The Random Vortex Method for 2D Incompressible Flows

Candidate Number: 1064282

Part C Mathematics

University of Oxford

Word Count: 7430*

April 27, 2025

*This word count excludes the bibliography and the Code appendix (appendix B), as per departmental guidelines.

Contents

1 Introduction

The *Random Vortex Method* (RVM) is a mesh-free numerical method for modelling the incompressible fluid flow of the Navier-Stokes equations. It works by simulating the stochastic motion of *Random Vortices* which transport vorticity through the fluid. The vortices are convected by the fluid, and randomly diffused by viscosity.

The *Random Vortex Method* is domain specific. In this paper we present derivations in two domains; $\mathbb{R}^2$ and the half-plane $D = \{y < 0\}$. Then, we investigate the efficacy of the method through several novel numerical simulations. Finally, we explore the phenomena of boundary layer turbulence. We propose and demonstrate the effectiveness of a method for using the *Random Vortex Method* to study boundary layer phenomena.

The Random Vortex Method was first introduced by Chorin in the foundational paper [5], with further work in [6]. The convergence of the Random Vortex Method in two-dimension was studied first in [11], then improved by Long in [15]. Much of this paper follows from the work of Qian, Cherepanov et al in ([19], [4], [3]). For comprehensive reviews on the Random Vortex Method and turbulent flow respectively, see [16] and [17].

2 The Random Vortex Method on $\mathbb{R}^2$

In this chapter, the *Random Vortex Method* on $\mathbb{R}^2$ is introduced. First, it is demonstrated how the Navier-Stokes equations for the fluid velocity $\boldsymbol{u}(\boldsymbol{x}, t)$ can be converted into a problem for the vorticity $\omega(\boldsymbol{x}, t)$ - *The Vorticity Formulation.* Next, a solution to this problem is derived and linked to the random paths given by a stochastic differential equation. Finally, the complete *Random Vortex Method* is presented, which gives the fluid velocity $\boldsymbol{u}(\boldsymbol{x}, t)$ in terms of the behaviour of these random paths. For a review of the PDE methods used in this section, including Greeen's functions, see Evans [8].

2.1 Navier Stokes Equations

Consider the 2D incompressible Navier-Stokes equations for viscous fluids, with constant density ρ. Our aim is to find the fluid velocity $\boldsymbol{u}(x, y, t)$ which is a two-dimensional vector $\boldsymbol{u} = (u^1, u^2, 0)$ embedded in $\mathbb{R}^3$ for convenience. Often $\boldsymbol{u}(x, y, t)$ will be written as $\boldsymbol{u}(\boldsymbol{x}, t)$ where $\boldsymbol{x}$ denotes $(x, y) \in \mathbb{R}^2$. Here, ν is the kinematic viscosity defined as $\frac{\mu}{\rho}$, where μ is the dynamic viscosity. P is the fluid pressure, and $\boldsymbol{F}$ are external forces acting on the fluid.

Definition 1. ***The Navier-Stokes problem on*** $\mathbb{R}^2$. *The Navier-Stokes problem for the fluid velocity* $\boldsymbol{u}$ *on an unbounded domain* $\mathbb{R}^2$ *is as follows,*

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\frac{1}{\rho}\nabla P + \nu \nabla^2 \boldsymbol{u} + \boldsymbol{F}, \quad \boldsymbol{x} \in \mathbb{R}^2 \tag{1}$$

$$\nabla \cdot \boldsymbol{u} = 0, \quad \boldsymbol{x} \in \mathbb{R}^2 \tag{2}$$

We now introduce a key quantity, vorticity. Vorticity $\boldsymbol{\omega}$ is defined as the curl of the velocity field,

$$\boldsymbol{\omega} = \nabla \times \boldsymbol{u}. \tag{3}$$

For two-dimensional fluid flow, the only non-vanishing component of $\boldsymbol{\omega}$ will be the z-component, denoted by a lower-case ω so that,

$$\boldsymbol{\omega} = (0, 0, \omega) = (0, 0, \frac{\partial u^2}{\partial x} - \frac{\partial u^1}{\partial y}). \tag{4}$$

2.2 The Vorticity Formulation

Throughout this paper, we will make use of the *Vorticity Formulation* of the Navier-Stokes equations [14]. This formulation converts the problem of finding the velocity field $\boldsymbol{u}(\boldsymbol{x}, t)$, to one of finding the vorticity ω. As will be shown, the velocity field can then be recovered from the vorticity via convolution with the kernel $\boldsymbol{K}$. One immediate consequence of this formulation is that the fluid pressure P is eliminated.

2.2.1 Vorticity Equation

At the centre of the *Vorticity Formulation*, is the *Vorticity Equation* for ω. The *Vorticity Equation* is obtained by applying the curl operator to the Navier-Stokes momentum equation (eq. (1)). The calculation proceeds as follows.

Theorem 1. ***The Vorticity Equation.*** *The vorticity ω satisfies the following equation.*

$$\frac{\partial \omega}{\partial t} + (\boldsymbol{u} \cdot \nabla)\omega = \nu \nabla^2 \omega + G_{\boldsymbol{F}} \tag{5}$$

where $G_{\boldsymbol{F}}$ is such that

$$\nabla \times \boldsymbol{F} = (0, 0, G_{\boldsymbol{F}}) \tag{6}$$

Proof. Consider applying the curl operator to (eq. (1)), labelling the terms (A), (B), (C), (D), (E) from left to right.

$$\nabla \times \frac{\partial \boldsymbol{u}}{\partial t} + \nabla \times (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\frac{1}{\rho}\nabla \times (\nabla P) + \nu \nabla \times (\nabla^2 \boldsymbol{u}) + \nabla \times \boldsymbol{F} \tag{7}$$

(A) For the first term, $\nabla\times$ commutes with $\frac{\partial}{\partial t}$ so that,

$$\nabla \times \frac{\partial \boldsymbol{u}}{\partial t} = \frac{\partial \boldsymbol{\omega}}{\partial t}. \tag{8}$$

(B) For the second term, observe the following vector calculus identity,

$$(\boldsymbol{u} \cdot \nabla)\boldsymbol{u} \equiv \frac{1}{2}\nabla(|\boldsymbol{u}|^2) - \boldsymbol{u} \times (\nabla \times \boldsymbol{u}). \tag{9}$$

It follows that

$$\begin{aligned}
\nabla \times (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} &= -\nabla \times (\boldsymbol{u} \times (\nabla \times \boldsymbol{u})) \\
&= -\nabla \times (\boldsymbol{u} \times \boldsymbol{\omega}) \\
&= -\boldsymbol{u}(\nabla \cdot \boldsymbol{\omega}) + \boldsymbol{\omega}(\nabla \cdot \boldsymbol{u}) - (\boldsymbol{\omega} \cdot \nabla)\boldsymbol{u} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{\omega} \\
&= -\boldsymbol{u}(0) + \boldsymbol{\omega}(0) - (\boldsymbol{\omega} \cdot \nabla)\boldsymbol{u} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{\omega} \\
&= -(\boldsymbol{\omega} \cdot \nabla)\boldsymbol{u} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{\omega} \\
&= (\boldsymbol{u} \cdot \nabla)\boldsymbol{\omega}.
\end{aligned}$$

The first line uses the fact that the gradient of a curl vanishes. The second line uses that $\nabla \times (A \times B) = B \cdot \nabla A - (\nabla \cdot A)B + (\nabla \cdot B)A - A \cdot \nabla B$. The fourth line makes use of incompressibility $\nabla \cdot \boldsymbol{u} = 0$, and the vorticity being divergence-free as it is defined as a curl of the velocity field, so $\nabla \cdot \boldsymbol{\omega} = 0$. The simplification in the final line follows as $(\boldsymbol{\omega} \cdot \nabla)\boldsymbol{u} = \omega \frac{\partial \boldsymbol{u}}{\partial z} = 0$.

(C) The third term vanishes as the curl of a gradient is zero,

$$\nabla \times (\nabla P) = 0. \tag{10}$$

(D) For the fourth term, the vector Laplacian is equivalent to,

$$\nabla^2 \boldsymbol{u} \equiv \nabla(\nabla \cdot \boldsymbol{u}) - \nabla \times (\nabla \times \boldsymbol{u}). \tag{11}$$

Taking the curl of eq. (11), it follows that,

$$\begin{aligned} \nabla \times (\nabla^2 \boldsymbol{u}) &= \mathbf{0} - \nabla \times (\nabla \times (\nabla \times \boldsymbol{u})) \\ &= -\nabla \times (\nabla \times \boldsymbol{\omega}) \\ &= \nabla^2 \boldsymbol{\omega}. \end{aligned}$$

(E). For the fifth term, observe that as $\boldsymbol{F} \in \mathbb{R}^2$, $\nabla \times \boldsymbol{F}$ will only have a z-component, so can be written as in (eq. (6)).
Finally, taking the z-component of eq. (7) gives,

$$\frac{\partial \omega}{\partial t} + (\boldsymbol{u} \cdot \nabla)\omega = \nu \nabla^2 \omega + G_{\boldsymbol{F}}, \quad \boldsymbol{x} \in \mathbb{R}^2. \tag{12}$$

□

To find a solution to the *Vorticity Equation*, an initial vorticity distribution ω_0 must be specified. Note the absence of the fluid pressure P from this formulation of the Navier-Stokes equations and the use of the scalar quantity ω.

$$\omega(\boldsymbol{x}, 0) = \omega_0(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^2 \tag{13}$$

2.2.2 Recovering the fluid velocity

Observe that the *Vorticity Equation* depends on the fluid velocity $\boldsymbol{u}$ which is unknown. Thus to close this system, an expression for $\boldsymbol{u}$ in terms of the vorticity ω is required. Note that throughout the rest of section (section 2), $G_{\boldsymbol{F}}$ is taken to be zero. However, in section (section 3) this force term is considered.

Consider the following expression for $\nabla \times \boldsymbol{\omega}$.

$$\begin{aligned} -\nabla \times \boldsymbol{\omega} &= \nabla \times (0, 0, \frac{\partial u^2}{\partial x} - \frac{\partial u^1}{\partial y}) \\ &= -(-\frac{\partial^2 u^1}{\partial x^2}, -\frac{\partial^2 u^2}{\partial y^2}, 0) \\ &= \nabla^2 \boldsymbol{u} \end{aligned}$$

This is an elliptic equation in $\boldsymbol{u}$, which can be solved using the Green's function $G(\boldsymbol{x}, \boldsymbol{y}) := \frac{1}{2\pi} log(|\boldsymbol{x} - \boldsymbol{y}|)$ on $\mathbb{R}^2$, and further simplified using integration by parts.

$$\begin{aligned} \boldsymbol{u}(\boldsymbol{x}, t) &= \int_{\mathbb{R}^2} G(\boldsymbol{x} - \boldsymbol{\xi})(-\nabla \times \boldsymbol{\omega}(\boldsymbol{\xi}, t)) d\xi^1 d\xi^2 && \boldsymbol{\xi} = (\xi^1, \xi^2) \\ &= \int_{\mathbb{R}^2} G(\boldsymbol{x}, \boldsymbol{\xi})(-\frac{\partial \omega(\boldsymbol{\xi}, t)}{\partial y}, \frac{\partial \omega(\boldsymbol{\xi}, t)}{\partial x}, 0) d\xi^1 d\xi^2 \\ &= \int_{\mathbb{R}^2} \omega(\boldsymbol{\xi}, t)(\frac{\partial G(\boldsymbol{x}, \boldsymbol{\xi})}{\partial y}, -\frac{\partial G(\boldsymbol{x}, \boldsymbol{\xi})}{\partial x}, 0) d\xi^1 d\xi^2 && \text{(IBP)} \\ &= \int_{\mathbb{R}^2} \omega(\boldsymbol{\xi}, t) \boldsymbol{K}(\boldsymbol{x} - \boldsymbol{\xi}) d\xi^1 d\xi^2 && (\boldsymbol{K} \text{ defined below}) \end{aligned}$$

Hence the fluid velocity $\boldsymbol{u}$ can be recovered from the vorticity ω using the Biot-Savart kernel $\boldsymbol{K}$ for $\mathbb{R}^2$ as follows. Since the velocity field $\boldsymbol{u}$ can be expressed in terms of the vorticity ω, the *Vorticity Equation* (eq. (12)) is now a closed problem with the only unknown being ω.

$$\boldsymbol{u}(\boldsymbol{x},t) = \int_{\mathbb{R}^2} \omega(\boldsymbol{\xi},t)\boldsymbol{K}(\boldsymbol{x}-\boldsymbol{\xi})d\xi^1 d\xi^2, \quad \boldsymbol{x} \in \mathbb{R}^2 \tag{14}$$

$$\textit{where } \boldsymbol{K}(x,y) = \boldsymbol{K}(\boldsymbol{x}) := \frac{1}{2\pi|\boldsymbol{x}|^2}(-y,x), \quad \boldsymbol{x} \in \mathbb{R}^2. \tag{15}$$

2.3 The SDE connection

Although the *Vorticity Equation* is now a closed problem for ω, it is not immediately obvious how it can be solved. The key insight of the *Random Vortex Method* is to link the *Vorticity Formulation* to a random process $\boldsymbol{X}_t \in \mathbb{R}^2$ which solves a stochastic differential equation (SDE) [12]. Consider the following SDE for $\boldsymbol{X}_t$, where $\boldsymbol{B}_t$ is a standard Brownian Motion.

$$d\boldsymbol{X}_t = \boldsymbol{u}dt + \sqrt{2\nu}d\boldsymbol{B}_t \tag{16}$$

The processes $\boldsymbol{X}_t$ are hereby called *Random Vortices*. Observe from (eq. (16)), that the vortices are convected with the fluid and that viscosity induces random diffusion. The transition probability density $p(\boldsymbol{x},t;\boldsymbol{\xi},s)$ of $\boldsymbol{X}_t$ is defined so that,

$$\mathbb{P}[\boldsymbol{X}_t \in A | \boldsymbol{X}_s = \boldsymbol{\xi}] = \int_A p(\boldsymbol{x},t;\boldsymbol{\xi},s)dx^1dx^2 \qquad \forall A \subseteq \mathbb{R}^2, \quad s < t \tag{17}$$

This transition probability p is also the fundamental solution/ Green's function for the operator $\partial t + (\boldsymbol{u}\cdot\nabla) - \nu\nabla^2$ on $\mathbb{R}^2$. It follows that we can write down the solution of ω in terms of the transition probability density p,

$$\omega(\boldsymbol{x},t) = \int_{\mathbb{R}^2} p(\boldsymbol{x},t;\boldsymbol{\xi},0)\omega_0(\boldsymbol{\xi})dx^1dx^2. \tag{18}$$

This solution can then be used to recover the fluid velocity. In the following, $\boldsymbol{X}_t^{\boldsymbol{\eta}}$ denotes a weak solution to the SDE (eq. (16)) with initial condition $\boldsymbol{X}_0 = \boldsymbol{\eta}$. Also, the following definition will be used.

$$\mathbb{E}\left[f(\boldsymbol{X}_t^{\boldsymbol{\eta}})\right] = \int_{\mathbb{R}^2} f(\boldsymbol{\xi})p(\boldsymbol{\xi},t;\boldsymbol{\eta},0)d\xi^1 d\xi^2$$

The proof proceeds as follows.

$$\begin{aligned}
\boldsymbol{u}(\boldsymbol{x},t) &= \int_{\mathbb{R}^2}\left\{\int_{\mathbb{R}^2} p(\boldsymbol{\xi},t;\boldsymbol{\eta},0)\omega_0(\boldsymbol{\eta})d\eta^1 d\eta^2\right\} \boldsymbol{K}(\boldsymbol{x}-\boldsymbol{\xi})dx^1dx^2 \\
&= \int_{\mathbb{R}^2}\left\{\int_{\mathbb{R}^2} p(\boldsymbol{\xi},t;\boldsymbol{\eta},0)\boldsymbol{K}(\boldsymbol{x}-\boldsymbol{\xi})dx^1dx^2\right\} \omega_0(\boldsymbol{\eta})d\eta^1 d\eta^2 \qquad \text{(Fubinis)} \\
&= \int_{\mathbb{R}^2} \mathbb{E}[\boldsymbol{K}(\boldsymbol{x}-\boldsymbol{X}_t^{\boldsymbol{\eta}})]\ \omega_0(\boldsymbol{\eta})d\eta^1 d\eta^2
\end{aligned}$$

2.4 Formulation of the Random Vortex Method

We now have all the pieces needed to state the Random Vortex Method.

Theorem 2. ***The Random Vortex Method on*** $\mathbb{R}^2$ *Given initial vorticity* ω_0*, the Navier-Stokes problem, (Dfn 1) on* $\mathbb{R}^2$ *is solved by,*

$$\boldsymbol{u}(\boldsymbol{x},t) = \int_{\mathbb{R}^2} \mathbb{E}[\boldsymbol{K}(\boldsymbol{x} - \boldsymbol{X}_t^{\boldsymbol{\xi}})]\ \omega_0(\boldsymbol{\xi}) d\xi^1 d\xi^2, \quad \boldsymbol{x} \in \mathbb{R}^2. \tag{19}$$

Where $\boldsymbol{X}^{\boldsymbol{\eta}}{}_t$ *is a weak solution to the SDE,*

$$\begin{aligned} d\boldsymbol{X}^{\boldsymbol{\eta}}{}_t = &\left[\int_{\mathbb{R}^2} \mathbb{E}[\boldsymbol{K}(\boldsymbol{X}^{\boldsymbol{\eta}}{}_t - \boldsymbol{X}_t^{\boldsymbol{\xi}})]\ \omega_0(\boldsymbol{\xi}) d\xi^1 d\xi^2\right] dt \\ &+ \sqrt{2\nu} d\boldsymbol{B}_t \qquad \textit{where} \qquad \boldsymbol{X}_0^{\boldsymbol{\eta}} = \boldsymbol{\eta},\ \forall \boldsymbol{\eta} \in \mathbb{R}^2. \end{aligned} \tag{20}$$

In the following chapter, the discretisation and numerical simulation of the *Random Vortex Method* on $\mathbb{R}^2$ is described. The key idea is to simulate the motion of several random vortices $\boldsymbol{X}_t$ to approximate (eq. (19)).

