AP	order	AR2	order	BDF2_XK	order	BDF2_GM	order
1.25	0.321		0.312			0.438		0.438
0.625	0.213	0.593	0.0794	1.98		0.103	2.09	0.103 2.09
0.3125	0.153	0.471	0.021	1.92		0.0379	1.44	0.0379 1.44
0.25	0.136	0.546	0.0142	1.76		0.0253	1.82	0.0253 1.82
0.125	0.0876	0.633	0.0052	1.45		0.00736	1.78	0.00736 1.78
0.0625	0.0525	0.74	0.00336	0.631		0.00353	1.06	0.00353 1.06

Table 6.5: L^∞ velocity error of circle-in-tube test case at $T = 5$ for various time step sizes, for each coupled time stepping scheme.

We observe the same convergence orders for each method as in the previous study.

6.3 Energy preservation

6.3.1 Dual vortices

We borrow an example from Zehnder et al. [46], initializing our velocity field with two vortices of the same sign:

$$\begin{aligned}\Omega &= [0, 1]^2 \\ \mathbf{x}_1 &= (0.45, 0.5) \\ \mathbf{x}_2 &= (0.55, 0.5) \\ \lambda &= 24 \\ \phi(\mathbf{x}) &= \exp(-\lambda \|\mathbf{x} - \mathbf{x}_1\|) + \exp(-\lambda \|\mathbf{x} - \mathbf{x}_2\|) \\ \mathbf{u}^0 &= \nabla \times \phi\end{aligned}$$

We observe the motion of the initial marker particles over time, taking constant time steps of size $\Delta t = 0.001$, and using a spatial resolution of 128×128 . An animation is provided in the supplementary material.

As expected, we find that the two vortices merge when using the standard Advection-Projection scheme, and separate for more energetic schemes such as first-order Advection-Reflection, as well as for more accurate schemes such as the others discussed. They also merge when using the Material Acceleration scheme but not with our improved version, supporting our claim that the original does not discretize fluid trajectories as intended.

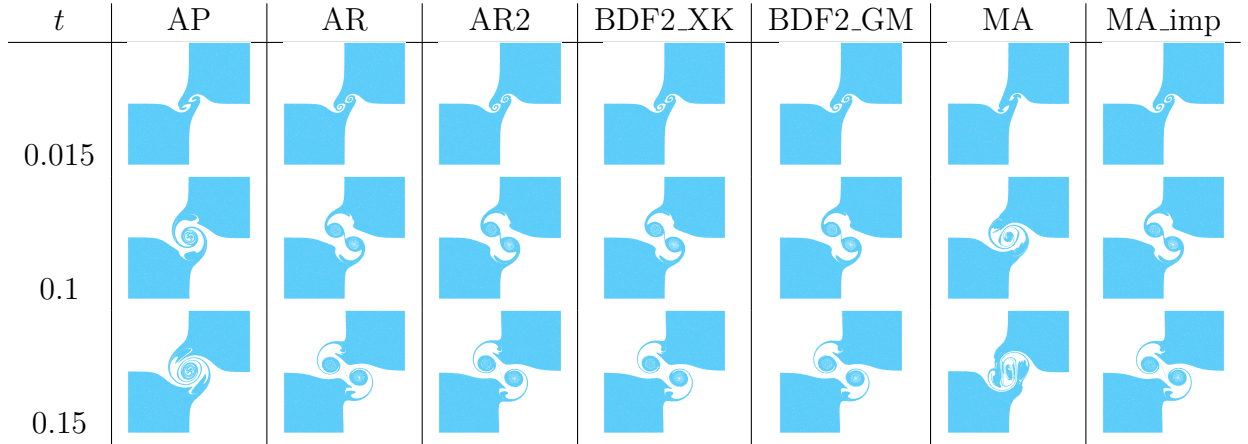


Figure 6.12: Dual vortex evolution for each fluid-only time stepping scheme. Vortices separate for each scheme except Advection-Projection and Material Acceleration.

6.3.2 Ellipse in box

We measure energy over time for an ellipse spinning inside a box of fluid, to demonstrate the energy preservation capabilities of our novel coupled Advection-Reflection scheme against the other coupling schemes. An animation of this scene is provided in the supplementary material.

