Ship traffic flow simulated by source panel method

by

Thomas Sjåstad

Thesis

for the degree of

Master of Science

Faculty of Mathematics and Natural Sciences
University of Oslo

August 2020

Abstract

A model simulating vessel trajectories based on the assumption that vessels can be represented as fluid particles is presented. Solving the Laplace equation to create flow fields to simulate flow around circular objects is carried out. The doublet potential is commonly used for circular object representation, we find that the source potential also is a good candidate for representing circular objects using the analytical solutions. By adjusting the strength coefficients of the potentials, we find that this affects the flow fields created and need to be set correctly in order to obtain satisfactory flows. For an increasing geometry complexity we tend to the source panel method. The source panel method is applied in Trondheimsfjorden compared with AIS data for verifying its performance. We find that source panel method is able to produce paths which mimic routes from AIS data, but only in certain areas and specific routes.

Acknowledgements

I would like to thank both my supervisors Morten-Hjorth Jensen and Thomas Mestl for their guidance and insight. This thesis would not have happened without you. Morten, your courses and motivation for teaching physics is nothing short of admirable. Thomas, thank you for providing different perspectives other than the academic discipline. A special thank you to Mai Nayeli Hauge, for your endless encouragement and patience. Thank you for guiding me through difficult times. I would also like to thank my good friend Eirik G. Ballo for your guidance and encouragement. Without you, this thesis would be considerably more difficult. Last, a thank you to my family (Adam, Martin, Lene and Mindor) for always believing in me.

Contents

Chapter 1

Introduction

1.1 Problem outline and motivation

Transportation on sea is still one of the biggest businesses within the global trade, as maritime shipping amounts to 90 % of the world's trade transportation method [1]. Route planning is central to obtain the route that is most favorable i.e. shortest (fuel saving), fastest (increase profit) and safest (environmental impact, safety of lives). Furthermore, raising awareness of potential problems and ensuring the vessel's safe passage are crucial factors in route planning. Nowadays, determining a route relies in most cases on knowledge of past voyages or the environment of a route e.g. map data. With the implementation of Automatic Identification System (AIS) data, the maritime industry is increasingly using AIS data to make predictions and estimates for vessels. Such use include streamlining cargo transfers at harbors, helping companies make better decisions and predicting traffic flow of ships in a given region and time period [2].

However, AIS data is not always available in every possible area to be traversed. The data could be limited by a malfunction on land or on the vessel i.e. the signal could be lacking/inadequate, resulting in loss of data. This leads to gaps between data points, which can be significantly distanced leading to uncertainty regarding the vessel's actual trajectory. In addition, the computation time is often quite long, as the amount of data required to make accurate predictions is significantly large [3]. A method which is not dependent of AIS data could therefore serve as a supplement to fill in the remaining trajectory of a route. The methods used in this thesis were inspired and done by a self-study literature search for path planning methods. This was necessary in order to find a suitable approach for the subject matter at hand. The literature overview is found in the next section, followed by the problem statement of this thesis.

1.2 Literature overview

An overview of the research conducted in the path-planning field is hereby presented. The research area constituting route predicting includes multiple disciplines such as mathematics, physics and informatics. Additionally, in the past years, the application of artificial intelligence, more specifically machine learning, has become quite popular within maritime vessel tracking because of its suitability for data treatment in multiple fields within the industry [3], [4], [5]. The research conducted within the field of path planning requires input provided by: AIS data, GPS data or starting and end point. Table 1.1 summarizes the methods presented in the literature overview with a list of advantages and disadvantages of each method.

An example of route prediction is an analysis of taxi-drivers' patterns in San Francisco using GPS data [6]. A clustering technique called Symmetrized Segment-Path Distance (SSPD) is based on the shape of individual trajectories, and is used to predict the destination of taxis. The SSPD manages to obtain good clustering, capturing the drivers behaviour in a proper manner compared to other non Euclidean distance measures.

For deriving knowledge of marine traffic, an unsupervised learning scheme called Traffic Route Extraction and Anomaly Detection (TREAD) is applied in [3] to predict routes in different locations. By clustering raw AIS data into waypoints[1] for certain areas in a route, future positions are predicted. They conclude that TREAD is able to learn traffic routes robustly and detect low-likelihood behaviours.

Three different path planning methods are presented in [7]. These methods require a map as input and a start and end point. They are applied to a real map, the Oslo fjord. The first method uses a Voronoi diagram[2] to solve path planning problems. A Voronoi diagram consists of connecting lines between vertices of polygons or points describing obstacles e.g. islands. Then, perpendicular lines at their half distance are drawn and connected creating Voronoi lines. Each line not intersecting the obstacles is the suggested route. By applying a shortest path algorithm e.g. Dijkstra's [8], and a waypoint reduction function, the model is able to produce a reduced path between a starting point a to the goal b.

The next method introduced in [7] is a path planning algorithm based on Rapidly-exploring

[1] A waypoint is a point on a route used for describing the motion patterns within a given area.

[2] A Voronoi diagram divides a map into regions where each region gives the nearest distance to a location e.g. finding the store closest to your apartment.

Random Trees[3] (RRT). To increase the convergence rate, a bias toward the goal is applied [9]. As in the latter method (Voronoi diagram) conducted by [7], both Dijkstra's shortest path algorithm and a waypoint reduction function is applied on RRT. Compared with the Voronoi diagram the RRT manages to produce a shorter path in between two points in the Oslo fjord.

The final method presented in [7] applies a Probabilistic roadmap[4] (PRM) which is based on sampling random points and checking if the line between a current point and a sampled point exists, i.e. no obstacles present. This is used for path planning routes for marine vessels. The PRM produces the shortest path in the Oslo fjord compared to both previous methods conducted in [7].

With the application from fluid mechanics, [10] introduces a marine vessel planner by applying potential flow theory. Setting up potentials with different flow characteristics (sink, source, doublet and uniform) a flow pattern emerges. A numerical solution called source panel method, is used to solve the flow around circular objects placed in an artificial map. Following certain streamlines[5] leads to the desired goal. In addition, a vessel guidance scheme is derived, guiding a vessel to its destination. The simulations conducted show promising results despite their simplicity.

To summarize the findings of the literature overview, it is clear that there are many possible methods used to plan or predict routes. However, the latter approach, using potential theory, will be the approach on which this thesis is based. This thesis will aim to apply the source panel method on real map data.

[3]The RRT algorithm is used in path planning and is based on sampling points within a search space and checking if connections exist. This generates a tree-shaped graph.

[4]A Probabilistic roadmap is motion planning routine. Creating a graph of possible paths, it determines a path between a start and end point.

[5]Streamlines is the path a particle will take in a fluid flow.