2.5 Discretisation

We now discuss how the *Random Vortex Method* on $\mathbb{R}^2$, as stated in (section 2.4), can be discretised. The challenge of discretisation lies in the approximation of the integral (eq. (19)). Here we present a possible discretisation, similar to in [19]. Firstly, the integral is approximated by a sum of expectations over M random paths $\{\boldsymbol{X}_t^{\boldsymbol{\eta}_m}\}_{m=1}^M$ which follow the SDE (eq. (20)), where $\boldsymbol{X}_0^{\boldsymbol{\eta}_m} = \boldsymbol{\eta}_m$. For simplicity, our points $\boldsymbol{\eta}_m \in \mathbb{R}^2$ are from a uniform mesh of width $h > 0$ in $\mathbb{R}^2$. While there are different ways of choosing the initial positions $\boldsymbol{\eta}_m$, one must ensure that the $d\xi^1 d\xi^2$ term from (eq. (19)) is correctly discretised. In the case of a uniform grid, the discretisation is simply $d\xi^1 d\xi^2 \mapsto h^2$. The region spanned by the initial positions $\boldsymbol{\eta}_m$ roughly determines the domain in which the approximation produced by this discretisation is sensible. Here, $\tilde{\boldsymbol{u}}$ denotes the approximate velocity field.

$$\tilde{\boldsymbol{u}}(\boldsymbol{X},t) \approx \sum_{m=1}^{M} \mathbb{E}[\boldsymbol{K}(\boldsymbol{X} - \boldsymbol{X}_t^{\boldsymbol{\eta}_m})]\ \omega_0(\boldsymbol{\eta}_m) h^2 \tag{21}$$

The next step is to approximate the expectation $\mathbb{E}[\boldsymbol{K}(\boldsymbol{X} - \boldsymbol{X}_t^{\boldsymbol{\eta}_m})]$. One method of approximation is to carry out a Monte-Carlo estimate by simulating multiple sets of random paths $\{\{\boldsymbol{X}_t^{\boldsymbol{\eta}_m,(n)}\}_{m=1}^M\}_{n=1}^N$ where each set is driven by an independent Brownian Motion $\boldsymbol{B}_t^{(n)}$. Another method of approximating is to instead use a single set of random paths $\{\boldsymbol{X}^{\boldsymbol{\eta}_m}\}_{m=1}^M$ where each path is driven by an independent Brownian motion $\boldsymbol{B}_t^m$. We call this method the 'Independent Random Vortex Method'. In this case, convergence was proved by Long in [15] to be of order $M^{-1/2} ln(M)$.

2.5.1 Monte-Carlo discretisation

We now describe a discretisation using a Monte-Carlo estimate of the expectation. Consider simulating N sets of random paths $\{\{\boldsymbol{X}_t^{\boldsymbol{\eta}_m,(n)}\}_{m=1}^M\}_{n=1}^N$ where each set has an independent Brownian Motion $\boldsymbol{B}_t^{(n)}$. By averaging over these sets, the expectation can be approximated by,

$$\tilde{\boldsymbol{u}}(\boldsymbol{x},t) \approx \sum_{m=1}^{M} \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{K}(\boldsymbol{x} - \boldsymbol{X}_t^{\boldsymbol{\eta}_m,(n)})\ \omega_0(\boldsymbol{\eta}_m) h^2. \tag{22}$$

Where the paths follow the SDE,

$$d\boldsymbol{X}_t^{\boldsymbol{\eta}_j,(k)} = \left[\sum_{m=1}^{M}\frac{1}{N}\sum_{n=1}^{N}\boldsymbol{K}(\boldsymbol{X}_t^{\boldsymbol{\eta}_j,(k)} - \boldsymbol{X}_t^{\boldsymbol{\eta}_m,(n)})\ \omega_0(\boldsymbol{\eta}_m)h^2\right]dt + \sqrt{2\nu}d\boldsymbol{B}_t^{(k)}, \qquad \boldsymbol{X}_0^{\boldsymbol{\eta}_j,(k)} = \boldsymbol{\eta}_j. \tag{23}$$

It is then trivial to simulate this system using, for example, Euler's method on (eq. (23)) with some timestep Δt. See algorithm (algorithm 1). Note that $\boldsymbol{B}_{\Delta t+t}^{(k)} - \boldsymbol{B}_t^{(k)}$ is calculated by sampling a normal distribution $\mathcal{N}(\boldsymbol{0}, \mathcal{I}\Delta t)$, where $\mathcal{I}$ is the identity matrix.

Algorithm 1 Euler's method for (eq. (22)) and (eq. (23)).

INPUT: Mesh width h, number of paths M, number of sets of paths N, timestep Δt, terminating time $T = K \times \Delta t$, for some $K \in \mathbb{N}$.

OUTPUT: The approximate velocity field $\tilde{\boldsymbol{u}}(\boldsymbol{x}, t)$ at $t = T$.

Step 1: Set $\boldsymbol{X}_0^{\boldsymbol{\eta}_j,(k)} = \boldsymbol{\eta}_j$ and $t = 0$.

Step 2: For each $k = 1, .., N$, $j = 1, .., M$, update the paths by,

$$\boldsymbol{X}_{\Delta t+t}^{\boldsymbol{\eta}_j,(k)} = \boldsymbol{X}_t^{\boldsymbol{\eta}_j,(k)} + \left[\sum_{m=1}^{M}\frac{1}{N}\sum_{n=1}^{N}\boldsymbol{K}(\boldsymbol{X}_t^{\boldsymbol{\eta}_j,(k)} - \boldsymbol{X}_t^{\boldsymbol{\eta}_m,(n)})\ \omega_0(\boldsymbol{\eta}_m)h^2\right]\Delta t + \sqrt{2\nu}(\boldsymbol{B}_{\Delta t+t}^{(k)} - \boldsymbol{B}_t^{(k)}) \tag{24}$$

Step 3: Set $t := t + \Delta t$.

Step 4: Repeat **Step 2-3**, K times, then recover $\tilde{\boldsymbol{u}}(\boldsymbol{x}, T)$ using (eq. (22)).

2.5.2 Independent discretisation

A brief description is provided of the *independent discretisation* method of the Random Vortex Method. Simply take algorithm (algorithm 1) with $N = 1$, except now the random vortices are each driven by an independent Brownian Motion, so satisfy the following SDE. Note, we drop the (n) superscript in the following and denote $\boldsymbol{X}_t^{\boldsymbol{\eta}_m,(1)} = \boldsymbol{X}_t^{\boldsymbol{\eta}_m}$

$$d\boldsymbol{X}_t^{\boldsymbol{\eta}_m} = \boldsymbol{u}dt + \sqrt{2\nu}d\boldsymbol{B}_t^m$$

In practice, increasing the number of sets of paths N in the Monte-Carlo simulation approach has little effect on the simulation. However, increasing the number of paths M can significantly improve accuracy. Hence the benefit of the independent discretisation approach is that $N = 1$ allows a larger M which produces more robust simulations without increasing computational complexity.

2.5.3 Smoothing the Kernel

Recall the definition of the Biot-Savarat kernel $\boldsymbol{K}$ (eq. (15)) on $\mathbb{R}^2$ has a singularity at the origin, where $\boldsymbol{K}(\boldsymbol{x}) \to \boldsymbol{\infty}$ as $\boldsymbol{x} \to \boldsymbol{0}$.

$$\boldsymbol{K}(x, y) = \boldsymbol{K}(\boldsymbol{x}) := \frac{1}{2\pi|\boldsymbol{x}|^2}(-y, x), \quad \boldsymbol{x} \in \mathbb{R}^2. \tag{25}$$

This singularity arises because the velocity field around a point vortex is singular. This feature of the kernel is greatly undesirable as it will produce an approximate velocity field $\tilde{\boldsymbol{u}}$ with singularities at each $\boldsymbol{X}^{\boldsymbol{\eta}_j,(k)}$. To remove this singularity, a smoothed kernel $\boldsymbol{K}_\delta$ is used. First, cutoff functions must be defined.

Definition 2. ***Cutoff functions***. *$f(r)$ is a cutoff function if it satisfies the following conditions.*

1. *$f(r) = \mathcal{O}(r)$ for $r \ll 1$*
2. *$f(r) \uparrow 1$ as $r \uparrow \infty$*

Below are several examples of cutoff functions used in existing work, as collated in [2]. Observe how each satisfies the conditions of definition (Dfn 2).

Example 1. *Chorin [5]* $f(r) = \begin{cases} 1 & \text{when } r > 1 \\ r & \text{when } r \leq 1 \end{cases}$

Example 2. *Kuhawara & Takami [13]* $f(r) = 1 - e^{-r^2}$,

Definition 3. ***Regularised kernel***. *The regularised kernel $\boldsymbol{K}_\delta$ is written in terms of a cutoff function $f(r)$, and cutoff radius δ as follows.*

$$\boldsymbol{K}_\delta(\boldsymbol{x}) = \boldsymbol{K}(\boldsymbol{x}) f\left(\frac{r}{\delta}\right) \tag{26}$$

$$\text{where} \quad r = \|\boldsymbol{x}\|_2$$

In all numerical simulations throughout the remainder of this paper, a regularised kernel $\boldsymbol{K}_\delta$ will always be used instead of $\boldsymbol{K}$. Specifically, a Chorin-cutoff (Ex 1) where $\delta = 0.1$ is used.

2.6 Simulation results

To close out this section, some novel simulation results from our implementation of the Random Vortex Method are presented. This simulation is written in Python [20], and is entirely our own. These simulations are all designed to illustrate key features of the Random Vortex Method.

Example 3. ***Two vortices (inviscid)***. *First, consider the inviscid case ($\nu = 0$), with the simple initial vorticity distribution of two point-vortices with opposite strengths placed at $(1, 0)$ and $(-1, 0)$. i.e.*

$$\omega_0(\boldsymbol{x}) = \delta(\boldsymbol{x} - (-1, 0)) - \delta(\boldsymbol{x} - (1, 0)). \tag{27}$$

This problem is simulated with two random vortices $\boldsymbol{X}_t^1, \boldsymbol{X}_t^2$ which are initialised at $\boldsymbol{X}_0^1 = (1, 0)$ and $\boldsymbol{X}_t^2 = (-1, 0)$. Due to the lack of viscosity, SDE eq. (20) now becomes deterministic.

$$d\boldsymbol{X}_t^1 = \boldsymbol{K}(\boldsymbol{X}_t^1 - \boldsymbol{X}_t^2)\omega_0(\boldsymbol{X}_0^2)dt$$
$$d\boldsymbol{X}_t^2 = \boldsymbol{K}(\boldsymbol{X}_t^2 - \boldsymbol{X}_t^1)\omega_0(\boldsymbol{X}_0^1)dt$$

It is a classical result [1] that the vortices will move side-by-side in a straight line with velocity $\frac{\Gamma}{2\pi d}$ where Γ is the vortex strength, and d is their separation. In this case, $\Gamma = 1$, $d = 2$ so we expect the vortices to move with speed $\frac{1}{4\pi} \approx 0.079$. Indeed, figure (fig. 1) confirms that the vortices behave as expected in this case. It is encouraging that the simulation can replicate this well known result.

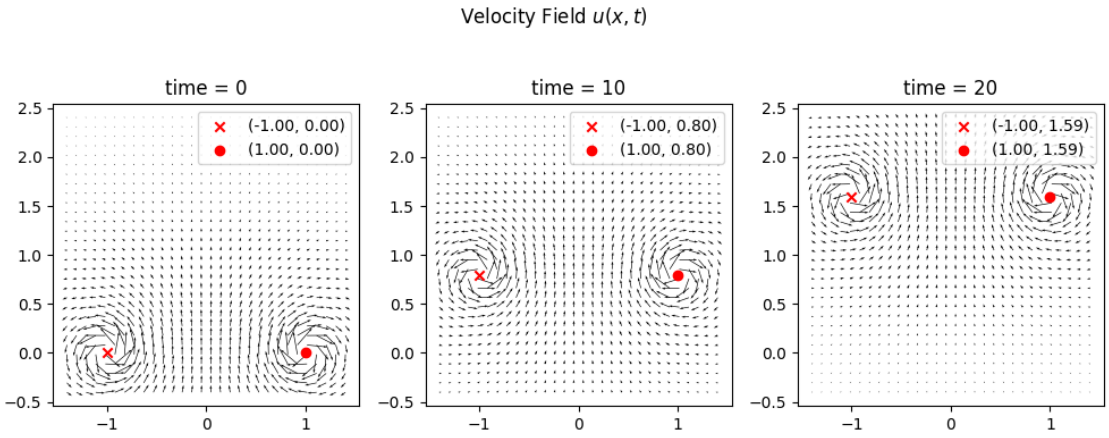

Figure 1: The velocity fields for two point-vortices with equal and opposite strength, and zero viscosity, as described in example (Ex 3). Here, a Chorin cutoff function with $\delta = 0.2$ is used. Observe the speed of the vortices is $0.08 \approx \frac{1}{4\pi}$, as predicted.

Example 4. ***Vorticity drives randomness.*** *Consider a random vortex moving in a static uniform velocity field* $\boldsymbol{u}(\boldsymbol{x}, t) = (1, 0)$ *with viscosity* ν*. In this example, it assumed that the vortex does not affect the velocity field* $\boldsymbol{u}$*. The motion will be driven by SDE (eq.* (28)*). This example illustrates how viscosity drives the randomness of the simulation. This is shown in figure (fig. 2). The vortex with no viscosity follows a straight deterministic path, whereas the vortex with viscosity follows a stochastic path.*

$$d\boldsymbol{X}_t = (1, 0)dt + \sqrt{2\nu}d\boldsymbol{B}_t \tag{28}$$

Example 5. ***Viscous fluid flow.*** *Consider the following simulation set-up for algorithm (algorithm 1); domain* $= [-\pi, \pi] \times [-\pi, \pi]$*,* $h = 2\pi/15$*,* $\nu = 0.15$*,* $\Delta_t = 0.002$*,* $T = 0.2$*. Take the following initial vorticity and velocity fields.*

$$\begin{aligned} \boldsymbol{u}(x, y, 0) &= (0, 40\sin(x)), \\ \omega_0(x, y, 0) &= 40\cos(x) \end{aligned}$$

We simulate this system with the independent discretisation approach (2.5.2). The results are shown in figure (fig. 3) and discussed below.

Vortex Density. *For the Random Vortex Method to produce a good approximation of the velocity field* $\boldsymbol{u}$ *on a domain, there must be sufficient density of random vortices present in the domain. The simulation is initialised with a grid of vortices on the domain* $[-\pi, \pi] \times [-\pi, \pi]$*, so it should be accurate in this region for small times* t*. As the simulation progresses however it is possible for the random vortices to leave this domain. As the density of random vortices within the domain decreases, so does the accuracy of the simulation.*

Finite Domain. *An inherent limitation of the Random Vortex Method is that it must be executed on a finite domain, in this case,* $[-\pi, \pi] \times [-\pi, \pi]$*. The simulation only considers the evolution of vorticity which originates inside this domain. So the effects of fluid vorticity which originates outside of this domain is not taken into account, which introduces error.*

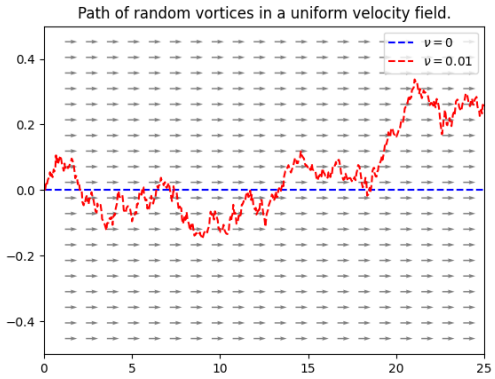

Figure 2: Paths followed by a random vortex moving in a uniform velocity field with different viscosities ν, as described in example (Ex 4). Observe how introducing viscosity drives the randomness of the motion

3 The Random Vortex Method on a half-plane

In the following section we consider how the *Random Vortex Method* can be adapted to solve the Navier-Stokes equations under a force $\boldsymbol{F}$ for fluid flow on the half-plane $D := \{(x, y) \in \mathbb{R}^2 \mid y < 0\}$. The key challenge to this section is how to incorporate the *no-slip boundary condition*. The derivation presented in this section is adapted from [19].

3.1 Derivation of the Random Vortex Method on a half-plane

In this section, the derivation of (section 2) is modified to work on the half-plane D and to include a force term $\boldsymbol{F}$. The Navier-Stokes equations now take the following form.