For each time step, we measure the total energy of the system as the sum of kinetic energies of the fluid and the rigid body:

$$\begin{aligned}
 KE &= KE_f + KE_s \\
 &= \frac{1}{2} \mathbf{u}^\top \mathbf{M}_f \mathbf{u} + \frac{1}{2} \boldsymbol{\nu}^\top \mathcal{M} \boldsymbol{\nu} \\
 &= \frac{1}{2} \rho \left(\sum_{i,j} \text{vol}_{i+1/2,j}^f u_{i+1/2,j} + \sum_{i,j} \text{vol}_{i,j+1/2}^f v_{i,j+1/2} \right) + \frac{1}{2} \mathbf{V}^\top \mathbf{M} \mathbf{V} + \frac{1}{2} I \omega^2
 \end{aligned}$$

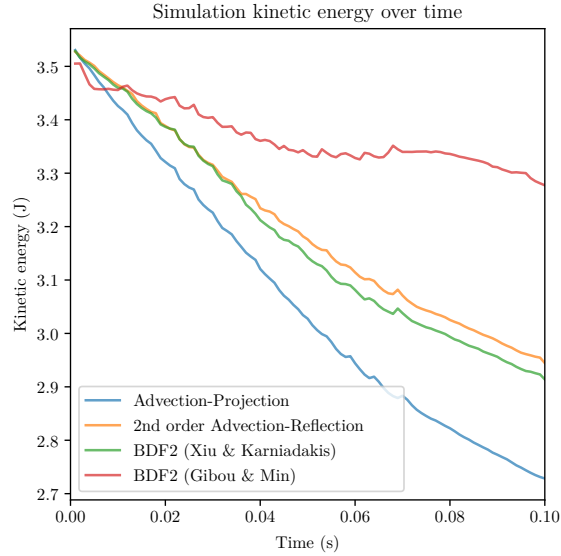


Figure 6.13: Total kinetic energy over time for ellipse spinning within square fluid domain.

We observe that energy decreases much more slowly for our BDF2 and second-order Advection-Reflection schemes than it does for the standard Advection-Projection scheme,

but all of these have generally stable and monotonic energies compared to the BDF2 method of Gibou and Min, which shows an increase in energy after an initial decrease. This scheme, like all others considered here, concludes with a pressure projection step to make the fluid velocity field divergence-free and to make the fluid and solid velocities compatible. As shown in section 4.3.2 and in the paper by Gibou and Min [19] itself, this step cannot cause an increase in kinetic energy. Furthermore, this BDF2 scheme differs from our modification of that of Xiu and Karniadakis only by the incremental pressure update, and Jiang et al. found that advection of an explicit pressure variable can introduce instability near the boundaries [25]. We suspect that this is the cause of the energy's non-monotonicity when using this scheme.

We can also see qualitative improvements of the other schemes over Advection-Projection in the form of increased vorticity and detail in the fluid flow after simulating for some time: See figure 6.14 and supplementary material.