Method	Author	Advantages	Disadvantages
Voronoi diagram	[7]	- Based on map i.e need polygons to describe obstacles - No need for AIS data (may be useful if wanting to straighten out route)	- Initial route may not be realistic - Must apply additional algorithm to obtain shortest route - Computationally heavy to generate
RRT	[7]	- Able to provide several routes from same or different starting points - Several trees can be initiated at once - No need for AIS data - Possible to optimize w.r.t other metrics - A bias parameter can decrease number of calculations	- Struggles in finding paths in narrow passages - Exploration method is symmetric which causes unnecessary computations
PRM	[7]	- Does not require AIS data - Based on map - Can compute multiple routes - Can implement a straightening out technique for shorter routes	- If wishing to find the shortest route, as is often the case, an additional algorithm is required - Computing multiple routes comes at a higher computational cost
Potential flow	[10]	- No need for AIS data - Not computationally heavy - Produces multiple routes by visualizing streamlines	- Must adjust strength-parameters for optimal map representation - Streamlines could be very close to objects and must therefore be selected carefully - If flow is non-uniform around a vessel, it leads to deviation from the streamline
SSPD	[6]	- Does not requires labeled data - Implementation able with cython for optimal computation speed	- Not easy to verify how much data is required for a satisfactory model
TREAD	[3]	- No prior information needed - Amount of data is immense - Model able to predict anomalies	- Model is not adequate for traffic with a low vessel density - Certain amount of pre-processing is necessary for satisfactory and precise motion patterns to be made

Table 1.1: Table summarizing and categorizing the AIS-based models (brown) and map-based models (light blue) together with advantages and disadvantages of each method.

1.3 Problem statement

Assuming shipping routes can be simulated by fluid flows, the objective of this thesis is to derive a methodology that can be used to generate a shipping route from any given points a to b. The methodology is based on principles and theory in the field of fluid mechanics, more specifically potential theory. Repulsive and attractive potentials are set out on a map in order to reach the desired goal. We will study how the model behaves in a simulated and realistic environment. Map information is required as input for the realistic case. We investigate if and how the model can be generalized e.g. creating multiple routes from different areas of the map. We will compare the flows with AIS data verifying the accuracy of the generated flows.

1.4 Goals

The goals of this thesis is to produce a flow field, consisting of streamlines, given map characteristics and construct a route in which vessels can base their travel on. The area of interest will be the Trondheimsfjord on the west coast of Norway. From the findings of [10], the potential flow solution is a good approach for a vessel's motion planner. We will take these findings one step further by testing this approach on a geographical area. We will attempt to implement both an analytical and numerical solver in this thesis. The reason for implementing both solvers is that the analytical does not take into account the map characteristics, but is rather simple to produce, computationally light and may assist in creating a complex fluid flow. Map characteristics are crucial when wanting to find the fluid velocities surrounding arbitrary geometries. By using boundary conditions and setting constraints in specified areas of the landscape, it is possible to give the map as input to the model and thus computing the flow field in the area of interest. This is what the numerical approach will allow us to achieve.

1.5 Constraints

We will assume land is the only obstacle for each vessel. In reality there could be several vessels traversing the same area at the same time which could lead to a change in course.

1.6 Outline of thesis

The thesis is further structured as follows: Part I includes the theory, more specifically the equations which emerge from the Navier-Stokes equations assuming irrotational and incompressible fluids. From these assumptions, we derive the analytical solutions to the Laplace equation which leads to fundamental flow characteristics and produce flow fields around circular objects. Increasing the complexity of the object geometry leads to a deduction of the source panel method. Part II includes the implementation of methods. A class is created for the various analytical solutions of the Laplace equation. Furthermore, we show how to implement the source panel method on a map using the Python package Basemap [11] to create maps. A route extraction scheme is introduced for creating routes given AIS data. The results are presented and discussed in part III. With simple geometries we examine how the analytical solutions are able produce flows around circular objects. Also, we conduct a verification of the generated source panel method, and with the use of AIS data, we compare the produced flow fields and study the performance of the source panel method in Trondheimsfjorden. At last, we conclude the findings conducted in this thesis and discuss future work.

Part I

Theory

Chapter 2

Vectors and vectors fields

In the following chapters the mathematics used in the thesis is presented. The theory will mostly be from the field of fluid mechanics. A brief introduction to the equations governing the properties and motion of fluids will be given, namely Navier-Stokes equations. From the general Navier-Stokes equations, assumptions and simplifications about a fluid characteristics will be made, more specifically, irrotational and incompressible flow. This will yield a set of equations which will be the used to describe the flow of fluids. These equations are part of what constitute potential theory in fluid mechanics.

In addition to an analytical flow solver, we introduce a numerical method (Source Panel Method) for solving the flow around objects with complex geometry. By approximating an arbitrary object with a set of panels and setting out a source type flow on each panel, we solve the unknown source strength coefficient using boundary conditions leading to a system of equations which is solved. A more detailed description can be found in [12], [13] and [14].

2.1 Vector

A vector is used to describe a quantity with direction and magnitude. Vectors are useful in navigation for representing quantities such as positions and velocities. From a point of reference this can be represented in a coordinate system, mathematically written as:

$$\vec{r} = [x, y, z]. \tag{2.1}$$

The arrow indicates that r is of the type vector. The first component moves along the x-axis, second along the y-axis and the third along the z-axis. The three components indicate that the vector $\vec{r}$ exists in a three-dimensional space. This is visualized in fig. (2.1) below:

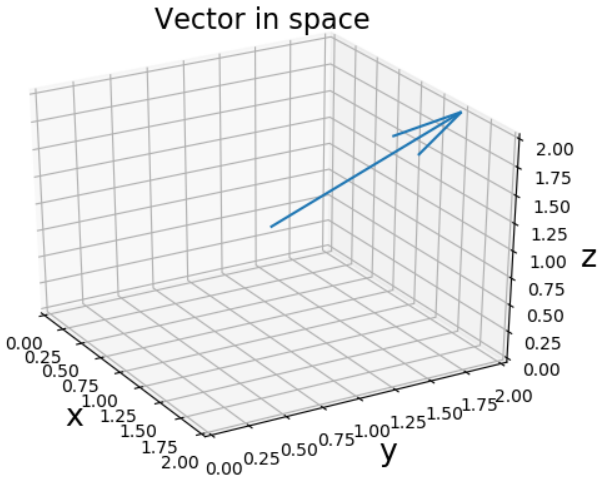

Figure 2.1: Above shows a typical vector representation $\vec{r}$ in 3d-space where the tail of the arrow represents its spacial coordinate. The arrow shows the vectors direction, its length represents its magnitude.