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\frac{1}{\rho}\nabla P + \nu \nabla^2 \boldsymbol{u} + \boldsymbol{F}, \quad \boldsymbol{x} \in D \tag{29}$$

$$\nabla \cdot \boldsymbol{u} = 0, \quad \boldsymbol{x} \in D \tag{30}$$

Definition 4. ***No-slip condition***. *The no-slip condition says fluid flow restricted to a domain D must have zero velocity on the boundary ∂D. i.e,*

$$\boldsymbol{u}(\boldsymbol{x}, t) = \boldsymbol{0}, \qquad \forall \boldsymbol{x} \in \partial D \tag{31}$$

Definition 5. ***The Navier-Stokes problem on*** D. *The Navier-Stokes problem for the fluid velocity $\boldsymbol{u}(\boldsymbol{x}, t)$ on the bounded domain $\boldsymbol{x} \in D$ is given by the equations (eq. (29)), (eq. (30)) and the no-slip boundary condition (Dfn 4).*

The first step in formulating the RVM is finding a closed-form expression for the vorticity ω. While the Vorticity Equation takes an identical form to (*eq.* (12)), ω must now satisfy a boundary condition due to the no-slip condition on $\boldsymbol{u}$. This boundary condition means that ω cannot immediately be recovered from the initial vorticity ω_0 as in (*eq.* (18)). For the $\mathbb{R}^2$ solution for ω (*eq.* (18)) to be valid, the vorticity ω must vanish on the boundary. As $\frac{\partial \boldsymbol{u}}{\partial x}(\boldsymbol{x}, 0) = 0$ on the boundary, ω must satisfy the following boundary condition.

$$\omega(x, 0, t) = -\frac{\partial u^1}{\partial y} \tag{32}$$

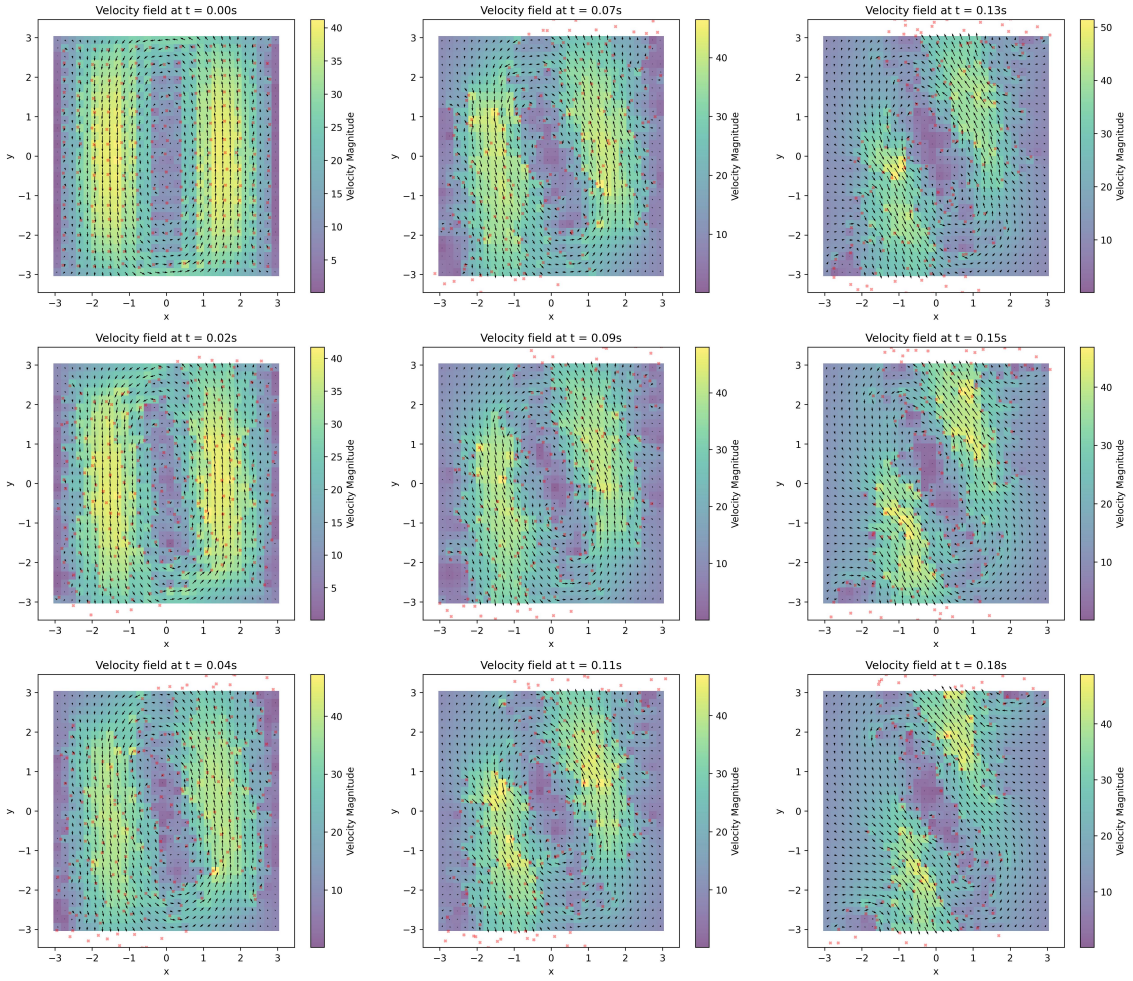

Figure 3: A simulation of the random vortex system described in example (Ex 5). Arrows show the direction of the velocity field, the shading shows the velocity magnitude $\|\boldsymbol{u}\|_2$, and the crosses show the location of the random vortices. This simulation took 12 seconds to run.

To remove this boundary condition we introduce a perturbation to the vorticity, w_ϵ which modifies it in a small layer near the boundary so that w_ϵ vanishes on the boundary. The Vorticity Equation (eq. (12)) can now be restated and solved in w_ϵ.

Definition 6. ***Perturbed Vorticity*** *Suppose that $\phi : \mathbb{R} \to [0,1]$ is a cutoff function (Dfn 2), and $\epsilon > 0$ is a small constant. Denote the boundary vorticity by θ so that $\theta(x,t) := \omega(x,0,t)$. Then w_ϵ is defined as follows.*

$$w_\epsilon(x,y,t) := \omega(x,y,t) - \theta(x,t)\phi(-y/\epsilon) \qquad \forall (x,y) \in D \tag{33}$$

Observe that $w_\epsilon(x,0,t) = \omega(x,0,t) + \frac{\partial u^1}{\partial y}(x,0,t) = 0$ by definition of ω. Hence w_ϵ vanished on the boundary ∂D.

Proposition 1. ***Perturbed Vorticity Equation*** *The perturbed vorticity w_ϵ satisfies the following equation.*

$$\begin{aligned} (\frac{\partial}{\partial t} + \boldsymbol{u}\cdot\nabla - \nu\nabla^2) w_\epsilon(\boldsymbol{x},t) &= g_\epsilon(\boldsymbol{x},t) \qquad \forall \boldsymbol{x} \in D \\ \text{where } w_\epsilon(\boldsymbol{x},t) &= 0 \qquad \forall \boldsymbol{x} \in \partial D \end{aligned} \tag{34}$$

where

$$\begin{aligned} g_\epsilon(x,y,t) := G_{\boldsymbol{F}} &+ \frac{\nu}{\epsilon^2}\phi''(-y/\epsilon)\theta(x,t) + \frac{1}{\epsilon}\phi'(-y/\epsilon)u^2(x,y,t)\theta(x,t) + \\ &\phi(-y/\epsilon)\left(\nu\frac{\partial^2\theta}{\partial x^2}(x,t) - \frac{\partial\theta}{\partial t}(x,t)\right) - \phi(-y/\epsilon)u^1(x,y,t)\frac{\partial\theta}{\partial x}(x,t) \end{aligned} \tag{35}$$

Proof. It was shown in definition (Dfn 6) that $w_\epsilon = 0$ on D. Start with the Vorticity Equation (*eq.* (12)) and substitute $\omega(x,y,t) = w_\epsilon(x,y,t) + \phi(-y/\epsilon)\theta(x,t)$.

$$\begin{aligned} \frac{\partial(w_\epsilon + \phi\theta)}{\partial t} + (\boldsymbol{u}\cdot\nabla)(w_\epsilon + \phi\theta) &= \nu\nabla^2(w_\epsilon + \phi\theta) + G_{\boldsymbol{F}} \\ \frac{\partial w_\epsilon}{\partial t} + \phi\frac{\partial\theta}{\partial t} + (\boldsymbol{u}\cdot\nabla)w_\epsilon + u^1\frac{\partial\theta}{\partial x}\phi - \frac{1}{\epsilon}u^2\frac{\partial\phi}{\partial y}\theta &= \nu\nabla^2 w_\epsilon + \nu\nabla^2(\phi\theta) + G_{\boldsymbol{F}} \end{aligned}$$

Collect the w_ϵ terms on the left. Here, ϕ' denotes $\frac{\partial\phi}{\partial y}$. The following product rule for the Laplacian is used, $\nabla^2(fg) = g\nabla^2 f + f\nabla^2 g + 2(\nabla f)\cdot(\nabla g)$.

$$\begin{aligned} (\frac{\partial}{\partial t} + \boldsymbol{u}\cdot\nabla - \nu\nabla^2) w_\epsilon &= -\phi\frac{\partial\theta}{\partial t} - u^1\frac{\partial\theta}{\partial x}\phi + \frac{1}{\epsilon}u^2\phi'\theta + \frac{\nu}{\epsilon^2}\phi''\theta - \frac{2\nu}{\epsilon}(0,\phi')\cdot(\frac{\partial\theta}{\partial x},0) + \nu\phi\frac{\partial^2\theta}{\partial x} + G_{\boldsymbol{F}} \\ &= -\phi\frac{\partial\theta}{\partial t} - u^1\frac{\partial\theta}{\partial x}\phi + \frac{1}{\epsilon}u^2\phi'\theta + \frac{\nu}{\epsilon^2}\phi''\theta + \nu\phi\frac{\partial^2\theta}{\partial x} + G_{\boldsymbol{F}} \\ &=: g_\epsilon \end{aligned}$$

□

Equation (eq. (34)) with boundary condition (eq. (32)) can now be solved using the fundamental solution on the half-plane D for the operator $\partial t + (\boldsymbol{u}\cdot\nabla) - \nu\nabla^2$. Recall that the fundamental solution on $\mathbb{R}^2$ is denoted by $p(\boldsymbol{x},t;\boldsymbol{\xi},s)$. By the reflection principle, the fundamental solution on the half-plane D is given by $p_D(\boldsymbol{x},t;\boldsymbol{\xi},s) := p(\boldsymbol{x},t;\boldsymbol{\xi},s) - p(\boldsymbol{x},t;\overline{\boldsymbol{\xi}},s)$. Note that here $\overline{\boldsymbol{x}}$ denotes reflection in the boundary ∂D, so that $\overline{(x,y)} = (x,-y)$. This fundamental solution is now used to state a solution for w_ϵ which is rearranged for ω.

Proposition 2. ***Solving for vorticity****. Equations (eq.* (34)*) with boundary condition (eq.* (32)*) have the following solution.*

$$w_\epsilon(\boldsymbol{x},t) = \int_D \left[p(\boldsymbol{x},t;\boldsymbol{\xi},0) - p(\boldsymbol{x},t;\overline{\boldsymbol{\xi}},0)w_\epsilon(\xi,0)\right] d\xi^1 d\xi^2 + \int_0^t \int_D \left[p(\boldsymbol{x},t;\boldsymbol{\xi},s) - p(\boldsymbol{x},t;\overline{\boldsymbol{\xi}},s)\right] g_\epsilon(\boldsymbol{\xi},s) d\xi^1 d\xi^2 ds \tag{36}$$

Further, this implies the following equation for ω. Note that from this point forward, $\sigma_\epsilon(x,y,t) := \theta(x,t)\phi(-y/\epsilon)$.

$$\omega(\boldsymbol{x},t) = \int_D \left[p(\boldsymbol{x},t;\boldsymbol{\xi},0) - p(\boldsymbol{x},t;\overline{\boldsymbol{\xi}},0)w_\epsilon(\xi,0)\right] d\xi^1 d\xi^2 + \int_0^t \int_D \left[p(\boldsymbol{x},t;\boldsymbol{\xi},s) - p(\boldsymbol{x},t;\overline{\boldsymbol{\xi}},s)\right] g_\epsilon(\boldsymbol{\xi},s) d\xi^1 d\xi^2 ds + \sigma_\epsilon(\boldsymbol{x},t) \tag{37}$$

The next step is to recover the fluid velocity $\boldsymbol{u}$ from the vorticity ω (eq. (37)). The following analysis for the half-plane differs only slightly from that of $\mathbb{R}^2$. Consider the following problem for $\boldsymbol{u}$.

$$\begin{aligned} \nabla^2 \boldsymbol{u}(\boldsymbol{x},t) &= -\nabla \times \boldsymbol{\omega}(\boldsymbol{x},t) && \forall \boldsymbol{x} \in D \\ \boldsymbol{u}(\boldsymbol{x},t) &= \boldsymbol{0} && \forall \boldsymbol{x} \in \partial D \end{aligned}$$

This problem is once again solved by a Green's function for the half-plane D, denoted G_D.

$$\boldsymbol{u}(\boldsymbol{x},t) = \int_D G_D(\boldsymbol{x},\boldsymbol{\xi})(-\nabla \times \boldsymbol{\omega}(\boldsymbol{\xi},t)) d\xi^1 d\xi^2 \tag{38}$$

$$\text{where } G_D(\boldsymbol{x},\boldsymbol{y}) := \frac{1}{4\pi} \log(1 - \frac{4x_2 y_2}{\|\boldsymbol{y} - \bar{\boldsymbol{x}}\|_2^2}), \qquad (\boldsymbol{x},\boldsymbol{y}) = ((x^1,x^2),(y^1,y^2)). \tag{39}$$

As before, using integration by parts, this formula can be rewritten as a Biot-Savart law with a kernel $\boldsymbol{K}_D$ as follows.

$$\boldsymbol{u}(\boldsymbol{x},t) = \int_D \omega(\boldsymbol{\xi},t) \boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{\xi}) d\xi^1 d\xi^2 \tag{40}$$

Where the kernel $\boldsymbol{K}_D$ can be defined in terms of the $\mathbb{R}^2$ kernel $\boldsymbol{K}$ as follows.

$$\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{y}) := \boldsymbol{K}(\boldsymbol{x},\boldsymbol{y}) - \boldsymbol{K}(\overline{\boldsymbol{x}},\boldsymbol{y}) \tag{41}$$

An explicit formula for $\boldsymbol{K}_D$ is stated below. Here, $\boldsymbol{K}_D = (K_D^1, K_D^2)$ and $\boldsymbol{x} = (x_1,x_2), \boldsymbol{y} = (y_1,y_2)$.

$$K_D^1(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{2\pi}\left(\frac{y_2 - x_2}{\|\boldsymbol{y} - \boldsymbol{x}\|_2^2} - \frac{y_2 + x_2}{\|\boldsymbol{y} - \overline{\boldsymbol{x}}\|_2^2}\right) \tag{42}$$

$$K_D^2(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{2\pi}\left(\frac{y_2 - x_1}{\|\boldsymbol{y} - \overline{\boldsymbol{x}}\|_2^2} - \frac{y_1 - x_1}{\|\boldsymbol{y} - \boldsymbol{x}\|_2^2}\right) \tag{43}$$

To close out this section, formula (*eq.* (37)) can be substituted into (eq. (40)) to state *The Random Vortex Method on the half-plane D.*

Theorem 3. ***The Random Vortex Method for the half-plane****. Given initial vorticity ω_0, the Navier-Stokes problem on the half-plane D (Dfn 5) is solved by the following.*

$$\boldsymbol{u}(\boldsymbol{x},t) = \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\boldsymbol{\xi},0})\mathbb{1}_D(\boldsymbol{X}_t^{\boldsymbol{\xi},0}) - \boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\overline{\boldsymbol{\xi}},0})\mathbb{1}_D(\boldsymbol{X}_t^{\overline{\boldsymbol{\xi}},0})\right] w_\epsilon(\boldsymbol{\xi},0) d\xi^1 d\xi^2 +$$

$$\int_0^t \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{X}_t^{\boldsymbol{\xi},s})\mathbb{1}_D(\boldsymbol{X}_t^{\boldsymbol{\xi},s}) - \boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},s})\mathbb{1}_D(\boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},s})\right] g_\epsilon(\boldsymbol{\xi}, s) d\xi^1 d\xi^2 ds + \int_D \sigma_\epsilon(\boldsymbol{\zeta}, t)\boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{\zeta}) d\zeta^1 d\zeta^2 \tag{44}$$

Where $\boldsymbol{X}_t^{\boldsymbol{\xi},s}$ *evolves according to the following SDE.*

$$\begin{aligned} d\boldsymbol{X}_t^{\boldsymbol{\eta},s} = &\left\{ \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{X}_t^{\boldsymbol{\eta},s}, \boldsymbol{X}_t^{\boldsymbol{\xi},0})\mathbb{1}_D(\boldsymbol{X}_t^{\boldsymbol{\xi},0}) - \boldsymbol{K}_D(\boldsymbol{X}_t^{\boldsymbol{\eta},s}, \boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},0})\mathbb{1}_D(\boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},0})\right] w_\epsilon(\boldsymbol{\xi}, 0) d\xi^1 d\xi^2 + \right. \\ &\int_0^t \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{X}_t^{\boldsymbol{\eta},s}, \boldsymbol{X}_t^{\boldsymbol{\xi},s})\mathbb{1}_D(\boldsymbol{X}_t^{\boldsymbol{\xi},s}) - \boldsymbol{K}_D(\boldsymbol{X}_t^{\boldsymbol{\eta},s}, \boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},s})\mathbb{1}_D(\boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},s})\right] g_\epsilon(\boldsymbol{\xi}, s) d\xi^1 d\xi^2 ds + \\ &\left. \int_D \sigma_\epsilon(\boldsymbol{\zeta}, t)\boldsymbol{K}_D(\boldsymbol{X}_t^{\boldsymbol{\eta},s}, \boldsymbol{\zeta}) d\zeta^1 d\zeta^2 \right\} dt + \sqrt{2\nu} d\boldsymbol{B}_t, \quad \forall t \geq s \\ &\textit{where} \quad \boldsymbol{X}_s^{\boldsymbol{\eta},s} = \boldsymbol{\eta} \end{aligned} \tag{45}$$

Proof. Recall the interpretation of the fundamental solution p as the translation probability for the stochastic process *eq.* (16). This fact is used in the following proof to convert integrals to expectations over the random process $\boldsymbol{X}_t^{\boldsymbol{\xi},s}$. To begin, substitute (eq. (37)) into (eq. (40)) to find the fluid velocity $\boldsymbol{u}$ in terms of $\boldsymbol{X}_s^{\boldsymbol{\xi},s}$. Note the following result.

$$\mathbb{E}\left[f(\boldsymbol{X}_t^{\boldsymbol{\xi},s})\mathbb{1}_D(\boldsymbol{X}_t^{\boldsymbol{\xi},s})\right] = \int_D f(\boldsymbol{\eta}) p(\boldsymbol{\eta}, t; \boldsymbol{\xi}, s) d\eta^1 d\eta^2$$

$$\begin{aligned} \boldsymbol{u}(\boldsymbol{x}, t) &= \int_D \omega(\boldsymbol{\eta}, t)\boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{\eta}) d\eta^1 d\eta^2 && \text{by } (\textit{eq.}\ (40)) \\ &= \int_D \left\{ \int_D \left[p(\boldsymbol{\eta}, t; \boldsymbol{\xi}, 0) - p(\boldsymbol{\eta}, t; \bar{\boldsymbol{\xi}}, 0) w_\epsilon(\boldsymbol{\xi}, 0)\right] d\xi^1 d\xi^2 + \right. \\ &\left. \int_0^t \int_D \left[p(\boldsymbol{\eta}, t; \boldsymbol{\xi}, s) - p(\boldsymbol{\eta}, t; \bar{\boldsymbol{\xi}}, s)\right] g_\epsilon(\boldsymbol{\xi}, s) d\xi^1 d\xi^2 ds + \sigma_\epsilon(\boldsymbol{\eta}, t) \right\} \boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{\eta}) d\eta^1 d\eta^2 && \text{by } (\textit{eq.}\ (37)) \\ &= \int_D \left\{ \int_D \left[(p(\boldsymbol{\eta}, t; \boldsymbol{\xi}, 0) - p(\boldsymbol{\eta}, t; \bar{\boldsymbol{\xi}}, 0))\boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{\eta})\right] d\eta^1 d\eta^2 \right\} w_\epsilon(\boldsymbol{\xi}, 0) d\xi^1 d\xi^2 + \\ &\int_0^t \int_D \left\{ \int_D \left[(p(\boldsymbol{\eta}, t; \boldsymbol{\xi}, s) - p(\boldsymbol{\eta}, t; \bar{\boldsymbol{\xi}}, s))\boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{\eta})\right] d\eta^1 d\eta^2 \right\} g_\epsilon(\boldsymbol{\xi}, s) d\xi^1 d\xi^2 ds + \\ &\int_D \sigma_\epsilon(\boldsymbol{\eta}, t)\boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{\eta}) d\eta^1 d\eta^2 && \text{(Fubinis)} \\ &= \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{X}_t^{\boldsymbol{\xi},0})\mathbb{1}_D(\boldsymbol{X}_t^{\boldsymbol{\xi},0}) - \boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},0})\mathbb{1}_D(\boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},0})\right] w_\epsilon(\boldsymbol{\xi}, 0) d\xi^1 d\xi^2 + \\ &\int_0^t \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{X}_t^{\boldsymbol{\xi},s})\mathbb{1}_D(\boldsymbol{X}_t^{\boldsymbol{\xi},s}) - \boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},s})\mathbb{1}_D(\boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},s})\right] g_\epsilon(\boldsymbol{\xi}, s) d\xi^1 d\xi^2 ds + \\ &\int_D \sigma_\epsilon(\boldsymbol{\zeta}, t)\boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{\zeta}) d\zeta^1 d\zeta^2 && (\boldsymbol{\eta} \to \boldsymbol{\zeta}) \end{aligned}$$

$$\text{Where } \mathbb{1}_D(\boldsymbol{x}) = \begin{cases} 1 & \text{when } \boldsymbol{x} \in D \\ 0 & \text{when } \boldsymbol{x} \notin D \end{cases}$$

Finally, substitute this expression for $\boldsymbol{u}$ into SDE (eq. (45)). □

3.2 The ϵ_∞ and ϵ_0 formulations

In section (section 3.1), the Random Vortex Method for the half-plane D (Thm 3) was derived. This statement of the Random Vortex Method depended on a parameter ϵ. As demonstrated in [19], by considering the limits of $\epsilon \to 0$ and $\epsilon \to \infty$, two alternative statements of the Random Vortex Method can be derived. We label these two statements as the ϵ_0 and ϵ_∞ formulations. In our simulations, we consider just the ϵ_0 statement, which takes the limit $\epsilon \to 0$.