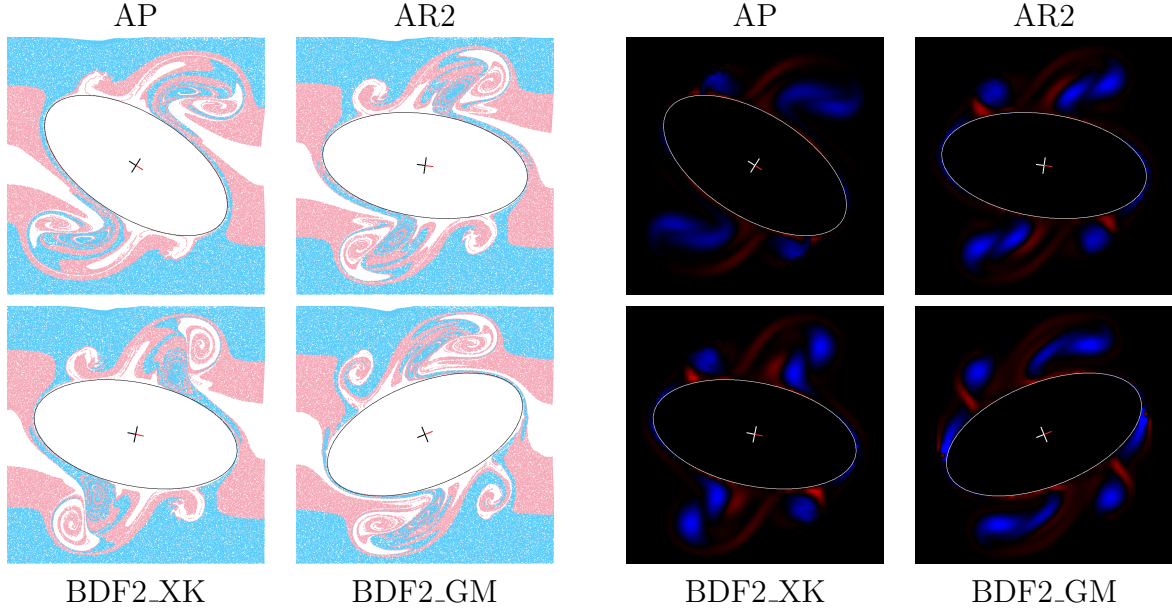


Figure 6.14: Marker particle positions (left) and vorticity (right) of fluid at time $t = 0.30$ for each coupled time stepping method. Schemes other than Advection-Projection result in more intricate vortical motion in the fluid, observable as higher values of vorticity.

6.4 Computational cost

We compare the run time of each time stepping method for two scenarios: the Taylor-Green vortex (described above) with no additional rigid bodies, and a scene with four submerged squares moving in different directions (see supplementary material). We measure total run time of the simulation averaged over 100 runs, on a *MacBook Air (M2, 2022)* with 8 GB of memory, using 8 threads for tasks parallelized using OpenMP version 4.5 [14].

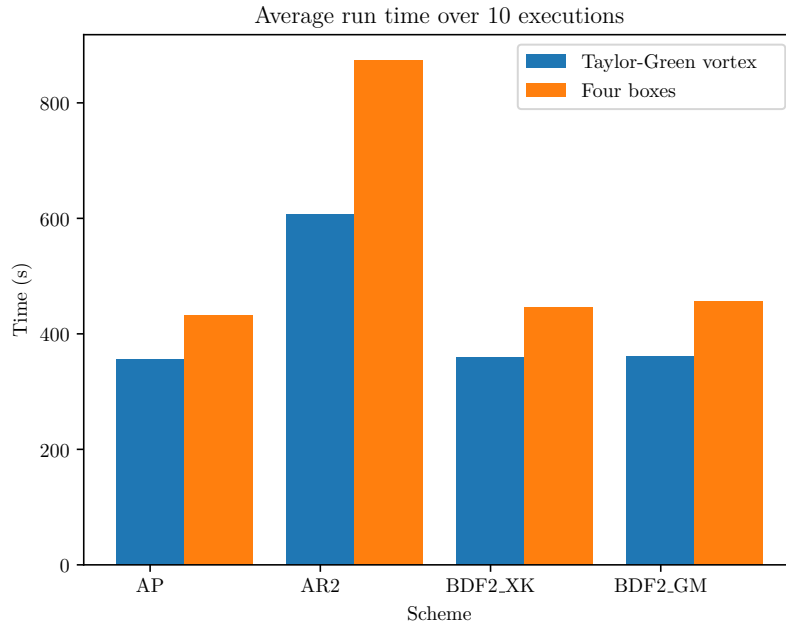


Figure 6.15: Running time of 50 time steps with each coupled scheme, for Taylor-Green vortex and four submerged boxes.

The implicit Euler and BDF2-based methods have similar run time, and the implicit midpoint-based method takes approximately twice as long. This suggests that the run time is dominated by the pressure projection step, as the former group each has one pressure projection with varying numbers of advections, while the latter scheme has two pressure projections. This bottleneck could be improved by adopting more specialized methods for solving the linear equation for pressure, such as the method of Aanjaneya [1] discussed previously.

Chapter 7

Conclusion

We have discussed fluid time stepping methods in the computer graphics literature, and developed corresponding time stepping methods for fluid-rigid body systems coupled using the framework of Batty et al. Furthermore, we have shown that second-order time stepping methods for the coupled system result in more visually enticing simulations. We now consider the shortcomings of our work and possible directions for future work.

7.1 Limitations

Given a fluid time stepping scheme, the methodology employed in this thesis cannot always suggest a corresponding coupled fluid-rigid body scheme. In our work, we first noticed that second-order Advection-Reflection was a realization of the Implicit Midpoint method for ODEs, and so applied Implicit Midpoint to the entire coupled system to devise the coupled time stepping scheme presented in section 5.3. We took a similar approach with the BDF2 method of Xiu and Karniadakis in 5.2.2. Our approach was to find the general ODE method that generates the given fluid time stepping scheme, then derive a coupled scheme by applying the general method to the coupled system. This approach is incompatible with fluid time stepping schemes such as (first-order) Advection-Reflection, which cannot be derived as a realization of any particular ODE method. In other words, we have not provided a generic method of translating any fluid scheme into a coupled scheme.