2.2 Vector field

As well as vectors, vector fields are useful when wanting to describe velocities, forces or accelerations within an area. A vector field $\vec{F}$ consists of a set of vectors assigned points within a space, see fig. (2.2). Vector fields are commonly used when visualizing fluid motion and vector quantities can vary. Here they will represent velocities. The field is represented with vectors which are functions of spacial coordinates and time, denoted as:

$$\vec{F} = \vec{F}(x, y, z, t). \tag{2.2}$$

This can further be written as:

$$\vec{F}(x, y, z, t) = [F_x(x, y, z, t), F_y(x, y, z, t), F_z(x, y, z, t)], \tag{2.3}$$

where F_x, F_y, F_z are scalar fields. The field $\vec{F}$ is called stationary if it is independent of time. See fig. (2.2) below for a stationary vector field.

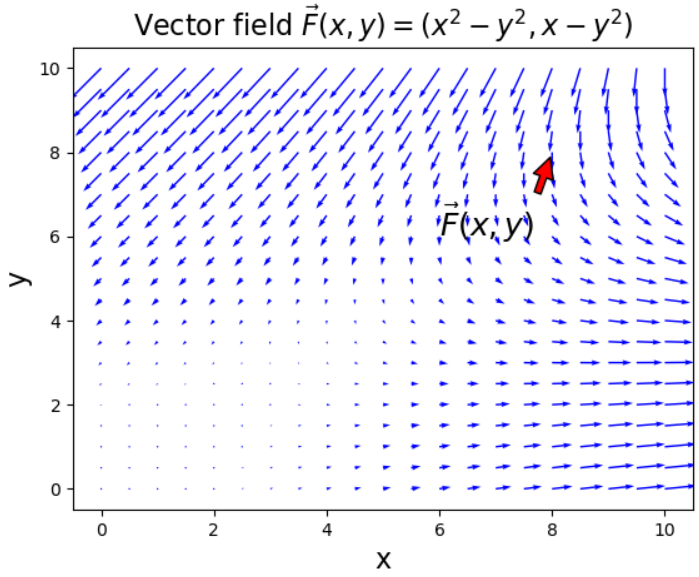

Figure 2.2: A vector field $\vec{F}$ is visualized taking in two parameters x and y and calculates the field in a grid defined from [0-10] in both x and y-direction. The arrow length is the magnitude at that specific point, also indicating the field's direction. A grid cell are the points where the field $\vec{F}$ is evaluated. The red arrow point to a vector in the field located at $(8, 8)$

Chapter 3

Navier-Stokes equations

3.1 Navier-Stokes equations

The Navier-Stokes equations [15] are the equations describing the motion of a fluid. The equations are derived using the conservation laws of physics and the quantities include mass, momentum and energy. A fluid is assumed to obey these laws [16]. These conservation laws can be expressed in terms of mathematical statements. Here, we will only be concerned about the conservation laws regarding mass and momentum. The continuity equation emerge from the conservation law of mass which states that mass can not be created or disappear. The general equation of continuity is stated below:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}) = 0, \tag{3.1}$$

where ρ is the fluid density, $\nabla = [\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}]$ the gradient and $\vec{u}$ the velocity vector. We will assume incompressible fluids i.e. the density ρ is constant. The continuity equation for incompressible fluid states:

$$\nabla \cdot \vec{u} = 0. \tag{3.2}$$

Both density terms are excluded as $\frac{\partial \rho}{\partial t} = 0$, and dividing by the density ρ.
This leads up to the next equation, the momentum equation for incompressible fluids. It states that a fluid's density times acceleration is proportional to the forces acting upon the fluid,

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla)\vec{u} = -\frac{\nabla p}{\rho} + \nu \nabla^2 \vec{u} + f_v. \tag{3.3}$$

The left hand side states the inertia term where $\vec{u}$ is the fluid velocity, $\frac{\partial \vec{u}}{\partial t}$ and the $(\vec{u} \cdot \nabla)\vec{u}$ are the local and convective acceleration, respectively. The right hand side gives pressure

gradient ∇p, the fluid density ρ, the kinematic viscosity ν. Viscous forces due to the stickiness of a fluid is given by the term $\nu\nabla^2\vec{u}$ and external forces f_v. The "shipping" fluid is stationary $\frac{\partial\vec{u}}{\partial t} = 0$ and inviscous $\nu = 0$. These assumptions lead further to potential flow theory which will be covered next.

Chapter 4

Irrotational and incompressible flow

The Navier-Stokes equations are nonlinear, integral equations which cannot be solved analytically. Two approaches are therefore common to apply in order to solve the Navier-Stokes equations. The first approach, is to simplify the problem at hand to the extent that the equations become linear and can be solved analytically. The second is using numerical methods to solve the equations, allowing for a more realistic simulations and solutions. The nonlinear integral terms are replaced by a discretized approach yielding solutions at specified points (or grid). This is more commonly known as Computational Fluid Dynamics (CFD) [16]. We will cover these methods in the following chapters.

4.1 Stream function

Given a vector field $\vec{F} = \vec{F}(x, y, z, t)$, it is possible to visualize the field $\vec{F}$ by field lines. Field lines can be described as tracing a line following the direction of a vector field $\vec{F}$. By looking a specific point P in time, field lines have a vector in the field as tangent. Suppose $d\vec{r} = [dx, dy, dz]$ is a small arc element on a field line at point P. The cross product between the vector $\vec{F}$ and $d\vec{r}$ is then zero. If the vector $\vec{F}$ represents the velocity vector $\vec{u} = [u(x, y, z, t), v(x, y, z, t), w(x, y, z, t)]$ of a fluid, the cross product between the velocity $\vec{u}$ and $d\vec{r}$ can be written as:

$$\vec{u} \times d\vec{r} = 0. \tag{4.1}$$

Writing out the components from the cross product gives:

$$(vdz - wdy)\vec{i} = 0,$$

$$(wdx - udz)\vec{j} = 0,$$

$$(udy - vdx)\vec{k} = 0.$$

This gives the equations:

$$\frac{dx}{u} = \frac{dy}{v} = \frac{dz}{w}, \tag{4.2}$$

where $\vec{i}, \vec{j}$ and $\vec{k}$ are unit vectors in the x, y and z- direction, respectively and assuming $u, v, w \neq 0$. Now, $\vec{F}$ represents a velocity $\vec{u}$, the corresponding field lines are then called streamlines. As mentioned above, assuming a fluid's flow is stationary, the trajectory and streamlines are the same. Assuming a 2-dimensional space, streamlines give rise to differential equations with the velocity components u and v known. Rearranging eq. (4.2) and with u and v known the trajectory, thus streamlines can be found by integrating:

$$\frac{dy}{dx} = \frac{v}{u}. \tag{4.3}$$

This differential equation yields solutions on the form $y(x, C)$, where y represents streamlines with a given integration constant C. Next, we see how we can relate the velocities to a stream function ψ.