A crucial benefit of the ϵ_0 statement over the ϵ_∞ statement or the original statement (Thm 3) is that all terms containing a derivative of the boundary vorticity $\theta(x,t) = \omega(x,0,t)$ vanish. This is beneficial as while the RVM can readily approximate the fluid velocity $\boldsymbol{u}$, it is more difficult to compute the vorticity $\omega(x,y,t)$.

Theorem 4. ***The ϵ_∞ formulation of the Random Vortex Method for the half-plane*** *Given initial vorticity ω_0, the Navier-Stokes problem on the half-plane D (Dfn 5) is solved by the following.*

$$\boldsymbol{u}(\boldsymbol{x},t) = \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\boldsymbol{\xi},0})\mathbb{1}_D(\boldsymbol{X}_t^{\boldsymbol{\xi},0}) - \boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},0})\mathbb{1}_D(\boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},0})\right] W(\boldsymbol{\xi},0)d\xi^1 d\xi^2 + \tag{46}$$

$$\int_0^t \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\boldsymbol{\xi},s})\mathbb{1}_D(\boldsymbol{X}_t^{\boldsymbol{\xi},s}) - \boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},s})\mathbb{1}_D(\boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},s})\right] g(\boldsymbol{\xi},s)d\xi^1 d\xi^2 ds + \tag{47}$$

$$\int_D \theta(\zeta^1,t)\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{\zeta})d\zeta^1 d\zeta^2 \tag{48}$$

where $W(x,y,t) = \omega(x,y,t) - \omega(x,0,t)$

and $g(\boldsymbol{\xi},t) = G_{\boldsymbol{F}} + \left(\nu\dfrac{\partial^2\theta}{\partial x^2}(x,t) - \dfrac{\partial\theta}{\partial t}(x,t)\right) - \phi(-y/\epsilon)u^1(x,y,t)\dfrac{\partial\theta}{\partial x}(x,t)$

Proof. Let $\epsilon \to \infty$. See [19] for details. □

Theorem 5. ***The ϵ_0 formulation of the Random Vortex Method for the half-plane*** *Given initial vorticity ω_0, the Navier-Stokes problem on the half-plane D (Dfn 5) is solved by the following.*

$$\boldsymbol{u}(\boldsymbol{x},t) = \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\boldsymbol{\xi},0})\mathbb{1}_D(\boldsymbol{X}_t^{\boldsymbol{\xi},0}) - \boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},0})\mathbb{1}_D(\boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},0})\right] \omega_0(\boldsymbol{\xi})d\xi^1 d\xi^2 \tag{49}$$

$$\int_0^t \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\boldsymbol{\xi},s})\mathbb{1}_D(\boldsymbol{X}_t^{\boldsymbol{\xi},s}) - \boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},s})\mathbb{1}_D(\boldsymbol{X}_t^{\bar{\boldsymbol{\xi}},s})\right] G(\boldsymbol{\xi},s)d\xi^1 d\xi^2 ds \tag{50}$$

$$-2\nu\int_0^t \left.\frac{\partial}{\partial\xi^2}\right|_{\xi^2=0} \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\boldsymbol{\xi},s})\mathbb{1}_D(\boldsymbol{X}_t^{\boldsymbol{\xi},s})\right] \theta(\xi^1,s)d\xi^1 d\xi^2 ds + \tag{51}$$

Proof. Let $\epsilon \to 0$. See [19] for details. □

The theorems (Thm 5) (Thm 4) for a half-plane are harder to discretise than theorem (section 2) on $\mathbb{R}^2$. This is due to the time integral terms which involve vortices initialised at some time $s > 0$, $\boldsymbol{X}_t^{\boldsymbol{\xi},s}$. Specifically, the terms (eq. (47)), (eq. (51)) and (eq. (50)). This contrasts to the $\mathbb{R}^2$ case where all vortices are initialised at $s = 0$.

3.2.1 Discretising the ϵ_0 formulation

In this section, a discretisation of the ϵ_0 formulation (Thm 5) is motivated and presented. The discretisation of the integral (eq. (49)) will not be covered as it proceeds in an identical fashion to the Random Vortex Method on $\mathbb{R}^2$ with the addition of indicator functions $\mathbb{1}$ and the half-plane kernel $\boldsymbol{K}_D$. The key challenge to address is the calculation of the time integrals. A discretisation of the force term (eq. (50)) is not presented here due to its computational complexity. Hence $G_{\boldsymbol{F}}$ is set to 0. Thus we now examine the discretisation of the boundary term (eq. (51)), which is labelled as follows.

$$\boldsymbol{I}(\boldsymbol{x},t) := \int_0^t \left.\frac{\partial}{\partial \xi^2}\right|_{\xi^2=0} \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{X}_t^{\boldsymbol{\xi},s}) \mathbb{1}_D(\boldsymbol{X}_t^{\boldsymbol{\xi},s})\right] \theta(\xi^1, s) d\xi^1 d\xi^2 ds \tag{52}$$

The key observation for discretising $\boldsymbol{I}(\boldsymbol{x},t)$, as in [19] is that unlike with the integral (eq. (47)) we only need to consider random vortices emitted close to the boundary ∂D, rather than throughout the whole domain D. This is due to the integrand being a derivative evaluated at $\xi^2 = 0$. This is achieved in practice by introducing a new set of random vortices. These vortices will follow the canonical SDE $d\boldsymbol{X}_t = \boldsymbol{u}dt + \sqrt{2\nu} d\boldsymbol{B}_t$, but will only be used in the calculation of $\boldsymbol{I}$, not in (*eq.* (49)). Discretisations tend to partition a section of the boundary $\{(x,y) \mid y = 0,\ a \le x \le b\}$ using a 1-dimensional mesh of width h. Define $\boldsymbol{\eta}_m = (a + m(b-a)h, 0)$ and $\boldsymbol{\eta}_{m\epsilon} = (a + m(b-a)h, -\epsilon)$ for some small $\epsilon > 0$, $1 \le m \le N_b$ and $hN_b \approx 1$. [19] used the following discretisation.

$$\tilde{\boldsymbol{I}}(\boldsymbol{x}, (n+1)\Delta t) = h * \Delta t \sum_{i=1}^{n} \sum_{m=1}^{N_b} \frac{\boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{X}_{n\Delta t}^{\boldsymbol{\eta}_m}) \mathbb{1}_D(\boldsymbol{X}_{n\Delta t}^{\boldsymbol{\eta}_m}) - \boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{X}_{n\Delta t}^{\boldsymbol{\eta}_{m\epsilon}}) \mathbb{1}_D(\boldsymbol{X}_{n\Delta t}^{\boldsymbol{\eta}_{m\epsilon}})}{\epsilon} \theta(\boldsymbol{\eta}_m, i\Delta t) \tag{53}$$

Remark 1. ***Initial position perturbations***. *Here, we present a novel step to the discretisation which provides a more robust approximation to* $\tilde{\boldsymbol{I}}$. *A problem with the discretisation (eq.* (53)*) is that all the random vortices are emitted from the same set of lattice points* $\boldsymbol{\eta}_m$. *Due to the no-slip condition, these vortices tend not to move very far from* $\boldsymbol{\eta}_m$. *This causes 'clumps' of these vortices to build up which produce a non-smooth velocity field. Our novel way to solve this problem is to perturb the starting positions* $\boldsymbol{\eta}_m$ *at every timestep* $i\Delta t$. *Specifically, introduce* $\tilde{\boldsymbol{\eta}}_{m,i\Delta t}$ *as follows.*

$$\tilde{\boldsymbol{\eta}}_{m,i\Delta t} = \boldsymbol{\eta}_m + U_{i\Delta t}, \quad \textit{where} \quad U_{i\Delta t} \overset{iid}{\sim} \textit{Uniform}[-h/2, h/2] \tag{54}$$

It is important that all vortices are perturbed by the same quantity at each timestep so that they still form a mesh of width h. *The resulting discretisation is identical to (eq.* (53)*) except* $\boldsymbol{\eta}_m \to \tilde{\boldsymbol{\eta}}_{m,i\Delta t}$. *The effect of this perturbation is demonstrated in figure (fig.* 4*). Our novel method, as shown on the left, produces a smoother velocity field and avoids the 'clumping' of random vortices around the arbitrarily chosen initial positions. Removing this 'clumping' behaviour increases the smoothness and physicality of the resulting velocity field.*

3.2.2 Simulation results

We implemented the ϵ_0 formulation (*Thm* 5) in a novel, vectorised, Python simulation.

Example 6. ***Sinusoidal initial velocity***. *Consider the following novel simulation set-up for solving the Navier-Stokes equations on the domain* $D = \{y < 0\}$. $\nu = 0.15$, $Re = 25000$, $L = 2\pi$, $U_0 = \nu Re/L \approx 600$, $N_b = 10$ *with the following initial data.*

$$\boldsymbol{u}(x,y,0) = U_0 \sin(y)\,(1, \sin(x))\,, \quad \omega_0(x,y) = U_0\,[\cos(x)\sin(y) - cos(y)]$$

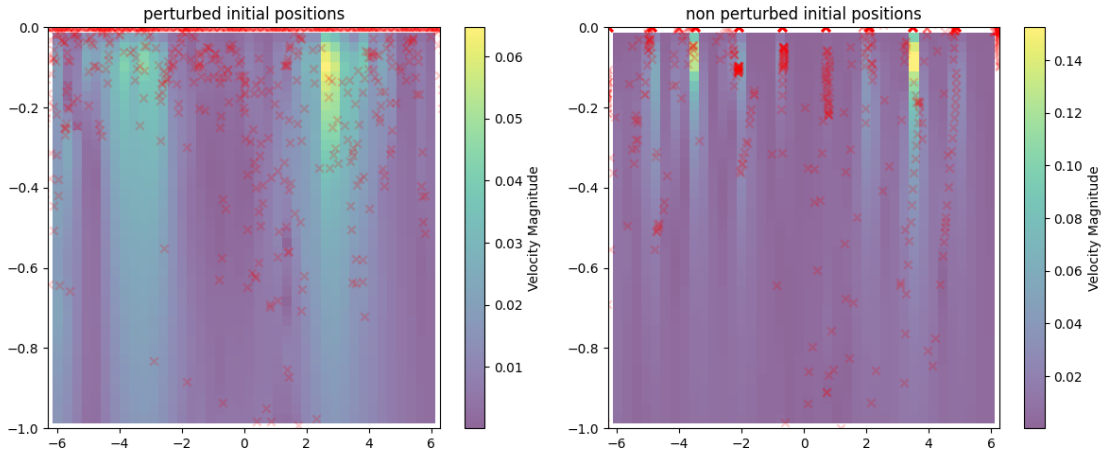

Figure 4: The boundary velocity $\boldsymbol{I}$ for example (Ex 6) at $t = 0.02$ under the two discretisation methods. In this plot, the red 'x's are the positions of the random vortices, and the shading gives the magnitude of the velocity field. The boundary $y = 0$ is located along the top of the plot.

This system is simulated on the domain $[-L, L] \times [-L, 0]$, with $\Delta t = 0.0005$, a 30×30 grid of random vortices, a Chorin-cutoff with $\delta = 0.1$, and an independent discretisation approach as in (section 2.5.2). See figure (fig. 5). This simulation behaves sensibly and demonstrates key features of the dynamics of fluid in a half-plane such as the viscosity inducing chaotic and random of the fluid. Observe also that as desired, the no-slip condition is satisfied at $y = 0$.

3.3 The t_0 formulation

A key breakthrough in the study of the Random Vortex Method was made in [4]. The author realised that the RVM could be formulated using only vortices emitted from $s = 0$, $\boldsymbol{X}_t^{\boldsymbol{\eta},0}$ without the need for vortices emitted at a later time $s > 0$, $\boldsymbol{X}_t^{\boldsymbol{\eta},s}$. This discovery significantly reduces the computational complexity of the Random Vortex Method by reducing its effective dimension by one. Now, random vortices need only to be generated across the spatial domain D rather than the joint spatial-temporal domain $D \times [0, t)$. This formulation, here termed the t_0 formulation, is once again considered in the case of flow in the half-plane $D = \{y < 0\}$. See [4] for the derivations of these formulations.

Theorem 6. *t_0 **formulation** Recall the definitions of w_ϵ, σ_ϵ, and g_ϵ from (Prop 1). Define $\gamma(\boldsymbol{X}, T)$ to be the reverse leaving time of the process $\boldsymbol{X}_t$ from the domain D. Specifically, $\gamma(\boldsymbol{X}, T) := \inf\{0 < s < T \mid \boldsymbol{X}_{T-s} \notin D\}$, with the convention that $\inf\{\} = \infty$. Then the Navier-Stokes equations for the domain D (Dfn 5), with initial data ω_0 are solved by the following.*

$$\begin{aligned}\boldsymbol{u}(\boldsymbol{x}, t) = &\int_D \boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{\eta}) \sigma_\epsilon(\boldsymbol{\eta}, t) d\eta^1 d\eta^2 \\ &+ \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{X}_t^{\boldsymbol{\eta}}) \mathbb{1}_{\{t < \gamma(\boldsymbol{X}^{\boldsymbol{\eta}}, t)\}}\right] w_\epsilon(\boldsymbol{\eta}, 0) d\eta^1 d\eta^2 \\ &+ \int_0^t \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x}, \boldsymbol{X}_t^{\boldsymbol{\eta}}) g_\epsilon(\boldsymbol{X}_s^{\boldsymbol{\eta}}, s) \mathbb{1}_{\{t-s < \gamma(\boldsymbol{X}_t^{\boldsymbol{\eta}}, t)\}}\right] d\eta^1 d\eta^2 ds \end{aligned} \tag{55}$$

It is not necessarily obvious how to interpret the indicator terms. First consider $\mathbb{1}_{\{t < \gamma(\boldsymbol{X}^{\boldsymbol{\eta}}, t)\}}$. This term vanishes and stays zero as soon as the random vortex $\boldsymbol{X}^{\boldsymbol{\eta}}$ leaves the domain D. On the other

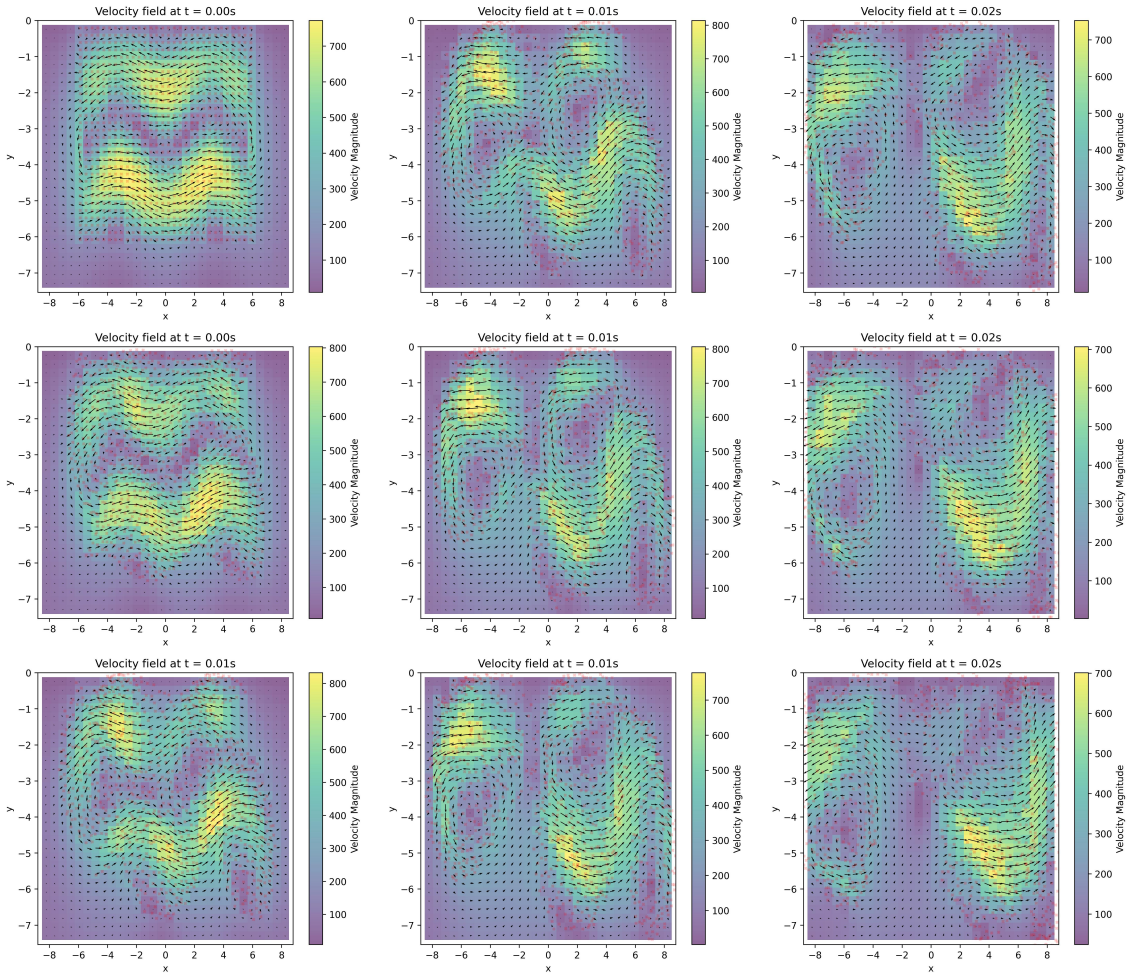

Figure 5: Simulation results for example (Ex 6). The vector field is the approximate velocity field $\tilde{\boldsymbol{u}}$, the red crosses are the random vortices, and the shading is for the velocity magnitude. The boundary $y = 0$ is located along the top of the plot. The domain of these plots is slightly larger than that of the simulation. This simulation took 20 seconds to run.

hand, as s varies, the indicator $\mathbb{1}_{\{t-s<\gamma(\boldsymbol{X}_t^{\boldsymbol{\eta}},t)\}}$ is zero up until s reaches the last leaving time of the process $\boldsymbol{X}^{\boldsymbol{\eta}}$, $s = t - \gamma(\boldsymbol{X}_t^{\boldsymbol{\eta}}, t)$. These indicator functions have the effect that once a random vortex leaves the domain D it will stop acting on other random vortices. However random vortices which are still in the domain will continue to act on it. So vortices outside the domain D continue to move.
This formulation mirrors that of theorem (Thm 3) with the key difference being that all diffusions start at $s = 0$. If one were to implement the t_0 as presented above, a significant challenge would be to calculate the derivatives of the boundary vorticity θ present in g_ϵ. See the discussion in section (section 3.2) for more detail. This difficulty can be avoided, as in the ϵ_0 formulation (Thm 5), by considering the limit as ϵ gets very small. This limit is trivial to take with the first two terms of (eq. (55)). Observe that no derivatives of the boundary vorticity θ are present.

$$\lim_{\epsilon\to 0}\sigma_\epsilon = 0 \quad \text{and} \quad \lim_{\epsilon\to 0} w_\epsilon(\boldsymbol{x},0) = \omega(\boldsymbol{x},0)$$

More care must be taken with the third term. Recall the definition of g_ϵ (eq. (35)). All terms vanish under the limit except for $\frac{\nu}{\epsilon^2}\phi''(-y/\epsilon)\theta(x,t)$. This leads to the following approximation.