In an effort to maintain compatibility with the dominant framework of fluid simulation in the field, we have extended upon the coupling method of Batty et al. [6], which assumes steps are taken within an operator splitting framework. In pairing it with other fluid

time stepping schemes, one of our aims was to present a method which better preserves the energy of the system. However, our methods as formulated are not compatible with simulation methods such as that of Mullen et al. [29], which is able to preserve energy *exactly* through its avoidance of operator splitting. We have not addressed the fundamental problem of energy loss in operator splitting-based coupled simulation.

7.2 Future work

We have observed the results of a 2D solver, but fluid-rigid body simulation is typically done in three dimensions to model real-world scenarios. Implementation and verification of a 3D simulation should be straightforward, as the coupling method of Batty et al. is near-identical in 3D, and the time stepping methods we provide are agnostic of dimensionality. Furthermore, Nielsen et al. [35] discuss other aspects of a time stepping scheme that affect its practicality in production, such as compatibility of adaptive grid layouts between velocity fields, while we consider this out of our scope. Further work may investigate the applicability of our work to simulation codes used in industry and academia.

In this work we concerned ourselves with coupled time stepping methods corresponding to second-order schemes for fluids, which we derived from second-order ODE schemes. The same approach can be taken for higher-order ODE methods such as BDF3. The graphics community may be interested in the resulting qualitative improvements, and their tradeoffs with increased complexity and cost.

Our solver does not support free-surface boundary conditions, which are commonly used to model a liquid-air interface. Extending our simulation to support these boundaries would allow us to evaluate each time stepping scheme’s visual improvements to liquid surface phenomena such as splashing and buckling, which may result from liquid-rigid body interaction.

In section 6.2.3, we found second-order convergence of error for both of our proposed time stepping schemes with refinement of time step size, in an exceedingly simple scenario. However, the lack of complex test cases makes it uncertain whether our methods would achieve this accuracy in all scenarios; indeed they may not. Further investigation may look into which aspects of the simulation might need to be improved to achieve this “true” second-order accuracy. This may entail more accurate discretization of irregular domain shapes [19, 26], accurate extrapolation of values into the “solid” parts of the simulation domain [33], investigating the influence of numerical boundary layers near obstacles, and more.

Further in this spirit, future work focused on the development of a framework for measuring global convergence of time stepping methods for fluids would be vastly beneficial to the community. Our convergence results for the coupled system were measured for simple scenarios, with velocity fields that do not vary over time, as the analytical solutions to these scenarios are known. These simple scenarios do not reflect the complex and interesting scenes that computer graphics applications are interested in. Furthermore, lack of variation in time means that the time stepping methods’ ability to accurately estimate characteristic curves is not fully evaluated. Due to the complexity of the Incompressible Euler equations, it is difficult to find more sophisticated analytical solutions. With this in mind, the *method of manufactured solutions* (MMS) may provide a framework for measuring accuracy in more complex scenes, as this method does not require a “natural” analytical solution. MMS entails choosing a target function, and inputting a corresponding external force such that the simulation reproduces the target function. The only constraints on the target function are of adequate smoothness and of satisfying expected boundary conditions [37]. In the case of a coupled solver, any flow and rigid body kinematics can be chosen so long as they satisfy the no normal flow boundary condition. One potential method of generating such flows is the stream function approach taken in section 6.1.3. Use of MMS, or another method of generating test cases with known solutions for coupled systems, would be a significant improvement upon our testing method, and would allow confidence in the numerical accuracy, and thus realism, of coupled fluid-rigid body simulations.