For a 2-dimensional description the equation for an incompressible flow yields:

$$\nabla \cdot \vec{u} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0, \tag{4.4}$$

with the velocity vector $\vec{u} = [u, v]$ and $\nabla = [\frac{\partial}{\partial x}, \frac{\partial}{\partial y}]$ the differential operator. We can relate the velocity vector to the stream function by defining the stream function as:

$$u = \frac{\partial \psi}{\partial y}, \tag{4.5}$$

$$v = -\frac{\partial \psi}{\partial x}, \tag{4.6}$$

where ψ is the stream function. By putting eqs. (4.5) and 4.6) into eq. (4.4) we find:

$$\nabla \cdot \vec{u} = \frac{\partial^2 \psi}{\partial x \partial y} - \frac{\partial^2 \psi}{\partial x \partial y} = 0, \tag{4.7}$$

satisfying the continuity equation. By assuming an irrotational velocity field, $\nabla \times \vec{u} = 0$, the Laplace equation is obtained (see next section for more details):

$$\nabla \times \vec{u} = \frac{\partial^2 \psi}{\partial y^2} + \frac{\partial^2 \psi}{\partial x^2} = \nabla^2 \psi = 0, \tag{4.8}$$

where eq. (4.5) and eq. (4.6) are substituted in as expressions for the velocity components.

Furthermore, eq. (4.3) gives us the conditions on what the stream function must look like in order for this to be true. Putting eq. (4.5) and eq. (4.6) into eq. (4.3) the stream function yields:

$$udy - vdx = \frac{\partial \psi}{\partial y}dy + \frac{\partial \psi}{\partial x}dx - = 0, \tag{4.9}$$

$$\partial \psi = 0, \tag{4.10}$$

$$\psi = Constant. \tag{4.11}$$

The stream function ψ is constant along a streamline. All stream functions ψ satisfying eq. (4.7) are valid choices for ψ. Different stream functions ψ can solve the Laplace equation yielding various flow characteristics. Sinks, sources and doublets are fundamental solutions. These will be derived in the next chapter. The velocity components v_r and v_θ given a stream function can be written in polar coordinates as:

$$v_r = \frac{1}{r}\frac{\partial \psi}{\partial \theta}, \tag{4.12}$$

$$v_\theta = -\frac{\partial \psi}{\partial r}, \tag{4.13}$$

and the Laplacian in polar coordinates:

$$\nabla^2 \psi = \frac{\partial^2 \psi}{\partial r^2} + \frac{1}{r}\frac{\partial \psi}{\partial r} + \frac{1}{r^2}\frac{\partial^2 \psi}{\partial \theta^2} = 0. \tag{4.14}$$

The radius r is related to a Cartesian coordinate system as $r = \sqrt{x^2 + y^2}$. The angle $\theta = \arctan(\frac{y}{x})$ is the angle from origo to r.

4.2 Velocity potential

Another way of relating a scalar function to the Laplace equation is the velocity potential. In the general case, the velocity potential is a scalar function denoted by $\Phi(x, y, z, t)$. In the following, we will consider a 2-dimensional scalar function $\Phi(x, y)$ in a steady state. We will use the same equation as for the stream function, but with a slightly different approach. Instead of starting with the continuity eq. (4.4), we assume an irrotational flow $\nabla \times \vec{u} = 0$ leading to the existence of a scalar function $\Phi(x, y)$ satisfying the irrotational

property. As for the stream functions the velocity is said to be derived from a velocity potential and can be written as:

$$\vec{u} = \nabla\Phi, \tag{4.15}$$

where $\vec{u}$ is the fluid velocity, $\nabla = [\frac{\partial}{\partial x}, \frac{\partial}{\partial y}]$ is the differential operator and $\Phi(x, y)$ a scalar function. Inserting the above equation into eq. (4.4) we find:

$$\nabla \cdot \vec{u} = \nabla \cdot (\nabla\Phi) = \nabla^2\Phi = 0. \tag{4.16}$$

We have now related a scalar function Φ to the Laplace equation (4.16) as we did with the stream function ψ. The velocity components for the vector $\vec{u} = [u, v]$ are then given the velocity potential Φ:

$$u = \frac{\partial\Phi}{\partial x}, \tag{4.17}$$

$$v = \frac{\partial\Phi}{\partial y}. \tag{4.18}$$

The velocities in an incompressible and irrotational flow can be expressed with a stream function ψ or velocity potential Φ as:

$$u = \frac{\partial\Phi}{\partial x}, \quad u = \frac{\partial\psi}{\partial y}, \tag{4.19}$$

$$v = \frac{\partial\Phi}{\partial y}, \quad v = -\frac{\partial\psi}{\partial x}. \tag{4.20}$$

This relation is known as the Cauchy Riemann relations. The equations derived in this section are the fundamental equations needed to describe fluid flows. The velocity potential and stream function theory are what is known as potential theory in fluid mechanics. Next, we look at the Laplace equation and derive some fundamental flow characteristics.

Chapter 5

Fundamental flows in two dimensions

5.1 Solutions of the Laplace equation

Here, analytical solutions of the Laplace equation are derived. The Laplace equation has been applied in motion planning and obstacle avoidance by building an artificial potential field creating flow characteristics based on the different solutions to the Laplace equation, [10], [17]. We will draw on this application of the Laplace equation to create flow characteristics for vessel trajectories. Common flow characteristics and solutions to the Laplace equation are: source, sink and doublet flow, each flow with its unique flow characteristic. The flows will be used to represent the following: A source flow representing a start point, sink flow representing an end point and a doublet flow representing circular objects. In this thesis will also use the a source type flow to represent circular objects. These flow characteristics can be combined by applying the superposition principle, thus simulating basic, but numerous flow patterns. The superposition states that if the velocity potential ϕ_1 is a solution of the Laplace equation eq. (4.16) and ϕ_2 also a solution, then the sum of ϕ_1 and ϕ_2 is also a solution.

5.1.1 Uniform flow

A uniform flow has constant velocity components u and v in the x and y-direction, respectively. By the relation from eq. (4.5) and eq. (4.6) this satisfies eq. (4.8) and is therefore a solution of the Laplace equation.