Proposition 3. ***The*** ϵt_0 ***formulation.*** *The following expression approximates (eq.* (55)*) for small* ϵ*.*

$$\boldsymbol{u}(\boldsymbol{x},t) = \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\boldsymbol{\eta}})\mathbb{1}_{\{t<\gamma(\boldsymbol{X}^{\boldsymbol{\eta}},t)\}}\right]\omega(\boldsymbol{\eta},0)d\eta^1 d\eta^2 \tag{56}$$

$$+\int_0^t\int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\boldsymbol{\eta}})G_{\boldsymbol{F}}(\boldsymbol{X}_s^{\boldsymbol{\eta}},s)\mathbb{1}_{\{t-s<\gamma(\boldsymbol{X}_t^{\boldsymbol{\eta}},t)\}}\right]d\eta^1 d\eta^2 ds \tag{57}$$

$$+\frac{\nu}{\epsilon^2}\int_0^t\int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\boldsymbol{\eta}})\phi''(-\boldsymbol{X}_s^{\boldsymbol{\eta}}/\epsilon)\theta(\boldsymbol{X}_s^{\boldsymbol{\eta}},s)\mathbb{1}_{\{t-s<\gamma(\boldsymbol{X}_t^{\boldsymbol{\eta}},t)\}}\right]d\eta^1 d\eta^2 ds \tag{58}$$

Note here that $\phi''(\boldsymbol{x}), \theta(\boldsymbol{x},s)$*, are abuses of notation, as* ϕ, θ *are scalar functions with scalar inputs. Define* $\phi''(\boldsymbol{x}) = \phi''(y)$ *and* $\theta(\boldsymbol{x},s) = \theta(x)$*, where* $\boldsymbol{x} = (x,y)$*.*

3.3.1 Discretising the ϵt_0 formulation

In this section, our discretisation of formulation (*Prop* 3) is presented. The independent discretisation method (*section* 2.5.2) is used. Consider the third term of (Prop 3), specifically (51). Suppose $t = (n+1)\Delta t$. Suppose also that the last leaving time operator γ is redefined to return only discrete times $i\Delta t$ as follows $\gamma(\boldsymbol{X}^{\boldsymbol{\eta}_m}, n\Delta t) = \min\{i\Delta t \mid 0 < i < n,\ \boldsymbol{X}_{(n-i)\Delta t} \notin D\}$. Then a possible discretisation is as follows.

$$\frac{h^2\Delta t\nu}{\epsilon^2}\sum_{i=1}^{n}\sum_{j=1}^{m}\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_{n\Delta t}^{\boldsymbol{\eta}_j})\phi''(-\boldsymbol{X}_{i\Delta t}^{\boldsymbol{\eta}_j}/\epsilon)\theta(\boldsymbol{X}_{i\Delta t}^{\boldsymbol{\eta}_j}, i\Delta t)\mathbb{1}_{\{(n-i)\Delta t<\gamma(\boldsymbol{X}^{\boldsymbol{\eta}_j},n\Delta t)\}}$$

$$= \frac{h^2\Delta t\nu}{\epsilon^2}\sum_{j=1}^{m}\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_{n\Delta t}^{\boldsymbol{\eta}_j})\sum_{i=1}^{n}\left[\phi''(-\boldsymbol{X}_{i\Delta t}^{\boldsymbol{\eta}_j}/\epsilon)\theta(\boldsymbol{X}_{i\Delta t}^{\boldsymbol{\eta}_j}, i\Delta t)\mathbb{1}_{\{(n-i)\Delta t<\gamma(\boldsymbol{X}^{\boldsymbol{\eta}_j},n\Delta t)\}}\right] \tag{59}$$

Observe that the term (eq. (59)) can be treated as a running sum and updated incrementally. Define $\mathcal{J}_n = \{i \le n \mid (n-i)\Delta t < \gamma(\boldsymbol{X}^{\boldsymbol{\eta}_j}, n\Delta t)\}$. Defining (eq. (59)) as $I_{n\Delta t}^j$, it can be updated as follows.

$$\text{define} \quad I_{n\Delta t}^j := \sum_{i\in\mathcal{J}_n}\left[\phi''(-\boldsymbol{X}_{i\Delta t}^{\boldsymbol{\eta}_j}/\epsilon)\theta(\boldsymbol{X}_{i\Delta t}^{\boldsymbol{\eta}_j}, i\Delta t)\right]$$

Proposition 4. ***Incremental Sum Formula***. *$I^j_{n\Delta t}$ can be calculated using the following recurrance relation.*

$$I^j_0 = 0$$
$$I^j_{n\Delta t} = \mathbb{1}_{\{\boldsymbol{X}^{\boldsymbol{\eta}_m}_{n\Delta t} \in D\}}[I^j_{(n-1)\Delta t} + \phi''(-\boldsymbol{X}^{\boldsymbol{\eta}_j}_{n\Delta t}/\epsilon)\theta(\boldsymbol{X}^{\boldsymbol{\eta}_j}_{n\Delta t})]$$

Proof. Proceed with proof by induction. The base case is trivial.
Suppose first that $\boldsymbol{X}^{\boldsymbol{\eta}_m}_{n\Delta t} \notin D$. In this case, $0 = \gamma(\boldsymbol{X}^{\boldsymbol{\eta}_j}_{n\Delta t})$. Hence $\forall i \leq n,\ \mathbb{1}_{\{(n-i)\Delta t < \gamma(\boldsymbol{X}^{\boldsymbol{\eta}_j}, n\Delta t)\}} = 0$ and so $I^j_{n\Delta t} = 0$.
Now suppose instead that $\boldsymbol{X}^{\boldsymbol{\eta}_m}_{n\Delta t} \in D$. Then $n \in \mathcal{J}_n$. It hence follows that $I^j_{n\Delta t} - I^j_{(n-1)\Delta t} = \phi''(-\boldsymbol{X}^{\boldsymbol{\eta}_j}_{n\Delta t}/\epsilon)\theta(\boldsymbol{X}^{\boldsymbol{\eta}_j}_{n\Delta t})$. □

Remark 2. ***Boundary Vortices***. *Depending on the choice of cutoff function ϕ, the formula (Prop 4) can be greatly simplified. Suppose, as in [4], a cutoff function of the form (eq.* (60)*) is used. Then the second derivative ϕ'' vanishes except for $y \in (1/3, 2/3]$. So to obtain a robust estimate of the integral (eq.* (58)*), there must be a suitable density of random vortices in this range. One way to achieve this is to introduce a set of 'boundary vortices' on a much smaller mesh on some boundary layer $y \in [-\delta, 0]$ for some small $\delta > 0$. These boundary vortices can be combined with a standard set of random vortices on a larger mesh. These boundary vortices also have the benefit of increasing the resolution of the simulation around the boundary where the flow is expected to be most complex.*

$$\phi(y) = \begin{cases} 1 & 0 \leq y \leq 1/3 \\ 1/2 + 54(y-1/2)^3 - 9/2(y-1/2) & 1/3 < y \leq 2/3 \\ 0 & 2/3 < y \end{cases} \tag{60}$$

Now we are in a position to state a fully-vectorised algorithm for calculating the approximate fluid velocity $\tilde{\boldsymbol{u}}(\boldsymbol{x}, t)$. See algorithm (algorithm 2).

Remark 3. ***Calculating*** θ. *It remains to discretise the calculation of the boundary vorticity θ. Recall that $\theta(x,t) = (\frac{\partial u^2}{\partial x} - \frac{\partial u^1}{\partial y})|_{(x,0,t)}$, where $\boldsymbol{u} = (u^1, u^2)$. This expression can be simply discretised using a finite-difference-derivative as follows.*

$$\tilde{\theta}(\boldsymbol{x}, t) = \frac{u^2(x+\epsilon, 0, t) - u^2(x, 0, t)}{\epsilon} - \frac{u^1(x, 0, t) - u^1(x, -\epsilon, t)}{\epsilon} \tag{61}$$

eq. (56) *We suggest a* ***novel*** *modification to this calculation to reduce the instability of this estimate. Numerical experiments show that the boundary term (eq.* (58)*) contributes significant variance to the calculation of $\tilde{\theta}$ and leads to blow-up. There will be an un-physical build-up of velocity at the points where this blow-up occurs which reduces the smoothness of the resulting velocity field. For our simulation, we just use the main term (eq.* (56)*) and the force term (eq.* (57)*) for computing u^1, u^2 in (eq.* (61)*), and neglect the boundary term (eq.* (58)*). See figure (fig. 6) for a demonstration of how this change reduces the instability of this calculation of θ.*

An alternative method of calculating θ was introduced in [4]. The idea is to differentiate the formula for $\boldsymbol{u}$, as in proposition (Prop 3), with respect to x and y so as to obtain a random vortex approximation for $\frac{\partial \boldsymbol{u}}{\partial x}$ and $\frac{\partial \boldsymbol{u}}{\partial y}$. These partial derivatives can be passed through the integrals and expectations to act on the kernel $\boldsymbol{K}_D$. Then the formula $\omega = \frac{\partial u^2}{\partial x} - \frac{\partial u^1}{\partial y}$ is used to calculate the vorticity. While this method produces a more accurate estimation of the boundary vorticity θ, it is also more computationally expensive and works best with an analytic expression for the partial derivatives of the kernel which can be cumbersome to derive.

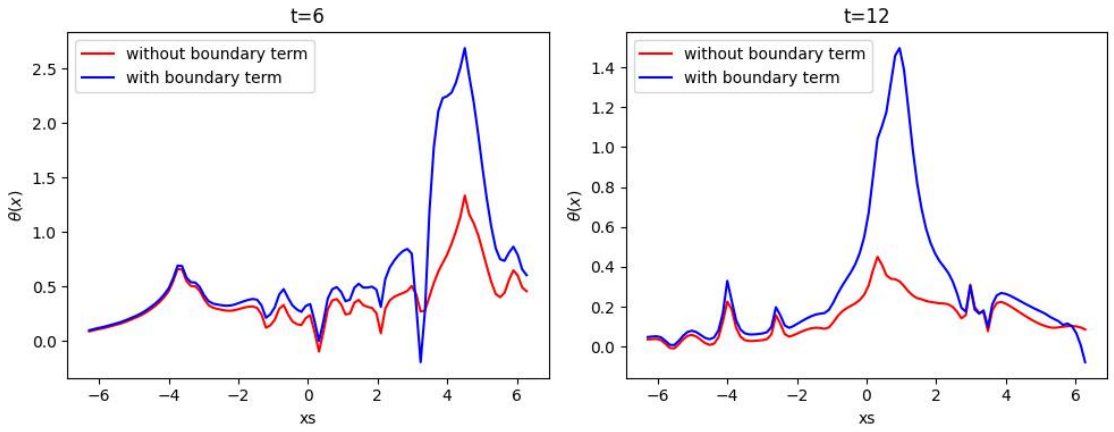

Figure 6: A comparison of numerically computed $\theta(x,t)$ both with and without using the boundary term (eq. (58)). These plots are given for two timesteps of the simulation (Ex 7). Observe how the boundary term introduces instability into the calculation of θ.

3.3.2 Simulation results

We implemented this method using the novel vectorised algorithm (algorithm 2) in Python. Below, some results of this simulation are presented.

Example 7. ***Constant force.*** *Consider the following simulation setup for solving the Navier-Stokes equations on the domain $D = \{y < 0\}$. $Re = 25000$, $\nu = 0.15$, $L = 2\pi$, $U_0 = \nu Re/L \approx 600$. The domain is $[-L, L] \times [-L, 0]$, with $\Delta t = 0.0012$, and the independent discretisation approach is used (section 2.5.2). The cutoff function ϕ (eq. (60)) and initial velocity $\boldsymbol{u}(\boldsymbol{x}, 0)$ is taken from [19]. However, we provide a novel extension to this system with a constant force $G_{\boldsymbol{F}} = 5000$.*

$$\boldsymbol{u}(x, y, 0) = U_0\,(-sin(y), 0)$$

The results are presented in figure (fig. 7). Observe how the applied force quickly accelerates the entirety of the fluid until the initial velocity distribution becomes irrelevant and all fluid is moving in the positive x direction. After about $0.15s$ nearly all the random vortices have left the simulation domain. Also, once again, the no-slip condition is satisfied on the boundary as desired.

Example 8. ***Vorticity Diffusion.*** *An interpretation of the Random Vortex Method is that the random vortices diffuse vorticity throughout the simulation. Each random vortex carries with it some vorticity $\omega(\boldsymbol{\eta})$ from the initial vorticity distribution. It is then convected by the fluid velocity, and diffused by the fluid vorticity according to SDE (eq. (20)). To better understand this interpretation, we present a plot in figure (fig. 8) of the evolution of the fluid vorticity $\omega(\boldsymbol{x}, t)$. This plot uses the same simulation setup as example (Ex 7).*
This plot provides a very clear visualisation of the evolution of the fluid. One can observe how the fluid's motion is originally driven by the initial vorticity distribution. The two sections of initial vorticity $y \in [-\pi, 0]$, $y \in [-2\pi, -\pi]$ move in opposite directions until about around $t = 6.0$. From about $t = 6.0$ onwards, the force acting on the fluid becomes the main driver of motion and all the vorticity is transported in the positive x direction.

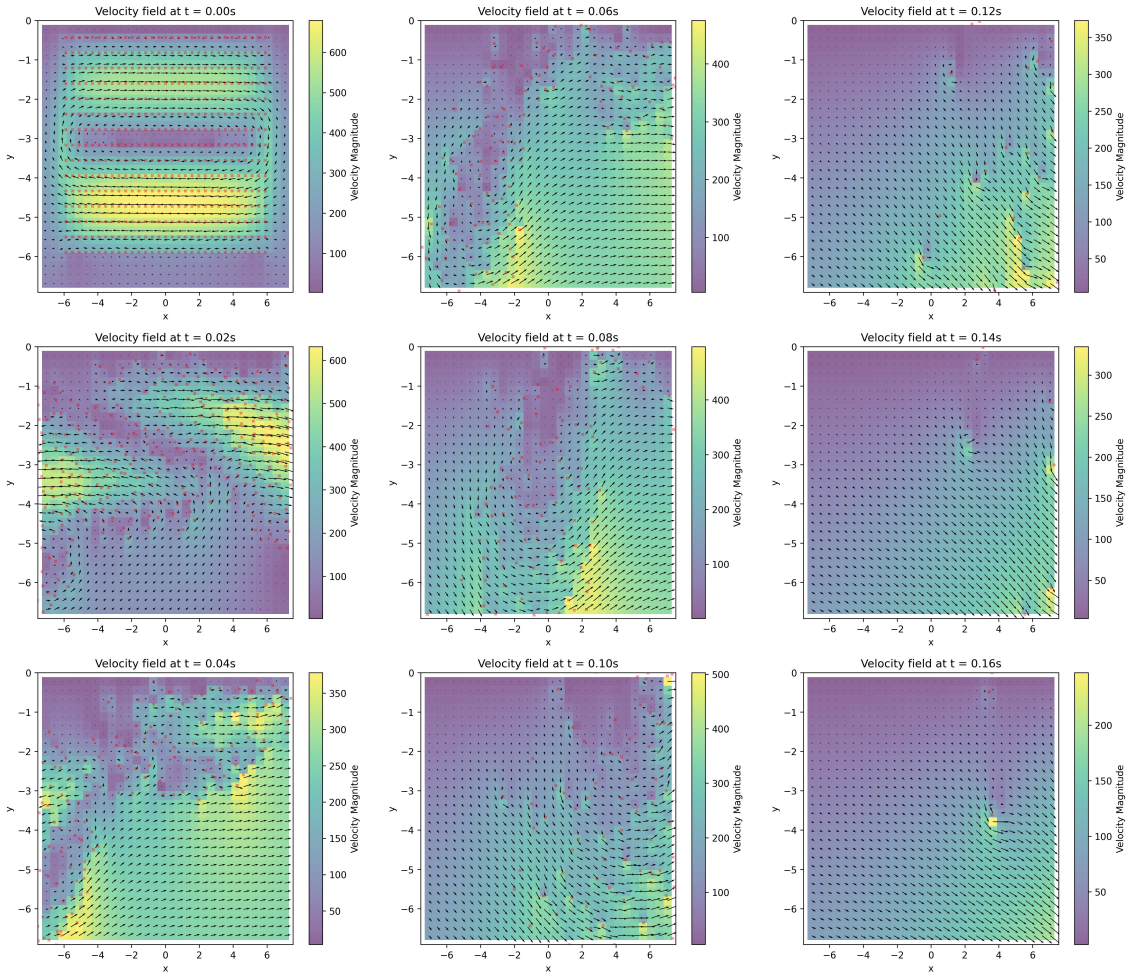

Figure 7: Simulation results for example (Ex 7). The vector field is the approximate velocity field $\tilde{\boldsymbol{u}}$, the red crosses are the random vortices, and the shading is for the velocity magnitude. The domain of these plots is slightly larger than that of the simulation. This simulation took 20 seconds to run.