References

- [1] Mridul Aanjaneya. “An Efficient Solver for Two-way Coupling Rigid Bodies with Incompressible Flow”. en. In: *Computer Graphics Forum* 37.8 (Dec. 2018), pp. 59–68. ISSN: 0167-7055, 1467-8659. DOI: [10.1111/cgf.13512](https://doi.org/10.1111/cgf.13512).
- [2] Uri M. Ascher and Linda R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. en. Philadelphia, PA: Society for Industrial and Applied Mathematics, Jan. 1998. ISBN: 978-0-89871-412-8. DOI: [10.1137/1.9781611971392](https://doi.org/10.1137/1.9781611971392).
- [3] Uri M. Ascher, Steven J. Ruuth, and Brian T. R. Wetton. “Implicit-Explicit Methods for Time-Dependent Partial Differential Equations”. en. In: *SIAM Journal on Numerical Analysis* 32.3 (1995), pp. 797–823.
- [4] Sheldon Axler. *Linear Algebra Done Right*. en. Undergraduate Texts in Mathematics. Cham: Springer International Publishing, 2015. ISBN: 978-3-319-11079-0. DOI: [10.1007/978-3-319-11080-6](https://doi.org/10.1007/978-3-319-11080-6).
- [5] Christopher Batty. “Simulating viscous incompressible fluids with embedded boundary finite difference methods”. en. Publisher: University of British Columbia Version Number: 1. PhD thesis. 2010.
- [6] Christopher Batty, Florence Bertails, and Rook Bridson. “A fast variational framework for accurate solid-fluid coupling”. In: *ACM SIGGRAPH 2007 papers*. SIGGRAPH ’07. New York, NY, USA: Association for Computing Machinery, July 2007, 100–es. ISBN: 978-1-4503-7836-9. DOI: [10.1145/1275808.1276502](https://doi.org/10.1145/1275808.1276502). URL: https://www.cs.ubc.ca/labs/imager/tr/2007/Batty_VariationalFluids/ (visited on 12/08/2025).
- [7] Harsh Bhatia et al. “The Helmholtz-Hodge Decomposition—A Survey”. In: *IEEE Transactions on Visualization and Computer Graphics* 19.8 (Aug. 2013), pp. 1386–1404. ISSN: 1941-0506. DOI: [10.1109/TVCG.2012.316](https://doi.org/10.1109/TVCG.2012.316).

- [8] Rook Bridson. *Fluid Simulation for Computer Graphics*. 2nd ed. A K Peters/CRC Press, Sept. 2015. URL: <https://www.cs.ubc.ca/~rbridson/fluidsimulation>.
- [9] Rook Bridson, Jim Houriham, and Marcus Nordenstam. “Curl-noise for procedural fluid flow”. en. In: *ACM Transactions on Graphics* 26.3 (July 2007), p. 46. ISSN: 0730-0301, 1557-7368. DOI: [10.1145/1276377.1276435](https://doi.org/10.1145/1276377.1276435).
- [10] Rook Bridson and Matthias Müller-Fischer. *Fluid Simulation - SIGGRAPH 2007 Course Notes*. en. Aug. 2007. URL: <https://www.cs.ubc.ca/~rbridson/fluidsimulation>.
- [11] Mark Carlson, Peter J. Mucha, and Greg Turk. “Rigid fluid: animating the interplay between rigid bodies and fluid”. In: *ACM Trans. Graph.* 23.3 (Aug. 2004). Place: New York, NY, USA Publisher: Association for Computing Machinery, pp. 377–384. ISSN: 0730-0301. DOI: [10.1145/1015706.1015733](https://doi.org/10.1145/1015706.1015733).
- [12] Jumyung Chang et al. “Curl-Flow: Boundary-Respecting Pointwise Incompressible Velocity Interpolation for Grid-Based Fluids”. en. In: *ACM Transactions on Graphics* 41.6 (Dec. 2022), pp. 1–21. ISSN: 0730-0301, 1557-7368. DOI: [10.1145/3550454.3555498](https://doi.org/10.1145/3550454.3555498).
- [13] Alexandre J. Chorin and Jerrold E. Marsden. *A Mathematical Introduction to Fluid Mechanics*. en. 3rd ed. Springer Texts in Statistics. New York, NY, USA: Springer-Verlag, 1993.
- [14] Leonardo Dagum and Ramesh Menon. “OpenMP: an industry standard API for shared-memory programming”. In: *Computational Science & Engineering, IEEE* 5.1 (1998), pp. 46–55.
- [15] Todd F. Dupont and Yingjie Liu. “Back and forth error compensation and correction methods for removing errors induced by uneven gradients of the level set function”. en. In: *Journal of Computational Physics* 190.1 (Sept. 2003), pp. 311–324. ISSN: 00219991. DOI: [10.1016/S0021-9991\(03\)00276-6](https://doi.org/10.1016/S0021-9991(03)00276-6).
- [16] Sharif Elcott et al. “Stable, circulation-preserving, simplicial fluids”. In: *ACM Trans. Graph.* 26.1 (Jan. 2007). Place: New York, NY, USA Publisher: Association for Computing Machinery, 4-es. ISSN: 0730-0301. DOI: [10.1145/1189762.1189766](https://doi.org/10.1145/1189762.1189766).
- [17] Olivier Génevaux, Arash Habibi, and Jean-Michel Dischler. “Simulating Fluid-Solid Interaction”. en. In: *Proceedings of Graphics Interface 2003* Halifax (2003). Ed. by Torsten Möller and Colin Ware. Artwork Size: 8 pages, 8.29 MB ISBN: 9781568812076 Medium: application/pdf Publisher: Canadian Human-Computer Communications Society and A K Peters Ltd., 8 pages, 8.29 MB. ISSN: 0713-5424. DOI: [10.20380/GI2003.04](https://doi.org/10.20380/GI2003.04). URL: <http://graphicsinterface.org/proceedings/gi2003/gi2003-4/>.