5.1.2 Sink and source flow

Consider a source flowing radially outwards from a point P at a steady rate Q. Finding the amount flowing outwards, the velocity components of the source flow can be found by integrating the flow over a circle with radius r. This can be expressed as:

$$Q = \int_0^{2\pi} v_r r d\theta = 2\pi r v_r, \tag{5.1}$$

where Q is the flow strength. Substituting eq. (4.12) into eq. (5.2) for the radial velocity and integrating. This type of flow has the characteristics of a source and has the stream function:

$$\psi_{source} = \frac{Q}{2\pi}\theta. \tag{5.2}$$

This is also a solution of the Laplace equation (4.8). The corresponding velocity components can be written as:

$$u_{source} = \frac{Qx}{2\pi(x^2+y^2)}, \tag{5.3}$$

$$v_{source} = \frac{Qy}{2\pi(x^2+y^2)}, \tag{5.4}$$

where eq. (4.12) is equated with eq. (5.1) for the radial velocity and substituting $r = \sqrt{x^2+y^2}$. A flow which absorbs fluid isotropically is called a sink and has the same stream function as a source but with a negative sign:

$$\psi_{sink} = -\frac{Q}{2\pi}\theta, \tag{5.5}$$

sharing the same terms as the source but with opposite sign.

$$u_{sink} = -\frac{Qx}{2\pi(x^2+y^2)}, \tag{5.6}$$

$$v_{sink} = -\frac{Qy}{2\pi(x^2+y^2)}. \tag{5.7}$$

5.1.3 Doublet flow

A sink and source flow coinciding at the same point is called a doublet flow. By placing a sink in a point, $(\epsilon, 0)$ with angle θ_2 and a source in $(-\epsilon, 0)$ with angle θ_1. ϵ is considered

to be a small distance. By using superposition principle, adding the two stream functions ψ_{sink} and ψ_{source} gives another stream function:

$$\psi = -\frac{Q}{2\pi}(\theta_1 - \theta_2), \tag{5.8}$$

taking the tangent on each side gives:

$$\tan\Big(-\frac{2\pi\psi}{Q}\Big) = \tan(\theta_1 - \theta_2) = \frac{\tan(\theta_1) - \tan(\theta_2)}{1 + \tan(\theta_1)\tan(\theta_2)}. \tag{5.9}$$

Further we rewrite $tan(\theta_1)$ as $\frac{y}{x-\epsilon}$ together with $tan(\theta_1)$ as $\frac{y}{x+\epsilon}$ and insert this into eq. (5.9).

$$\tan\Big(-\frac{2\pi\psi}{Q}\Big) = \frac{\frac{y}{x-\epsilon} - \frac{y}{x+\epsilon}}{1 + \frac{y^2}{(x^2-\epsilon^2)}}. \tag{5.10}$$

After cleaning up the equation reads:

$$\tan\Big(-\frac{2\pi\psi}{Q}\Big) = \frac{2y\epsilon}{x^2 + y^2 - \epsilon^2}. \tag{5.11}$$

Taking the inverse tangent on both sides of and remembering that ϵ is small a distance gives small values in the argument of $arctan$. This gives:

$$-\frac{2\pi\psi}{Q} = \arctan\Big(\frac{2y\epsilon}{x^2 + y^2 - \epsilon^2}\Big) \approx \frac{2y\epsilon}{x^2 + y^2 - \epsilon^2}, \tag{5.12}$$

and solving for the stream function yields:

$$\psi = -\frac{Q\epsilon y}{\pi(x^2 + y^2 - \epsilon^2)}. \tag{5.13}$$

Let $\epsilon \to 0$ and $Q \to \infty$, leading to their product being constant. This can be written as:

$$\lim_{\substack{\epsilon\to 0 \\ Q\to\infty}} -\frac{Q\epsilon y}{\pi(x^2 + y^2 - \epsilon^2)} = -\frac{ky}{\pi(x^2 + y^2)}, \tag{5.14}$$

where k is the strength of the doublet. The stream function equation for a doublet type flow reads:

$$\psi_{doublet} = -\frac{ky}{\pi(x^2 + y^2)}. \tag{5.15}$$

The velocity components u and v for a doublet are found from eq. (4.5) and eq. (4.6), respectively:

$$u = \frac{k(y^2 - x^2)}{(x^2 + y^2)^2}, \tag{5.16}$$

$$v = -\frac{kyx}{(x^2 + y^2)^2}. \tag{5.17}$$

Figs. (5.1 - 5.3) visualize fundamental flows derived from the Laplace equation. Fig. (5.4) shows how the Laplace equation is able to create more complex flow with the superposition principle.

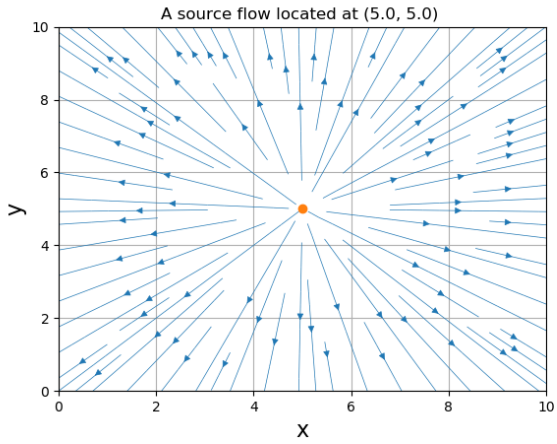

Figure 5.1: This flow characteristic is called a source flow. The velocity components points outward from its origin (orange point). This flow characteristic emits fluid making it a representation for initial positions and circular objects.

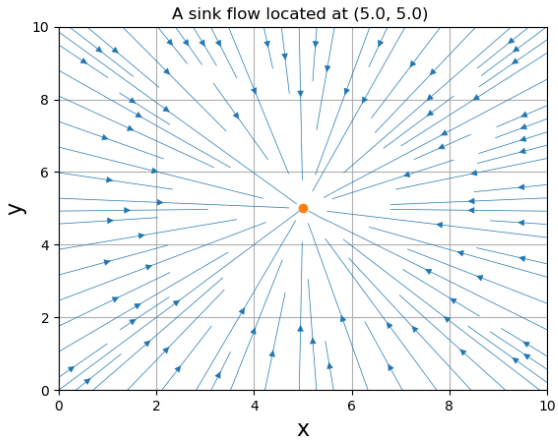

Figure 5.2: A solution of the Laplace equation produces streamlines directed radially inward to its origin (orange point). The velocity decreases inversely proportional from its distance from the origin. This flow characteristic is called a sink flow. This flow is useful when representing end point of routes as the sink absorbs the fluid.

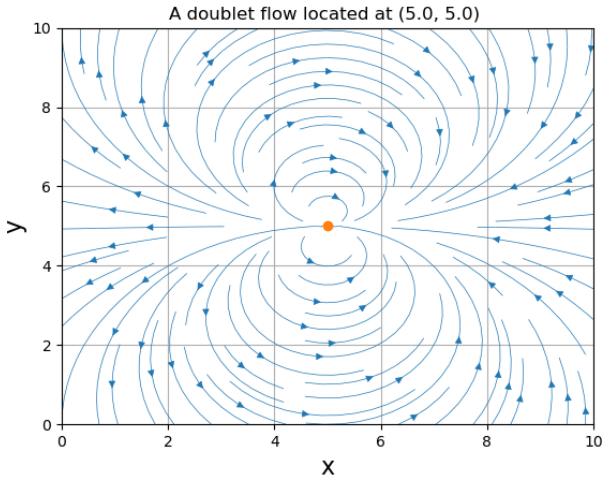

Figure 5.3: A doublet emerges from a sink and source coinciding in the point. A doublet represents circular objects for motion planning. From the figure we see the streamlines flow around the defined point of origin (orange).