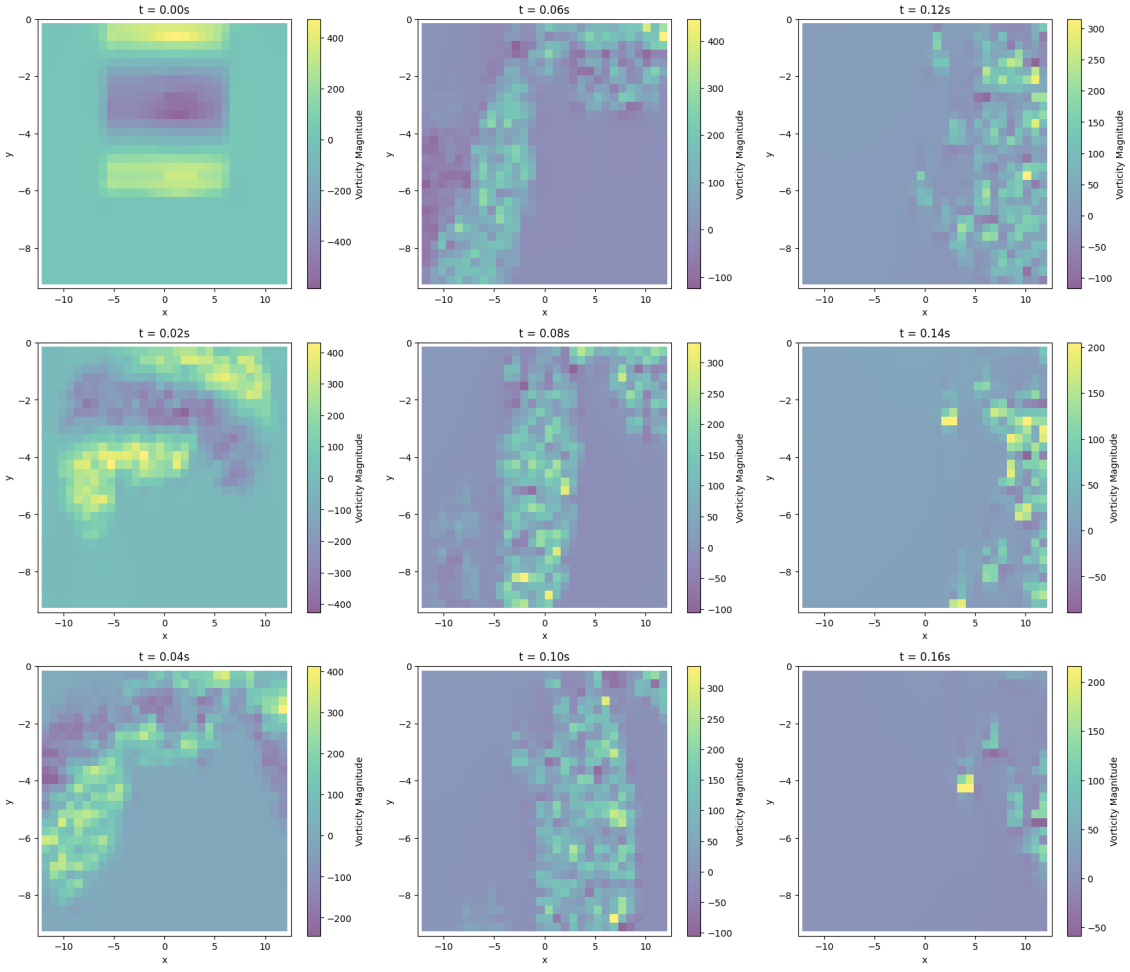

Figure 8: A plot of the evolution of the fluid vorticity $\omega(\boldsymbol{x}, t)$ from the simulation setup stated in example (Ex 7). This simulation took 20 seconds to run.

Algorithm 2 Discretising the ϵt_0 formulation.

INPUT: Mesh width h, number of paths M, timestep Δt, terminating time $T = K \times \Delta t$, for some $K \in \mathbb{N}$.

OUTPUT: The approximate velocity field $\tilde{\boldsymbol{u}}(\boldsymbol{x}, t)$ at $t = T$.

NOTATION: $\boldsymbol{X}_t^* = \{\boldsymbol{X}_t^{\boldsymbol{\eta}_m}\}_{m=1}^M$, $\boldsymbol{x}^* = \{\boldsymbol{x}_i\}_{i=1}^K$, $\boldsymbol{I}_t = I_t^{j\,M}{}_{m=1}$.

Step 1: Set $\boldsymbol{X}_0^{\boldsymbol{\eta}_j} = \boldsymbol{\eta}_j$ and $t = 0$.

Step 2: Define the set of vortices $\boldsymbol{X}_{s,\gamma(t)}^* = \{\boldsymbol{X}_t^{\boldsymbol{\eta}_m} \mid t - s < \gamma(\boldsymbol{X}_t^{\boldsymbol{\eta}_m}, t)\}$. Likewise for the vectorised initial vorticities $\boldsymbol{\omega}^*{}_{s,\gamma(t)} = \{\omega_0(\boldsymbol{\eta}_m) \mid t - s < \gamma(\boldsymbol{X}_t^{\boldsymbol{\eta}_m}, t)\}$

Step 3: Compute the first velocity term using the vectorised $\boldsymbol{K}^*$ as in algorithms (algorithm 3) and (algorithm 4).

$$\boldsymbol{U}_1 = h * \mathrm{SUM}(\boldsymbol{K}^*(\boldsymbol{x}^*, \boldsymbol{X}_{0,\gamma(n\Delta t)}^*) * \boldsymbol{\omega}^*{}_{0,\gamma(n\Delta t)}, \mathrm{axis} = 1)$$

Step 3:. Update the incremental sums for the force and boundary terms $I_{n\Delta t}^j$, as in proposition (Prop 4).

Step 4:. Computer the boundary velocity term.

$$\boldsymbol{U}_2 = h * \Delta t * \mathrm{SUM}(\boldsymbol{K}^*(\boldsymbol{x}^*, \boldsymbol{X}_{n\Delta t}^*) * \boldsymbol{I}_{n\Delta t}, \mathrm{axis} = 1)$$

Step 5:. Computer the force term U_3.

Step 6:. Return the velocity. $\tilde{\boldsymbol{u}}(\boldsymbol{x}, t) = \boldsymbol{U}_1 + \boldsymbol{U}_2 + \boldsymbol{U}_3$.

4 Boundary Layer Turbulence

This chapter contains details of our investigation of how the Random Vortex Method can be used to investigate the behaviour of boundary layers within a fluid. Our key contribution in this chapter is proposition (Prop 5) which suggests how the RVM can be used to isolate the effect of a boundary on fluid flow. Of particular interest is the case of a turbulent boundary layer. The concept of a boundary layer was introduced by Prandtl in [18]. Boundary layers are a thin layer close to an obstacle which ensures that viscous fluids satisfy the no-slip condition. One can expect to observe turbulence in a fluid with a Reynolds number greater than 4000. Turbulent flows are characterised by chaotic and irregular motion which make them challenging to model. See [17] for a review of turbulence. Traditional methods of modelling turbulent flows, such as the finite-difference method, are very computationally intensive due to the small spatial and temporal timesteps necessary to produce turbulent effects. The idea of using the Random Vortex Method to model turbulence was first proposed by Chorin in his foundational paper on the method [5] and it has since been demonstrated that the Random Vortex Method is well suited to modelling turbulent effects [10]. In this section, we show that the Random Vortex Method seems to produce key behaviours of turbulent fluids such as *vortex shedding* and *boundary layer separation*.

Another unique strength of the Random Vortex Method is that it analytically separates the velocity field into three separate components. Consider the $t\epsilon_0$ formulation in (*Prop* 3), which is used exclusively in this section due to its computational efficiency. This formulation decomposes the velocity field into three terms; a term due to the initial vorticity distribution (eq. (56)), a term due to force acting on the fluid (eq. (57)), and a term due to effects of boundary vorticity (eq. (58)). These terms are hereby named the *main term* $\boldsymbol{u}_M$, *force term* $\boldsymbol{u}_F$ and *boundary term* $\boldsymbol{u}_B$ respectively. The velocity field can hence be written as $\boldsymbol{u} = \boldsymbol{u}_M + \boldsymbol{u}_F + \boldsymbol{u}_B$. If instead the Navier-Stokes equations were solved using the finite difference method, there would be no clear way of decomposing the velocity field into *main*, *force* and *boundary* terms. This is due to the complex non-linear way in which these three factors interact in the Navier-Stokes equations. It is a unique and fascinating feature of the RVM that it can decompose the velocity field in this fashion. Further, it should be noted that the RVM, and thus this decomposition is *exact*. No heuristic approximations are necessary. In this section, the *boundary term* (eq. (58)) is analysed.

Proposition 5. *We propose that the boundary term (eq.* (58)*) provides a lens through which to isolate and investigate the effect of a boundary on fluid flow. Hence this term can be studied to investigate the phenomena of fluid boundary layers.*

For convenience, the ϵt_0 formulation is restated below. In this section, we present results of our simulations of the Random Vortex Method on a new domain - flow around a cylinder. In the forthcoming examples, plots will be presented which show either all the velocity terms, or just the *boundary term* (*eq.* (51)). The idea of plotting just the *boundary term* to investigate the boundary layer follows from our proposition (Prop 5). In this section, the flow will be illustrated with streamline plots. In previous examples, a vector field plot was used to show the fluid velocity. This becomes unsuitable however when plotting the flow in the boundary layer due to the complex nature of the flow.

$$\begin{aligned}\boldsymbol{u}(\boldsymbol{x},t) = &\int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\boldsymbol{\eta}})\mathbb{1}_{\{t<\gamma(\boldsymbol{X}^{\boldsymbol{\eta}},t)\}}\right]\omega(\boldsymbol{\eta},0)d\eta^1 d\eta^2 \\ &+ \int_0^t \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\boldsymbol{\eta}})G_{\boldsymbol{F}}(\boldsymbol{X}_s^{\boldsymbol{\eta}},s)\mathbb{1}_{\{t-s<\gamma(\boldsymbol{X}_t^{\boldsymbol{\eta}},t)\}}\right]d\eta^1 d\eta^2 ds \\ &+ \frac{\nu}{\epsilon^2}\int_0^t \int_D \mathbb{E}\left[\boldsymbol{K}_D(\boldsymbol{x},\boldsymbol{X}_t^{\boldsymbol{\eta}})\phi''(-\boldsymbol{X}_s^{\boldsymbol{\eta}}/\epsilon)\theta(\boldsymbol{X}_s^{\boldsymbol{\eta}},s)\mathbb{1}_{\{t-s<\gamma(\boldsymbol{X}_t^{\boldsymbol{\eta}},t)\}}\right]d\eta^1 d\eta^2 ds\end{aligned}$$

4.1 Turbulent boundary layer

In this section, simulation results are presented which illustrate how the behaviour of turbulent boundary layers can be studied. Following our proposition (Prop 5), we investigate the behaviour of boundary layers by considering the boundary term (*eq.* (51)). Turbulence is usually generated by a Reynolds number of at least 4000. Consider the following simulation setup for the half-plane D.

Example 9. ***Turbulent Boundary Layer past a flat plane****. Consider the Navier-Stokes equations for the half-plane (Dfn 5), and fix* $Re = 10000$, $\nu = 0.15$, $L = 2\pi$, $U_0 = \nu * Re/L \approx 240$. *Take the domain* $[-L, L] \times [-L, 0]$, *and set* $\omega_0 = -U_0$. *This corresponds to an initial velocity field* $\boldsymbol{u}(x, y, t) = U_0(-y, 0)$.

The ϵt_0 *formulation (Prop 3) is used with* $\Delta t = 0.01$, ϕ *is given by (eq.* (60)*) and* $\epsilon = 0.5$. *The Random Vortices are initialised on two separate grids. The first grid of vortices is of shape* $(30, 25)$ *on the domain* $[-L, L] \times [-L, -eps]$ *to simulate the outer flow, and the other is of shape* $(30, 15)$ *on the domain* $[-L, L] \times [-eps, 0]$ *to simulate the boundary layer. These 'boundary layer vortices' are introduced so that the fluid can be simulated to a higher resolution in the boundary*

layer. The independent discretisation method (section 2.5.2), where each random vortex is driven by an independent Brownian motion, is used. This simulation took 100 seconds to execute.

A challenge to overcome with simulations of a turbulent boundary layer is that the outer flow must be very fast moving as by definition of the Reynolds number, $Re \propto [U]$, where $[U]$ is the typical velocity scale of the fluid. This is problematic as this large velocity will mean the Random Vortices in the outer flow will quickly leave the simulation domain. A solution to this problem is to introduce periodic boundary conditions at $x = \pm L$ whereby each vortices x-position is updated according to $x \leftarrow\!\shortmid (x+L) \pmod{2L} - L$. This periodic boundary condition helps to ensure the outer flow remains sufficiently rapid so that the behaviour of the boundary layer can be studied. One limitation of this periodisation is that random vortices can still leave the domain in the negative-y direction. Over time, these escaping vortices cause the magnitude of the outer flow velocity to decrease. A possible future extension to this work would be to modify the Biot-Savart Kernel $\boldsymbol{K}$ to include these periodic boundary conditions.

The results from example (Ex 9) are now discussed. Figure (fig. 11) shows the flow produced by the *boundary term* (eq. (58)) in a region near the boundary, figure (fig. 12) shows the flow produced by all velocity terms in a region near the boundary, figure (fig. 9) shows the flow in the outer region produced by all velocity terms, and figure (fig. 10) shows the flow in the outer region produced by the *boundary term*.

Outer region, all velocity terms. Consider the outer flow in figure (fig. 9). Firstly, as desired, the periodic boundary conditions enforce a quasi-uniform outer flow in the negative x-direction. This outer flow should be sufficiently rapid to observe turbulent effects at the boundary. Observe how the magnitude of the fluid velocity in the outer flow decreases over time. This can be ascribed to random vortices leaving the fluid domain in the negative-y direction. One can immediately observe how the uniform streamlines become chaotic at the fluid boundary.

Outer region, boundary term. Consider now the *boundary term* flow in the fluid bulk, in figure (fig. 10). The purpose of this plot is to illustrate that the boundary term is most significant near the boundary and has little effect on the fluid bulk. Observe how the large velocity magnitudes are concentrated at the boundary and are shed into the fluid bulk.

Boundary layer, boundary term. Consider now the *boundary term* flow in a layer close to the boundary $y = 0$, as in figure (fig. 11). Firstly, at $t = 0$ the boundary velocity field is initially zero. This is because the *boundary term* is an integral over $[0, t]$, so initially vanishes. Secondly, observe the chaotic nature of the flow as depicted by the streamlines. This corresponds to the irregular nature of fluid in a turbulent boundary layer. Interestingly, the flow appears to demonstrate *vortex shedding* where small regions of high vorticity build up in the boundary and are shed into the fluid bulk. These can be identified as the bright yellow 'blobs' in the plot. It is encouraging that the *boundary term* seemingly produces a classical behaviour of boundary layers. This provides key supporting evidence for our proposition (Prop 5).

Boundary layer, all velocity terms. Consider now the complete fluid flow from all velocity terms in a layer close to the boundary, as in figure (fig. 12). This plot is useful as it indicates how the *boundary term* behaves in comparison to the main term in the boundary layer. Observe firstly that the typical scale of the magnitude of the velocity field from the *main term* is significantly larger than that of the *boundary term* in the boundary layer. In the former case, figure (fig. 12) suggests a typical scale of about 60, whereas figure (fig. 11) suggests a scale of about 15. One possible explanation for this could be that as the *boundary term* (eq. (58)) is a time integral,

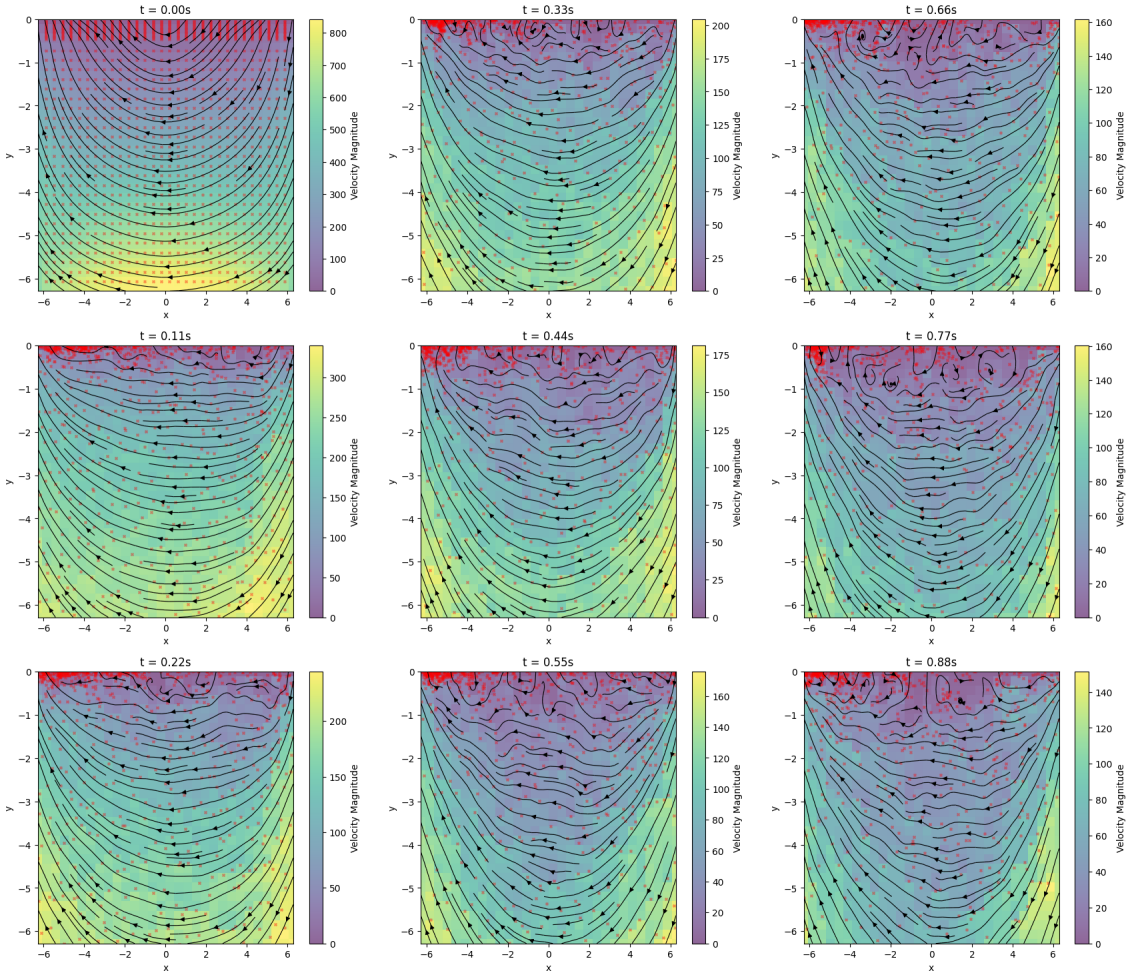

Figure 9: A plot of the fluid flow from all velocity terms for example (Ex 9) in the fluid bulk.

it will become more significant over large timescales. This simulation however only runs up until $t = 0.88$. Notice how the flow becomes increasingly irregular over time. Also, the colourmap of velocity magnitude shows a smoother distribution than with the *boundary term* where regions of large velocity magnitude formed into 'blobs'.