- [18] Frederic Gibou et al. “A Second-Order-Accurate Symmetric Discretization of the Poisson Equation on Irregular Domains”. en. In: *Journal of Computational Physics* 176.1 (Feb. 2002), pp. 205–227. ISSN: 00219991. DOI: [10.1006/jcph.2001.6977](https://doi.org/10.1006/jcph.2001.6977).
- [19] Frédéric Gibou and Chohong Min. “Efficient symmetric positive definite second-order accurate monolithic solver for fluid/solid interactions”. en. In: *Journal of Computational Physics* 231.8 (Apr. 2012), pp. 3246–3263. ISSN: 00219991. DOI: [10.1016/j.jcp.2012.01.009](https://doi.org/10.1016/j.jcp.2012.01.009).
- [20] Roland Glowinski, Stanley J. Osher, and Wotao Yin, eds. *Splitting Methods in Communication, Imaging, Science, and Engineering*. en. Scientific Computation. Cham: Springer International Publishing, 2016. ISBN: 978-3-319-41587-1. DOI: [10.1007/978-3-319-41589-5](https://doi.org/10.1007/978-3-319-41589-5).
- [21] Eran Guendelman et al. “Coupling water and smoke to thin deformable and rigid shells”. en. In: (July 2005).
- [22] Gaël Guennebaud, Benoît Jacob, et al. *Eigen v3*. <http://eigen.tuxfamily.org>. 2010.
- [23] J.L. Guermond, P. Mineev, and Jie Shen. “An overview of projection methods for incompressible flows”. en. In: *Computer Methods in Applied Mechanics and Engineering* 195.44-47 (Sept. 2006). Publisher: Elsevier BV, pp. 6011–6045. ISSN: 0045-7825. DOI: [10.1016/j.cma.2005.10.010](https://doi.org/10.1016/j.cma.2005.10.010).
- [24] Francis H. Harlow and J. Eddie Welch. “Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface”. en. In: *The Physics of Fluids* 8.12 (Dec. 1965), pp. 2182–2189. ISSN: 0031-9171. DOI: [10.1063/1.1761178](https://doi.org/10.1063/1.1761178).
- [25] J. Jiang et al. “A Second-Order Explicit Pressure Projection Method for Eulerian Fluid Simulation”. en. In: *Computer Graphics Forum* 41.8 (2022), pp. 95–105. ISSN: 1467-8659. DOI: [10.1111/cgf.14627](https://doi.org/10.1111/cgf.14627).
- [26] Hans Johansen and Phillip Colella. “A Cartesian Grid Embedded Boundary Method for Poisson’s Equation on Irregular Domains”. In: *Journal of Computational Physics* 147.1 (Nov. 1998), pp. 60–85. ISSN: 0021-9991. DOI: [10.1006/jcph.1998.5965](https://doi.org/10.1006/jcph.1998.5965).
- [27] Bryan M. Klingner et al. “Fluid animation with dynamic meshes”. In: *ACM Trans. Graph.* 25.3 (July 2006). Place: New York, NY, USA Publisher: Association for Computing Machinery, pp. 820–825. ISSN: 0730-0301. DOI: [10.1145/1141911.1141961](https://doi.org/10.1145/1141911.1141961).
- [28] Chohong Min and Frédéric Gibou. “Geometric integration over irregular domains with application to level-set methods”. In: *Journal of Computational Physics* 226.2 (Oct. 2007), pp. 1432–1443. ISSN: 0021-9991. DOI: [10.1016/j.jcp.2007.05.032](https://doi.org/10.1016/j.jcp.2007.05.032).