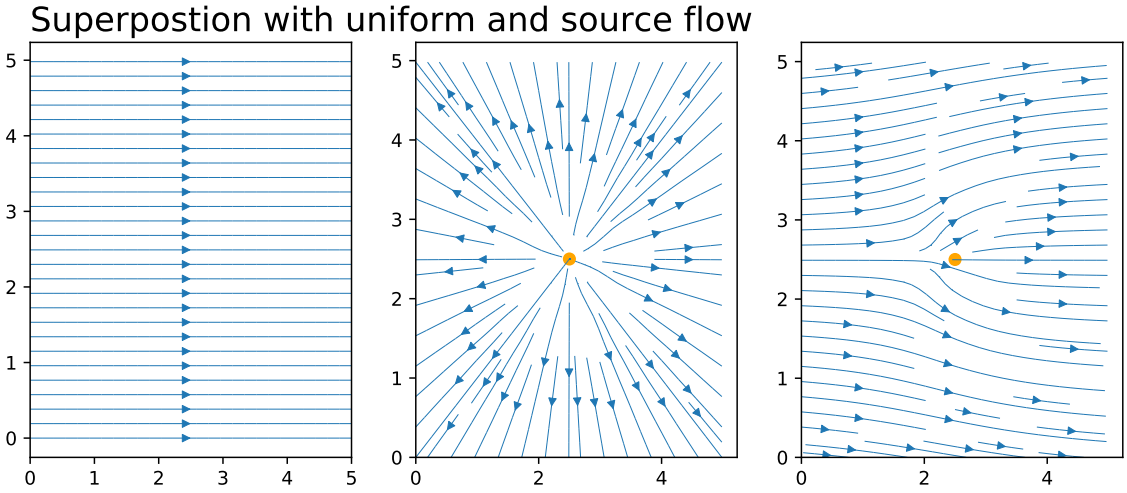

Figure 5.4: A uniform flow with velocity $\vec{u} = (const, 0)$ is shown in $left$. A source flow located at $(2.5, 2.5)$ is shown in the $middle$. A combination of both uniform and source flow using the superposition principle is shown in the $right$. Note how the flow bends around the source flow origin $(2.5, 2.5)$. This will be used to represent objects for motion planning.

Chapter 6

Source Panel Method (SPM)

When the situation gets more complex and the geometry can no longer be described by fundamental flows, numerical methods need to be applied to solve for velocities in such areas. The source panel method (SPM) approximates arbitrary geometric shapes with a set of panels. By using boundary conditions at the surface and placing a small source potential at the center of each panel, a system of equations is set up and solved. The idea is to distribute sources around a body in order to deflects incoming flow, creating a flow around or along a body. The following section explains in detail how the source panel method is set up with boundary conditions and geometry. How to calculate the source strengths and velocity flow at every grid cell in the defined area leads to a flow characteristic.

6.1 Boundary conditions

From the previous section we have different solutions leading to various flow characteristics such as uniform, source, sink and doublet flow. It was also found that the superposition principle could be applied to the Laplace equation because of its linearity property. The superposition principle allows for more realistic flow types than elementary solutions of the Laplace equation. One can simulate the effect from objects placed in the flow-field by restricting the flow at certain points in the region i.e. using boundary conditions, see fig. (6.1). Boundary conditions replaces islands and coastlines. This will provide us with an estimate of a shipping route.

Islands and coastlines are impenetrable. Mathematically, the normal velocity must be zero at the surface. In addition, for inviscid fluids the tangential velocity component at the surface of the body is finite. The equations which arise from these boundary conditions

can be written mathematically as:

$$\vec{V} \cdot \vec{n} = 0, \tag{6.1}$$

where $\vec{V}$ is the flow velocity and $\vec{n}$ is the normal vector pointing out of the body. The tangential boundary condition is denoted:

$$\vec{V} \cdot \vec{t} = c, \tag{6.2}$$

where $\vec{V}$ is the flow velocity, $\vec{t}$ is the tangential vector and c constant. For flow very far away from the body we assume the velocity to be uniform:

$$u = V_\infty, \tag{6.3}$$

where u is the velocity component in x-direction and V_∞ is a constant value resulting in a uniform flow in the x-direction, whereas the y component of the velocity vector is zero:

$$v = 0. \tag{6.4}$$

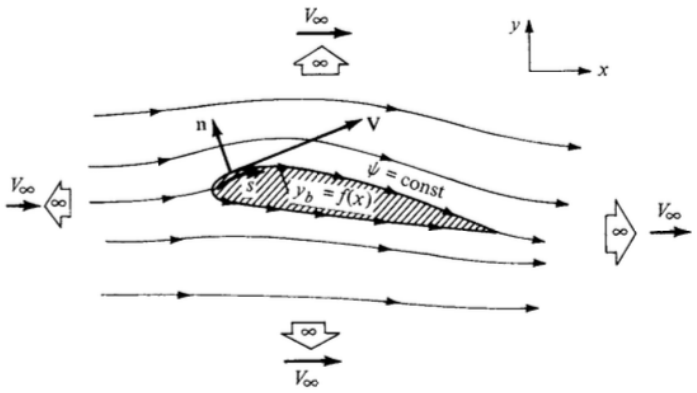

Figure 6.1: The figure shows flow around an airfoil. The uniform flow is the same as the equation above i.e. only contribution in the x-direction. Image taken from [14].

6.2 Approximating complex geometries with panels

When solving for the flow around a solid body e.g. the flow around a circular cylinder, the body's geometric shape is given. What about shapes that are not circular with arbitrary shape? The same principles of finding the velocity and strengths also apply to those objects, but it would be complicated to guess what combinations of flows constituting that

specific shape. It would be beneficial to need only one kind of shape. The source panel method (SPM) is based on dividing a solid body into sets of panels. Each panel is assigned a source type flow. These panel source strengths make up a source sheet and can be represented by a function $\lambda(s)$ where s is the distance along the sheet and λ its corresponding strength value. For a point P located a distance r away from the small piece of the panel segment ds, an infinitesimal small potential $d\phi$ occurs at P. This can be written as:

$$d\phi = \frac{\lambda(s)ds}{2\pi} \ln r, \tag{6.5}$$

where $r = \sqrt{x^2 + y^2}$ is the distance from ds inducing a $d\phi$ at point P and λ is the source strength. The full source sheet induced at P is found by integrating over the source sheet from a to b. The velocity potential ϕ can further be written as:

$$\phi = \int_a^b \frac{\lambda(s)ds}{2\pi} \ln r. \tag{6.6}$$

It was stated that the source strength from a panel was $\lambda(s)$ which gave the equation stated above. Now, assuming that the source strength over a single panel is constant, we can allocate each panel with an unknown source strength coefficient $\lambda_1, \lambda_2, ..., \lambda_n$ yielding the possibility to approximate the source strengths at each panel. Each panel may have different source strengths λ and area located at the center of each panel (also called control point). For n panels, n source strengths occur.