In this simulation, the flow from the *main term* is more significant in the boundary layer than that of the *boundary term*. Despite this, our interest remains focused on studying the *boundary term* in this region. The first reason for this is that in other simulations the boundary term could become more significant. For instance, the following changes will increase the magnitude of the *boundary term* (eq. (51)); decreasing ϵ, increasing ν, and running the simulation over a larger time horizon. The second reason, as proposed in (Prop 5), is that we believe that the boundary layer term is what gives rise to commonly observed boundary layer effects such as *vortex shedding*. This belief is further motivated by the apparent vortex shedding demonstrated in figure (fig. 11).

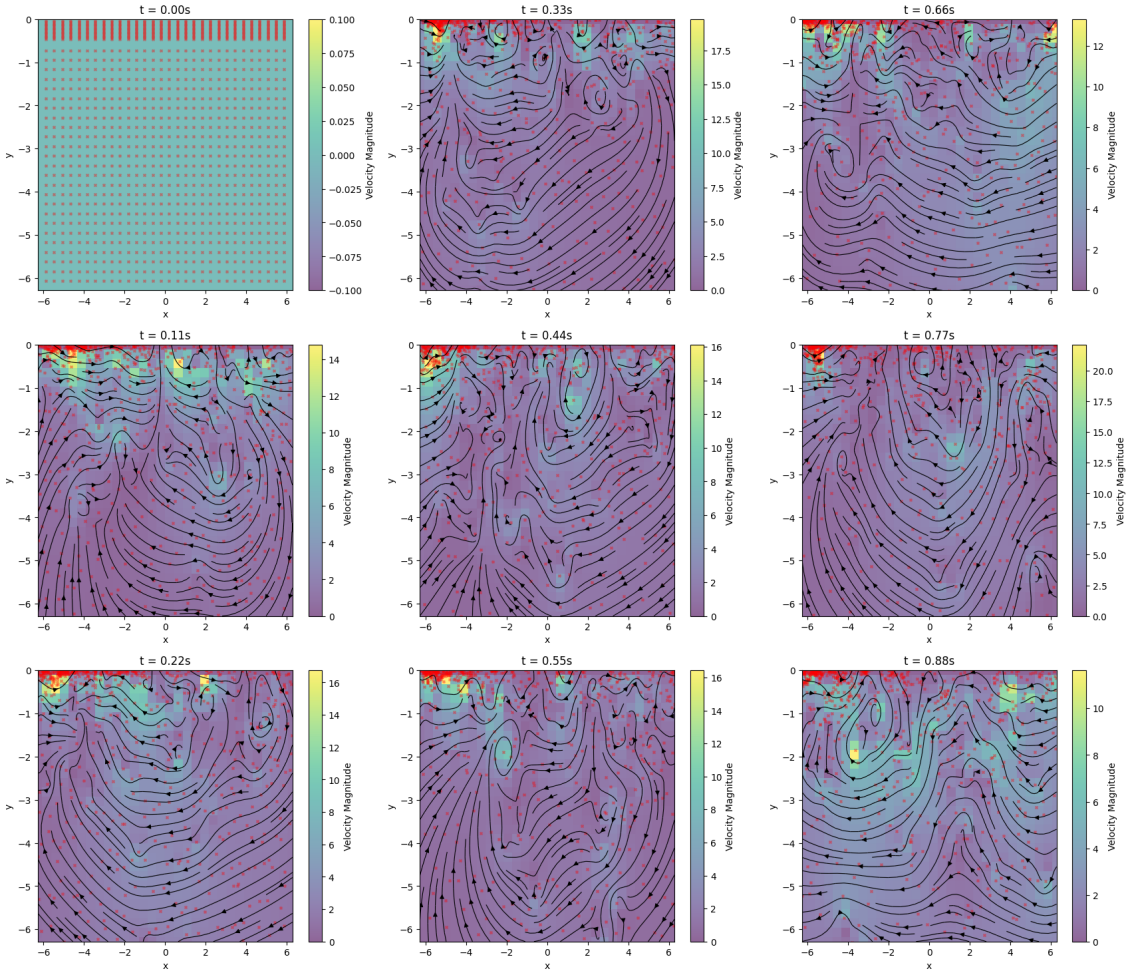

Figure 10: A plot of the fluid flow from the *boundary term* for example (Ex 9) in the fluid bulk.

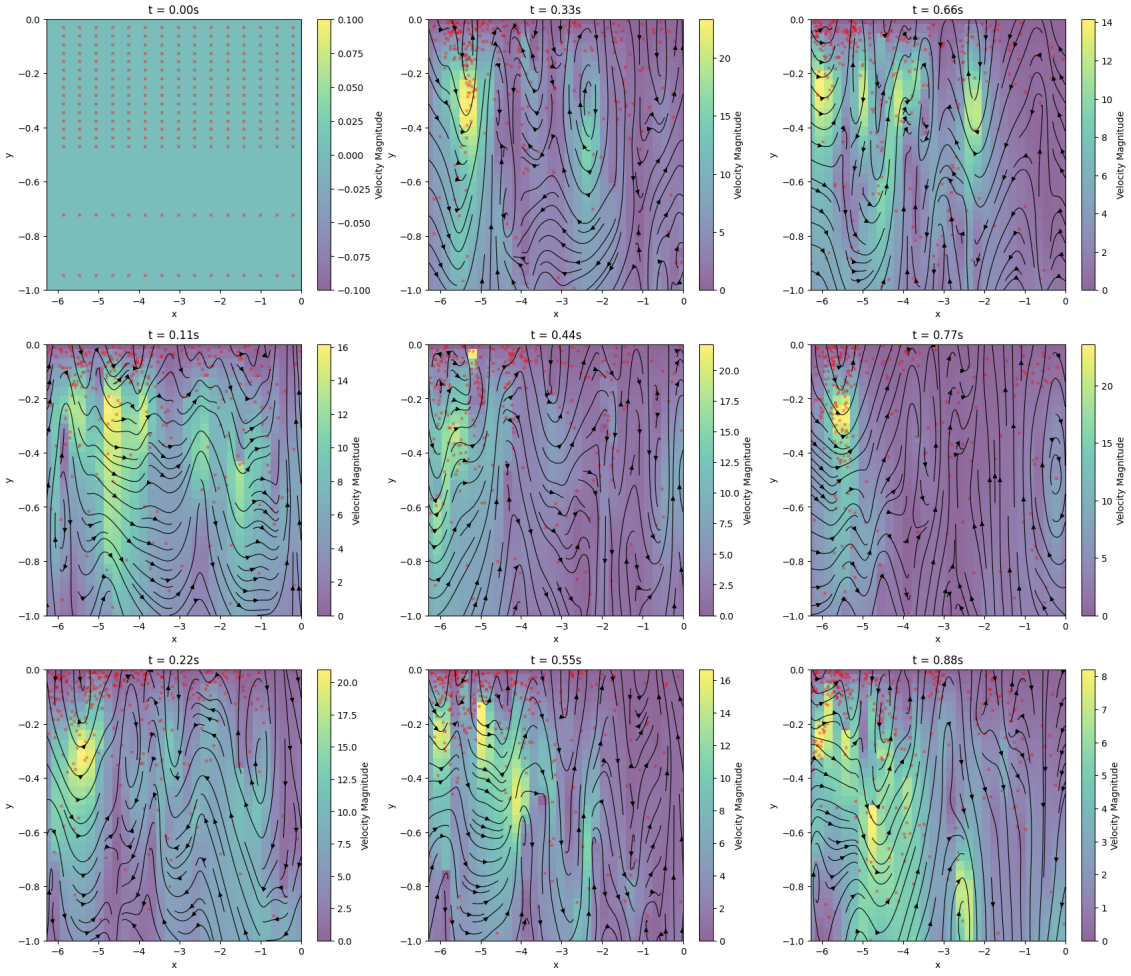

Figure 11: A plot of the fluid flow from the *boundary term* for example (Ex 9) in a region close to the boundary.

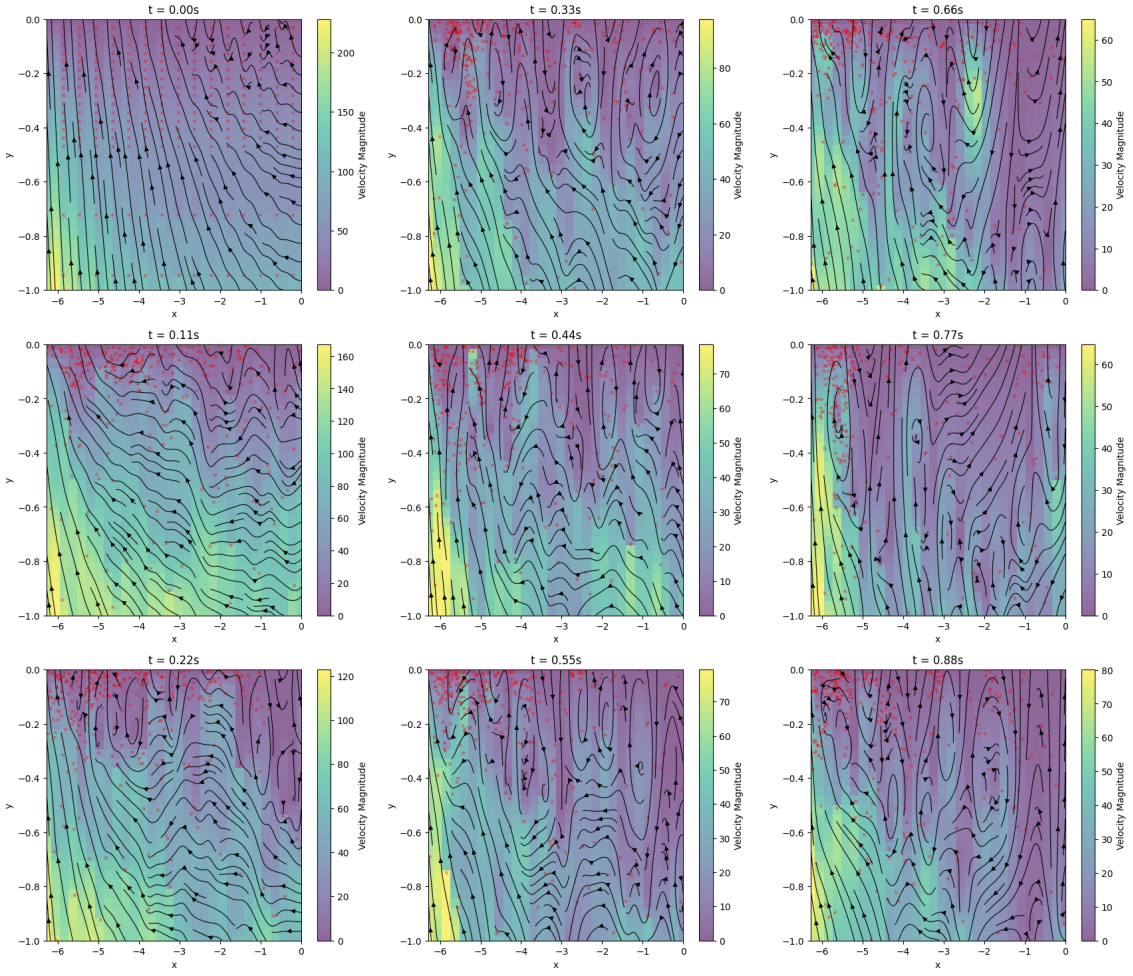

Figure 12: A plot of the fluid flow from all velocity terms for example (Ex 9) in a region close to the boundary.

4.2 Random Vortex Method on a General Domain

In sections (section 2), (section 3) derivations for the Random Vortex Method were presented on $\mathbb{R}^2$ and the half-plane $D = \{y < 0\}$. In [4], the Random Vortex Method was generalised to all simply connected domains $C \subseteq \mathbb{R}^2$. While we refer to the author's paper for the full derivation, some details are described here. Reimann's Mapping Theorem says that given any simply connected subset $C \subseteq \mathbb{R}^2$, there exists a conformal mapping $\boldsymbol{T} : C \mapsto D$, where D is the lower half-plane. Using this mapping, the elliptic Green's function on C can be written as $G_C = G(\boldsymbol{T}(\boldsymbol{x}), \boldsymbol{T}(\boldsymbol{y})) - G(\boldsymbol{T}(\boldsymbol{x}), \overline{\boldsymbol{T}(\boldsymbol{y})})$, where G is the elliptic Green's function on $\mathbb{R}^2$. The paper uses this expression to derive a formula for the Biot-Savarat kernel (their Lemma 3.1). For their simulations, the authors considered the domains of a semi-infinite flat plate $\mathbb{R}^2 \backslash \{(x, y) \mid x > 0, y = 0\}$, and a wedge $\{(x, y) \mid \arctan(y/x) \in [-\alpha, \alpha]\}$.
For this paper, we have derived a new implementation of the Random Vortex Method on the domain $S = \mathbb{R}^2 \backslash \{(x, y) \mid x^2 + y^1 < 1\}$. This domain represents *flow past a cylinder*. The behaviour of viscous uniform flow flow past a cylinder is well known. Depending on the Reynolds number, one can expect to observe a turbulent wake, vortex shedding, and boundary layer separation. The conformal mapping $\boldsymbol{T}$ for this domain is as follows.

$$\boldsymbol{T}(x, y) = \frac{(-2y, x^2 + y^2 - 1)}{(1 - x)^2 + y^2}$$

We adapt the ϵt_0 formulation of the Random Vortex Method to this domain in a **novel** manner. The first term (*eq.* (51)), and second term (eq. (50)) remain the same. To implement the third term, we introduce two changes. Firstly, a redefinition of $\phi''(-\boldsymbol{X}^{\boldsymbol{\eta}_j}_{i\Delta t}/\epsilon)$. Define the distance to the boundary as $r_S(\boldsymbol{x}) := \inf_{\boldsymbol{y} \in \partial S} \|\boldsymbol{x} - y\|_2$. Then recalling that ϕ is a scalar function, $\phi''(-\boldsymbol{x})$ is defined as $\phi''(r_S(\boldsymbol{x}))$. Likewise, $\theta(\boldsymbol{x})$ is defined as $\theta(P_S(\boldsymbol{x}))$ where is the P_S projection function onto the boundary ∂S. In the case of a circular boundary $\partial S = \{x^2 + y^2 = 1\}$, this is simple as $P_S(\boldsymbol{x}) = \boldsymbol{x}/\|\boldsymbol{x}\|_2$, and $r_S(\boldsymbol{x}) = 1 - \|\boldsymbol{x}\|_2$.

Example 10. ***Boundary Layer Separation and Turbulent Wake****. The following example exhibits boundary layer separation and a turbulent wake for flow past a cylinder. Consider the initial vorticity distribution* $\omega_0(x, y) = -U_0 * y$ *where* $U_0 = 750$*. This corresponds to a Reynolds number of about* 10000*. The timestep is* $\Delta t = 0.0008$ *and the independent discretisation approach is used (section 2.5.2). Further, an extra set of random vortices are initialised close to the boundary to simulate the boundary layer in more detail. Once again, periodic boundary conditions are used to sustain the rapid outer flow so that turbulent effects at the boundary can be studied.*

The results of this simulation are discussed below. For further discussion of the characteristics of fluid flow past a cylinder see [9].

The outer flow. Figure (fig. 13) illustrates the flow of all velocity terms in the fluid bulk. This plot replicates the classical phenomena of *boundary layer separation* at the trailing edge of an obstacle in uniform flow. This can be seen where the streamlines which follow the boundary of the cylinder at $y = 1$ and $y = -1$ detach from the cylinder at the trailing edge and flow in the positive x-direction. Boundary layer separation typically results in a *turbulent wake* which is an area of irregular fluid motion at the trailing edge of the obstacle. Here this can be observed through the complex streamlines at about $(x, y) = (1, 0)$. It is encouraging that the Random Vortex Method replicates these well-known behaviours of viscous flow past a cylinder.

The turbulent wake. Figure (fig. 14) zooms into a region about $(x, y) = (1, 0)$ to investigate the turbulent wake. Specifically, the *boundary term* is plotted. As expected, the boundary term produces chaotic and irregular streamlines close to the boundary and has little effect on the flow in the fluid bulk, which is here characterised by uniform streamlines.