- [29] Patrick Mullen et al. “Energy-preserving integrators for fluid animation”. en. In: *ACM Transactions on Graphics* 28.3 (July 2009), pp. 1–8. ISSN: 0730-0301, 1557-7368. DOI: [10.1145/1531326.1531344](https://doi.org/10.1145/1531326.1531344). URL: <https://www.cs.cmu.edu/~kmcrane/Projects/EnergyPreservingFluid/>.
- [30] Matthias Müller, David Charypar, and Markus Gross. “Particle-based fluid simulation for interactive applications”. In: *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. SCA ’03. event-place: San Diego, California. Goslar, DEU: Eurographics Association, 2003, pp. 154–159. ISBN: 1-58113-659-5.
- [31] Mohammad Sina Nabizadeh et al. “Covector fluids”. en. In: *ACM Transactions on Graphics* 41.4 (July 2022), pp. 1–16. ISSN: 0730-0301, 1557-7368. DOI: [10.1145/3528223.3530120](https://doi.org/10.1145/3528223.3530120).
- [32] Rahul Narain, Jonas Zehnder, and Bernhard Thomaszewski. “A Second-Order Advection-Reflection Solver”. en. In: *Proceedings of the ACM on Computer Graphics and Interactive Techniques* 2.2 (July 2019), pp. 1–14. ISSN: 2577-6193. DOI: [10.1145/3340257](https://doi.org/10.1145/3340257).
- [33] Yen Ting Ng, Chohong Min, and Frédéric Gibou. “An efficient fluid–solid coupling algorithm for single-phase flows”. en. In: *Journal of Computational Physics* 228.23 (Dec. 2009), pp. 8807–8829. ISSN: 00219991. DOI: [10.1016/j.jcp.2009.08.032](https://doi.org/10.1016/j.jcp.2009.08.032).
- [34] Michael B. Nielsen et al. “Physics-Based Combustion Simulation”. en. In: *ACM Transactions on Graphics* 41.5 (Oct. 2022), pp. 1–21. ISSN: 0730-0301, 1557-7368. DOI: [10.1145/3526213](https://doi.org/10.1145/3526213).
- [35] Michael B. Nielsen et al. “Physics-Based Combustion Simulation”. en. In: *ACM Transactions on Graphics* 41.5 (Oct. 2022), pp. 1–21. ISSN: 0730-0301, 1557-7368. DOI: [10.1145/3526213](https://doi.org/10.1145/3526213).
- [36] Charles S. Peskin. “The immersed boundary method”. en. In: *Acta Numerica* 11 (Jan. 2002), pp. 479–517. ISSN: 0962-4929, 1474-0508. DOI: [10.1017/S0962492902000077](https://doi.org/10.1017/S0962492902000077).
- [37] Patrick J. Roache. “Code Verification by the Method of Manufactured Solutions”. en. In: *Journal of Fluids Engineering* 124.1 (Mar. 2002), pp. 4–10. ISSN: 0098-2202, 1528-901X. DOI: [10.1115/1.1436090](https://doi.org/10.1115/1.1436090).
- [38] Avi Robinson-Mosher, R. Elliot English, and Ronald Fedkiw. “Accurate tangential velocities for solid fluid coupling”. en. In: *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. New Orleans Louisiana: ACM, Aug. 2009, pp. 227–236. ISBN: 978-1-60558-610-6. DOI: [10.1145/1599470.1599500](https://doi.org/10.1145/1599470.1599500).

- [39] Doug Roble, Nafees Bin Zafar, and Henrik Falt. “Cartesian grid fluid simulation with irregular boundary voxels”. en. In: *ACM SIGGRAPH 2005 Sketches on - SIGGRAPH '05*. Los Angeles, California: ACM Press, 2005, p. 138. DOI: [10.1145/1187112.1187279](https://doi.org/10.1145/1187112.1187279).
- [40] Patrick Schmidt et al. “TinyAD: Automatic Differentiation in Geometry Processing Made Simple”. In: *Computer Graphics Forum* 41.5 (2022).
- [41] Andrew Selle et al. “An Unconditionally Stable MacCormack Method”. en. In: *Journal of Scientific Computing* 35.2-3 (June 2008), pp. 350–371. ISSN: 0885-7474, 1573-7691. DOI: [10.1007/s10915-007-9166-4](https://doi.org/10.1007/s10915-007-9166-4).
- [42] Jos Stam. “Stable fluids”. en. In: *Proceedings of the 26th annual conference on Computer graphics and interactive techniques - SIGGRAPH '99* (1999), pp. 121–128. DOI: [10.1145/311535.311548](https://doi.org/10.1145/311535.311548).
- [43] Jos Stam and Eugene Fiume. “Depicting fire and other gaseous phenomena using diffusion processes”. en. In: *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques - SIGGRAPH '95*. Not Known: ACM Press, 1995, pp. 129–136. ISBN: 978-0-89791-701-8. DOI: [10.1145/218380.218430](https://doi.org/10.1145/218380.218430).
- [44] Nils Thuerey. “Physically based animation of free surface flows with the Lattice Boltzmann method”. PhD thesis. Erlangen: University of Erlangen–Nuremberg, 2007. URL: https://www10.cs.fau.de/publications/dissertations/Diss_2007-Thuerey.pdf.
- [45] Dongbin Xiu and George Em Karniadakis. “A Semi-Lagrangian High-Order Method for Navier–Stokes Equations”. In: *Journal of Computational Physics* 172.2 (Sept. 2001), pp. 658–684. ISSN: 0021-9991. DOI: [10.1006/jcph.2001.6847](https://doi.org/10.1006/jcph.2001.6847).
- [46] Jonas Zehnder, Rahul Narain, and Bernhard Thomaszewski. “An advection-reflection solver for detail-preserving fluid simulation”. en. In: *ACM Transactions on Graphics* 37.4 (Aug. 2018), pp. 1–8. ISSN: 0730-0301, 1557-7368. DOI: [10.1145/3197517.3201324](https://doi.org/10.1145/3197517.3201324).
- [47] Yongning Zhu and Rook Bridson. “Animating Sand as a Fluid”. en. In: (July 2005).