The objective is to solve each individual panel strength λ_i, $i = 1, 2, 3, ..., n$. The boundary condition for the normal component states that the velocity is zero in that direction. Furthermore, the boundary condition at each *panel point* allows for a set of equations to be expressed. Panel j giving rise to a potential $\phi(P)$ from eq. (6.5) can be written as:

$$\Delta\phi_j = \frac{\lambda_j}{2\pi} \int_j \ln r_{pj} ds_j, \tag{6.7}$$

where the distance from point P to panel j is $r_{pj} = \sqrt{(x_p - x_j)^2 + (y_p - y_j)^2}$. Point $P = (x_p, y_p)$. λ_j is constant over the panel j and moved outside the integral. The total potential $\phi(P)$ induced at point P is found by summing over each panel:

$$\phi(P) = \sum_{j=1}^{n} \Delta\phi_j = \sum_{j=1}^{n} \frac{\lambda_j}{2\pi} \int_j \ln r_{pj} ds_j, \tag{6.8}$$

giving the expression for the potential $\phi(P)$ at a point P.
Now, move the point P to a panel i and define the point (x_i, y_i) to be the control point of

panel i we can express the influence of every panel to the ith panel as:

$$\phi(x_i, y_i) = \sum_{j=1}^{n} \frac{\lambda_j}{2\pi} \int_j \ln r_{ij} ds_j, \tag{6.9}$$

with the distance $r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$. Next, we derive the expression for the normal velocities at each panel using eq. (6.9) and the freestream velocity vector.

Normal velocity calculation

The normal component boundary condition is applied at the control point (x_c, y_c) which is located at the center of each panel (see fig. 6.3). The freestream velocity is needed to generate a flow around the object is needed to evaluate the correct orientation of the vectors. Let, $\vec{n_i}$ be the normal vector located the control point of panel i. The freestream vector V_∞ can be generated at an angle α relative to the x-axis. The angle α is commonly known as the angle of attack. The vector normal to the freestream velocity vector at the control point at the ith panel can be expressed as:

$$V_{\infty,n} = \vec{V_\infty} \cdot \vec{n_i} = V_\infty \cos(\beta_i), \tag{6.10}$$

where β_i is the angle between $\vec{V_\infty}$ and $\vec{n_i}$. The normal velocity component V_n at panel i produced by every panel can be found by:

$$V_n = \frac{\partial}{\partial n_i}[\phi(x_i, y_i)]. \tag{6.11}$$

The derivative $\frac{\partial}{\partial n_i}$ acts on the distance variable r_{ij}. Note, when $j = i$ the calculation for the same panel is done on itself and the distance variable becomes zero. The term is now located in the denominator leading to a singularity. When $j = i$ the contribution has been shown to give $\frac{\lambda_i}{2}$, see Appendix A. This gives an update of the potential equation being expressed as:

$$V_n = \frac{\lambda_i}{2} + \sum_{j=1,(j\neq i)}^{n} \frac{\lambda_j}{2\pi} \int_j \frac{\partial}{\partial n_i} \ln r_{ij} ds_j. \tag{6.12}$$

This the normal vector component of the source panel contribution at panel i. The boundary states:

$$V_{\infty,n} + V_n = 0, \tag{6.13}$$

as the flow cannot penetrate the walls of a solid body, the latter equation can be expressed as:

$$V_\infty \cos(\beta_i) + \frac{\lambda_i}{2} + \sum_{j=1,(j\neq i)}^{n} \frac{\lambda_j}{2\pi} \int_j \frac{\partial}{\partial n_i} \ln r_{ij} ds_j = 0, \tag{6.14}$$

substituting eqs. (6.10) and (6.12) into eq. (6.13). We get an equation with n unknowns. The task now is to gather n equations to solve for the source strengths. Solving the source strength coefficients gives rise to the streamlines over the set of panels together with the freestream velocity. This equation can be expressed in matrix form solving the system of equation using numerical methods. This approach makes it possible to control the amount of accuracy needed for the model as the number of panels is used as input to the model. Next, we define the geometry used for SPM and tackle the integral term in eq. 6.14.

Thorough deduction of source panel method

The previous section provided an overview of the equations for calculating the flow around an arbitrary body. In this section, a more detailed derivation will be done. The integral terms will be calculated together with the geometry of the problem. This will be set up into a matrix form $Ax = b$ which can be solved thus leading to a solution.

6.3 Defining geometry

Each panel can be of different length and therefore each panel needs to be calculated separately. The starting point needs to be defined and thereafter calculated each panels property in an orderly fashion. This is necessary because the normal vector orientation will be affected by this choice. The suggested method for calculating panels are clockwise. Below, fig. (6.2) shows a circle approximated with panels given the boundary points.

$$dx = (x_2 - x_1), \tag{6.15}$$

$$dy = (y_2 - y_1), \tag{6.16}$$

$$s_1 = \sqrt{dx^2 + dy^2}, \tag{6.17}$$

and for the ith panels have the notation:

$$dx_i = x_{i+1} - x_i, \tag{6.18}$$

$$dy_i = y_{i+1} - y_i, \tag{6.19}$$

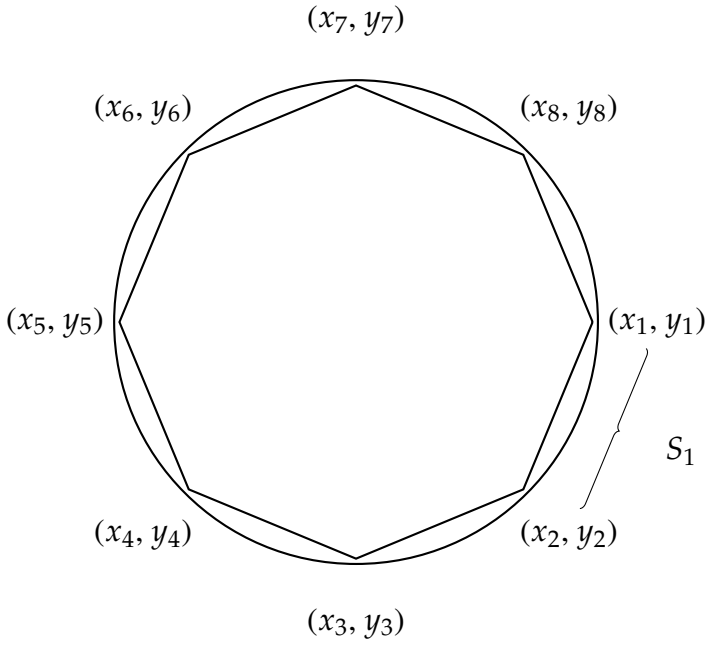

Figure 6.2: Approximating a circle with panels. n panels are created with boundary points representing the end points of each panel (x_b, y_b). From the figure, 8 panels s are created with 9 boundary points ((x_1, y_1) are counted twice for the first and last boundary point).