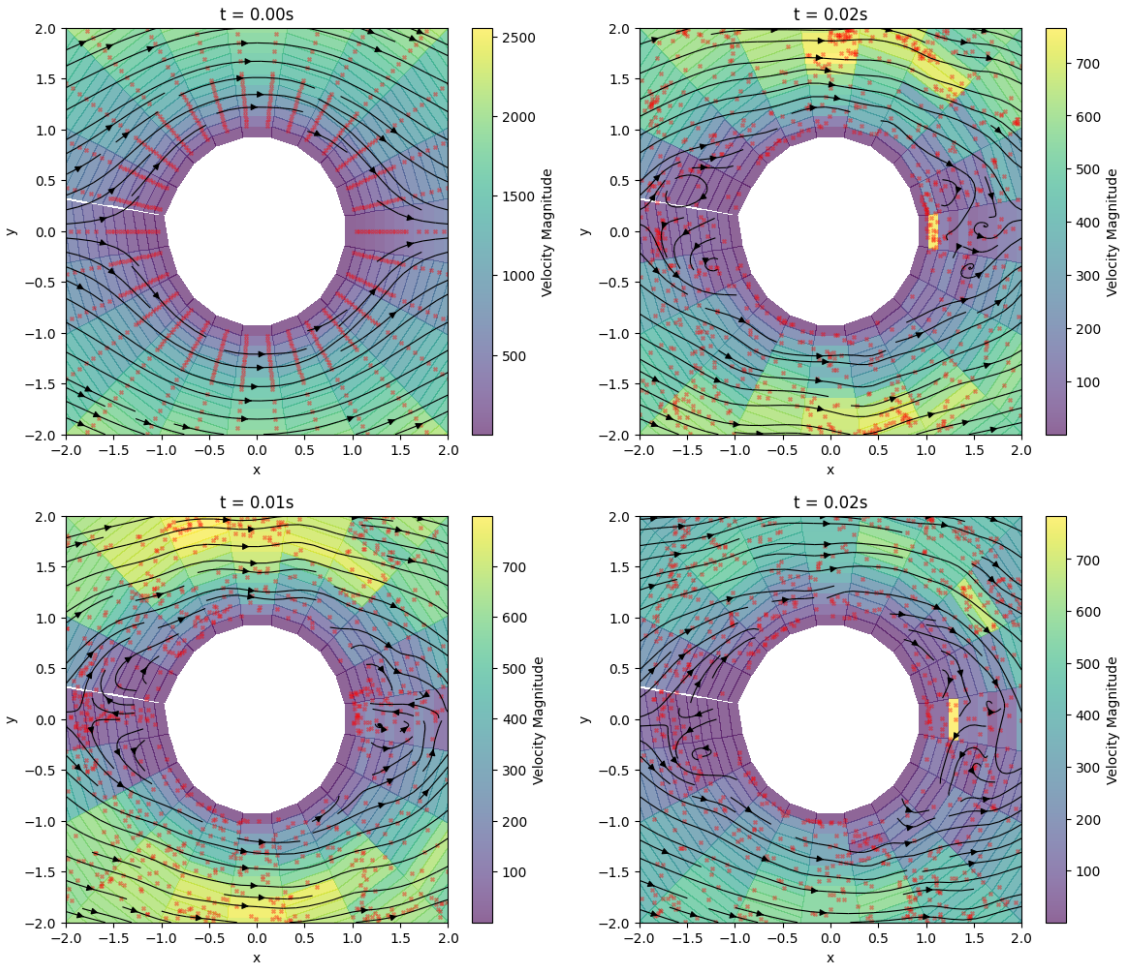

Figure 13: A plot of the fluid flow from all velocity terms in example (Ex 10).

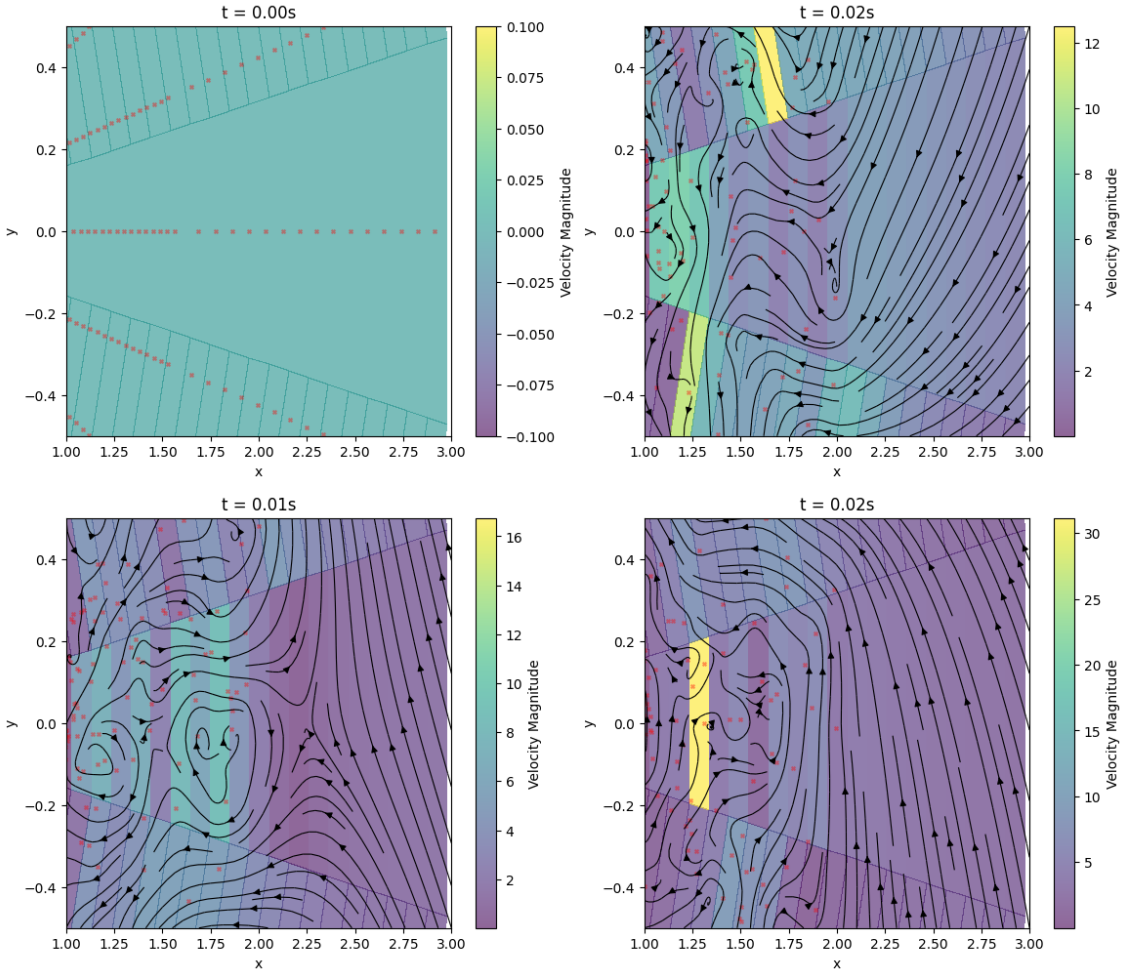

Figure 14: A plot of the fluid flow from the *boundary term* in example (Ex 10). This plot is magnified to show the flow in the turbulent wake.

5 Conclusion

In this paper, we demonstrated that the Random Vortex Method is an accurate and computationally efficient method for simulating fluid flow. We presented derivations for the RVM on $\mathbb{R}^2$ (section 2), and the half-plane $\{y < 0\}$ (section 3). Simulation results were presented on these domains as well as past a cylinder $\{x^2 + y^2 > 1\}$ (section 4.2). Particular attention was given to the application of the RVM to studying turbulent boundary layers (section 4). We suggested a novel technique (Prop 5) for applying the RVM to the investigation of turbulent boundary layers. Simulation results supported the efficacy of this technique by producing classical boundary layer effects like vortex shedding (Ex 9). Future work on the RVM could involve; further study into the behaviour of the boundary term; tackling the problem of random vortices leaving the simulation domain without the use of periodic boundary conditions; building simulations on more complex domains; and considering flow around obstacles in three dimensions.

A Vectorisation

The Random Vortex Method is fairly computationally intensive. Consider algorithm (algorithm 1). In total, there are $N_v := N*M$ random vortices. The RVM scales at $\mathcal{O}(N_v^2)$ as interactions between each pair of random vortices must be computed. To ensure this simulation is executable within a sensible time frame we utilised *vectorised operations* through the Python library *NumPy* [7]. NumPy allows scalar functions to be generalised to Universal Functions (ufuncs) which act element-wise on array elements. Ufuncs are very fast when the same operation needs to be applied to multiple elements of data. In algorithm (algorithm 3) the kernel $\boldsymbol{K} : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^2$ is vectorised to $\boldsymbol{K}^* : (\mathbb{R}^2)^p \times (\mathbb{R}^2)^q \to (\mathbb{R}^2)^{(p\times q)}$. Then, in algorithm (algorithm 4), this vectorised kernel is used to vectorise the calculation of the velocity field $\tilde{\boldsymbol{u}} : \mathbb{R}^2 \to \mathbb{R}^2$ to $\tilde{\boldsymbol{u}}^* : (\mathbb{R}^2)^p \to (\mathbb{R}^2)^p$.

Algorithm 3 Vectorising $\boldsymbol{K}$ (eq. (22)).

INPUT: Vectorised cutoff function $f_\delta : \mathbb{R}^r \mapsto \mathbb{R}^r$, positions $\boldsymbol{x}^* = \{\boldsymbol{x}_i\}_{i=1}^p, \boldsymbol{y}^* = \{\boldsymbol{y}_i\}_{i=1}^q$ where $\boldsymbol{x}_i, \boldsymbol{y}_i \in \mathbb{R}^2$ and $r, p, q \in \mathbb{N}_{\geq 1}$.

OUTPUT: The kernel $\boldsymbol{K}^*(\boldsymbol{x}^*, \boldsymbol{y}^*)$.

Step 1: Tile a $(p, q, 2)$ dimensional array $A_{\boldsymbol{x}^*}$ so that each of its rows is $\boldsymbol{x}^*$.

Step 2: Do the same for another $(p, q, 2)$ dimensional array $A_{\boldsymbol{y}^*}$ so that each of its columns is $\boldsymbol{y}^*$.

Step 3: Take the element-wise difference $B = A_{\boldsymbol{x}^*} - A_{\boldsymbol{y}^*}$.

Step 4: Apply the Euclidean norm along the last axis for $(p, q, 1)$ dimensional $R = \text{NORM}(B, \text{axis} = -1)$.

Step 5: To avoid division by zero, take the elementwise maximum of R with some EPS > 0, $R = \text{MAX}(R, \text{eps})$.

Step 6: Flip the positions $[a, b]$ in B to $[-b, a]$. $D = \text{STACK}(-B[..., 1], B[..., 0], \text{axis} = -1)$.

Step 7: Output the following elementwise multiplication, $\boldsymbol{K}^*(\boldsymbol{x}^*, \boldsymbol{y}^*) = D \times f_\delta(R) \times \frac{1}{2\pi} * R^{-2}$ where R^{-2} denotes the elementwise-square-reciprocal of R.

Algorithm 4 Vectorising the $\tilde{\boldsymbol{u}}$ (eq. (22)).

INPUT: Random vortices $\boldsymbol{X}^* = \{\{\boldsymbol{X}_t^{\boldsymbol{\eta}_m,(n)}\}_{m=1}^M\}_{n=1}^N$, and positions $\boldsymbol{x}^* = \{\boldsymbol{x}_i\}_{i=1}^K, \boldsymbol{x}_i \in \mathbb{R}^2$

OUTPUT: The approximate velocity field $\tilde{\boldsymbol{u}}^*(\boldsymbol{x}^*, t)$.

Step 1: Calculate $K = \boldsymbol{K}^*(\boldsymbol{x}^*, \boldsymbol{X}^*)$.

Step 2: Define the $(m, 1)$ array S of random vortex strengths such that $S_i = \omega_0(\boldsymbol{\eta}_i)/N$.

Step 3: Multiple each row of K elementwise with S for $U = K * S$.

Step 4: Output the columnwise sum $\tilde{\boldsymbol{u}}^* = \text{SUM}(U, \text{axis} = 1)$

B Code

The code for this project is entirely our own and written in Python. In this section, a high-level overview of our code is presented, along with an example of how a simulation can be configured and executed. We have implemented several versions of the Random Vortex Method: the Random Vortex Method on $\mathbb{R}^2$ (Thm 2), the Random Vortex Method on the half-plane $D = \{y < 0\}$ using the ϵt_0 formulation (Thm 5), the Random Vortex Method on D using (Prop 3), and the Random Vortex Method on a general simply connected domain as per [4.2].

The most important class in our program is the `RandomVortexSolver`. This is an Abstract Base Class which defines a blueprint for each implementation of the Random Vortex Method. It defines several methods which are common to all implementations of the Random Vortex Method, several of which are listed below.

- `u(self, x: np.ndarray, ...)` calculates the fluid velocity at a position `x`.
- `def omega(self, x: np.ndarray, ...)` calculates the fluid vorticity at a position `x`.
- `def run_simulation(self, iterations: int, ...)` runs the simulation for a given number of iterations.
- `def build_plot(...)` outputs a configurable plot of the fluid flow.

The random vortices are stored in the `VectorisedRandomVortices` class. This class' attributes consist of `ndarrays` where each row corresponds to a single random vortex. This structure is key as it enables a vectorised simulation. Below is our implementation of the vectorised Kernel algorithm (algorithm 3) as discussed in (appendix A).

```
def k_R2(self, x: np.ndarray, y: np.ndarray, eps: float = 0.00001) -> np.ndarray:
        x, y = np.atleast_2d(x), np.atleast_2d(y)

        # broadcast to matrices
        n, m = len(x), len(y)
        matrix_x = np.tile(x, (m, 1, 1))
        matrix_y = np.tile(y, (n, 1, 1)).transpose(1, 0, 2)

        # evaluate the kernel
        z = matrix_x - matrix_y
        r = np.linalg.norm(z, axis=-1)
        r = np.maximum(r, eps)  # avoid division by zero error
        transformed_points = np.stack((-z[..., 1], z[..., 0]), axis=-1)
        result = transformed_points * (
                self.cutoff_function(r, self.cutoff_radius) / (2 * np.pi * r * r)
        )[...,np.newaxis]

        # reduce dimension
        return np.squeeze(result)
```

The following code is an example of how a simulation can be configured and executed. The set-up is as follows. A grid of $30 * 25$ random vortices is initialised on the domain $[-L, L] \times [-L, 0]$, with initial vorticity distribution $\omega_0(\boldsymbol{x}) = (0, -U_0)$, using the ϵt_0 approach as in (Prop 3). The simulation is then run for 100 iterations at a timestep of 0.01, and the fluid flow is outputted in graphical form.

```
# import libraries
from main import VectorisedRandomVortices, Theorem2pt4, Kernel
import numpy as np

# utility functions
def Chorin_cutoff(r, delta):
        # cutoff function for smoothing the kernel
    return np.where(r > delta, 1, r / delta)

def dd_phi(x: np.ndarray) -> np.ndarray:
        # second derivative of the cutoff function, as in the £\epsilon t_0£ formulation
    x = np.abs(x)[..., 1]
    return 54*3*2*(x-1/2) * ((x < 2/3) & (x > 1/3))

def initial_vorticity_distribution(xs: np.ndarray) -> np.ndarray:
    y = xs[..., 1]
    return np.full_like(y, -U_0)

# simulation parameters
viscosity = 0.15
Re = 10000  # Reynolds number
L = 2*np.pi  # length scale
eps = 0.5  # boundary layer thickness
domain = [[-L, L], [-L, 0.0]]  # simulation domain
U_0 = viscosity * Re / L  # velocity scale

# generate the random vortices in a 30x25 grid throughout the domain
# here, the independent random vortex method is used
vortices = VectorisedRandomVortices.gen_rectangular_grid(
    domain = domain,
    strength = initial_vorticity_distribution,
    viscosity = viscosity,
    x_grid_size = 30,
    y_grid_size = 25,
    estimate_expectation=False,
    num_simulations=1
)

# assemble the simulation
solver = EpsilonT0Solver(
    kernel=Kernel(Chorin_cutoff, 0.2),
    random_vortices=vortices,
    timestep=0.01,
    dd_phi=dd_phi,
    eps=eps,
        periodic_boundary_conditions=True
)

# run the simulation and plot results in a 2x2 grid
solver.run_simulation(iterations=100, iterations_per_frame=1)
```

```
solver.build_plot(rows=2, columns=2)
```

References

[1] D.J Acheson. *Elementary Fluid Dynamics*. Oxford University Press, 1990.

[2] C.C Chang. Random vortex methods for the navier-stokes equations. *Journal of Computational Physics*, 76(2):281–300, 1988.

[3] V Cherepanov, J.G Liu, and Z Qian. On the dynamics of the boundary vorticity for incompressible viscous flows. *Journal of Scientific Computing*, 99(42), 2024.

[4] V Cherepanov and Z Qian. Monte-carlo method for incompressible fluid flows past obstacles. 2023.

[5] A.J Chorin. Numerical study of slightly viscous flow. *Journal of Fluid Mechanics*, 57:785 – 796, 1973.

[6] A.J Chorin and P Bernard. Discretization of a vortex sheet, with an example of roll-up. *Journal of Computational Physics*, 13:423–429, 1973.

[7] Harris et al. Array programming with numpy. *Nature*, 585:357–362, 2020.

[8] L.C Evans. *Partial Differential Equations*, volume 19 of *Graduate Studies in Mathematics*. American Mathematical Society, 2 edition, 2010.

[9] R Franke, W Rodi, and B Schönung. Numerical calculation of laminar vortex-shedding flow past cylinders. *Journal of Wind Engineering and Industrial Aerodynamics*, 35:237–257, 1990.

[10] A.F Ghoniem, A.J Chorin, and A.K Oppenheim. Numerical modelling of turbulent flow in a combustion tunnel. *Philosophical Transactions of the Royal Society A*, 304, 1982.

[11] J Goodman. Convergence of the random vortex method. *Communications on Pure and Applied Mathematics*, 40:189–220, 1987.

[12] N Ikeda and S Watanabe. *Stochastic Differential Equations and Diffusion Processes 2nd Edition*, volume 24. North Holland Mathematical Library, 1992.

[13] K Kuwahara. Numerical study of flow past an inclined flat plate by an inviscid model. *Journal of the Physics Society of Japan*, 35:1545–1551, 1973.

[14] L.D Landau and E.M Lifshitz. *Fluid Mechanics: Volume 6*, volume 6. Pergamon, 1987.

[15] D-G Long. Convergence of the random vortex method in two dimensions. *Journal of the American Mathematical Society*, 1(4):779–804, 1988.

[16] A.M Majda and A.L Bertozzi. *Partial Differential Equations*, volume 19 of *Graduate Studies in Mathematics*. American Mathematical Society, 2 edition, 2010.

[17] SB Pope. *Turbulent Flows*. Cambridge University Press, 2000.

[18] L Prandtl. International Congress of Mathematicians, Heidelberg, 1904.

[19] Z. Qian, Y Qiu, L Zhao, and J Wu. Monte-carlo simulations for wall-bounded fluid flows via random vortex method. *arXiv*, 2022.

[20] G Van Rossum and F.L Drake. Python 3 reference manual, 2009. Available at https://docs.python.org/3/.