APPENDICES

Appendix A

2D analogues of 3D operators

This thesis is concerned with 2D and 3D simulations, and is mostly written to be agnostic of dimension. Some computations involve the cross product $\times$ and curl operator $\nabla \times$, which are well-defined in $\mathbb{R}^3$ (and $\mathbb{R}^7$) but not in $\mathbb{R}^2$. For convenience, we continue to use these notations and define the following interpretations of these operators on $\mathbb{R}^2$, which lead to analogous computations for our purposes:

- For a vector $\mathbf{v} = (a, b) \in \mathbb{R}^2$ and a scalar $\omega \in \mathbb{R}$, the 2D cross product $\mathbf{v} \times \omega$ is a 2D vector. It can be computed by embedding these quantities in $\mathbb{R}^3$, computing their 3D cross product, and projecting back to $\mathbb{R}^2$:

$$\begin{pmatrix} a \\ b \end{pmatrix} \times \omega \equiv \begin{pmatrix} a \\ b \\ 0 \end{pmatrix} \times \begin{pmatrix} 0 \\ 0 \\ \omega \end{pmatrix} = \begin{pmatrix} \omega b \\ -\omega a \\ 0 \end{pmatrix} \equiv \begin{pmatrix} \omega b \\ -\omega a \end{pmatrix}$$

This is equivalent to computing $\omega \mathbf{v}$ and rotating the result by 90° .

- For a scalar quantity $\phi \in \mathbb{R}$, the 2D curl $\nabla \times \phi$ is a 2D vector. It can be computed by embedding ϕ in the z -component in $\mathbb{R}^3$, taking its 3D curl, and projecting back to $\mathbb{R}^2$:

$$\nabla \times \phi \equiv \nabla \times \begin{pmatrix} 0 \\ 0 \\ \phi \end{pmatrix} = \begin{pmatrix} \frac{\partial \phi}{\partial y} \\ -\frac{\partial \phi}{\partial x} \\ 0 \end{pmatrix} \equiv \begin{pmatrix} \frac{\partial \phi}{\partial y} \\ -\frac{\partial \phi}{\partial x} \end{pmatrix}$$

- For a vector quantity $\mathbf{v} = (a, b)$, the 2D curl $\nabla \times \mathbf{v}$ is a scalar. It can be computed

by embedding $\mathbf{v}$ in $\mathbb{R}^3$, taking its 3D curl, and projecting the z -component:

$$\nabla \times \begin{pmatrix} a \\ b \end{pmatrix} \equiv \nabla \times \begin{pmatrix} a \\ b \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \frac{\partial b}{\partial x} - \frac{\partial a}{\partial y} \end{pmatrix} \equiv \frac{\partial b}{\partial x} - \frac{\partial a}{\partial y}$$

Appendix B

Supplementary video

The supplementary video contains animations of several of the examples discussed in chapter [6](#):

- Marker particle positions for the the dual vortices scene discussed in section [6.3.1](#),
- Marker particle positions for the circle-in-tube test case discussed in section [6.2.3](#),
- Marker particle positions and vorticity for the ellipse-in-box scene discussed in section [6.1.3](#),
- Marker particle positions and vorticity for the four-boxes scene used in section [6.4](#).