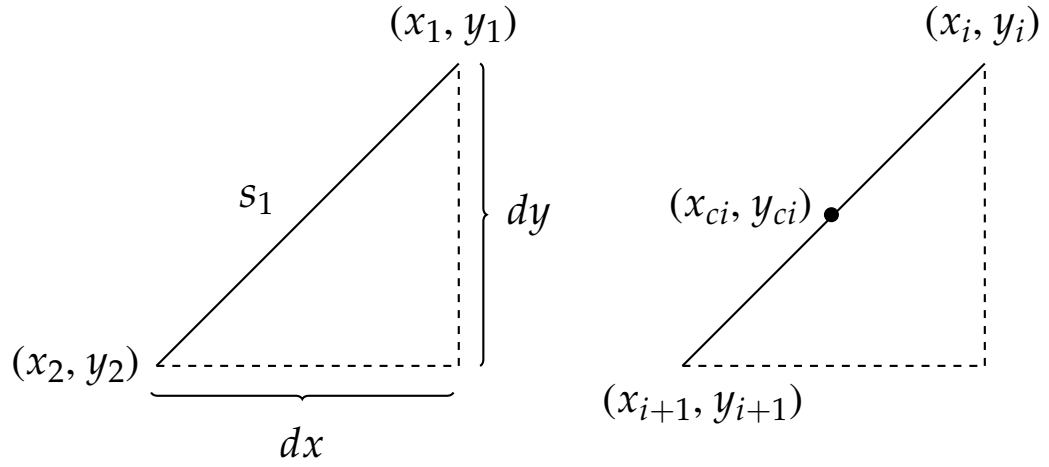

Figure 6.3: A closer view of panel S_1 from fig. (6.2) (left). The equations needed for the panel calculation can be seen above (right). The general case for the ith panel with corresponding boundary points and control points. With the boundary points defined, the control points can be calculated by eqs. (6.21) and (6.22).

$$s_i = \sqrt{dx_i^2 + dy_i^2}. \tag{6.20}$$

$i = 1, 2, 3..., n$ goes up to the number of panels n. Furthermore, from fig. (6.3) the control points for each panel is needed. This is where the source strength is set up. This point has the relation:

$$x_{ci} = \frac{x_i + x_{i+1}}{2}, \tag{6.21}$$

$$y_{ci} = \frac{y_i + y_{i+1}}{2}. \tag{6.22}$$

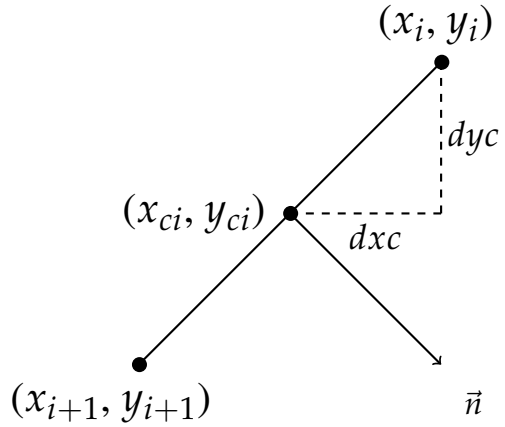

Figure 6.4: In this figure a normal vector $\vec{n}$ is added at the control point together with the change in x and y direction from the control point (dxc and dyc) which is necessary when calculating angles between panels and normal vectors.

The angle ϕ between the panel and x-axis can be found by:

$$\tan\phi_i = \frac{dy}{dx}. \tag{6.23}$$

Note, the angle ϕ_i does not change for a different placement of the x-axis. The angles δ_i and β_i are the angles from the x-axis to the normal vector, and from the freestream vector to the normal vector, respectively. They are expressed as:

$$\delta_i = \phi_i + \frac{\pi}{2}, \tag{6.24}$$

$$\beta_i = \delta_i - \alpha, \tag{6.25}$$

where α is the angle of attack from the freestream vectors. The angles are visualized in fig. (6.5). The normal vector components can be expressed as:

$$n_{xi} = x_{ci} + s_i \cos\delta_i, \tag{6.26}$$

$$n_{yi} = y_{ci} + s_i \sin\delta_i, \tag{6.27}$$

$$\vec{n} = n_{xi}\vec{i} + n_{yi}\vec{j}. \tag{6.28}$$

$\vec{i}$ and $\vec{j}$ are unit vectors in x and y - direction, respectively.

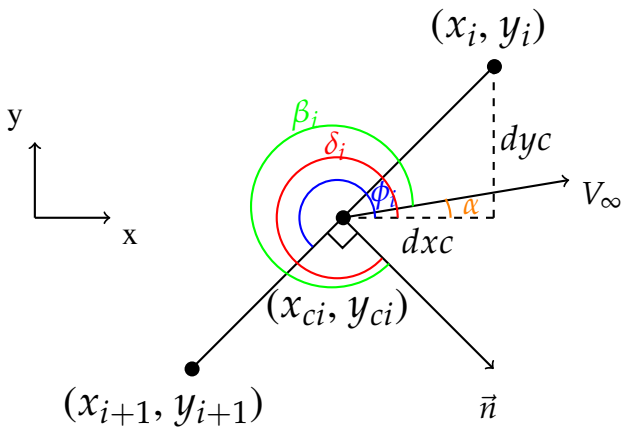

Figure 6.5: This figure shows the angles needed to compute the normal vectors of each panel derived from the boundary and control points. Going in clockwise direction with boundary points defined at (x_i, y_i) and (x_{i+1}, y_{i+1}). The control point (x_{ci}, y_{ci}) on the ith panel is located at the panels center.

Building complex flows

Suppose now that we have a curve with infinite sources located on its arc. The expression for calculating the velocity potential at a point is given by the integral:

$$\phi_p = \int_a^b \frac{\lambda(s)}{2\pi} \ln r_p(s) ds. \tag{6.29}$$

The integral assumes the source strength over the body's periphery (see fig. 6.6) depends on the distance s along the surface S. By breaking the surface up into straight lines the integral above can be manipulated into the equation:

$$\phi_p = \int_a^b \frac{\lambda(s_{ab})}{2\pi} \ln r_p(s_{ab}) ds_{ab} + \int_b^c \frac{\lambda(s_{bc})}{2\pi} \ln r_p(s_{bc}) ds_{bc}. \tag{6.30}$$

Fig. (6.7) depicts this integral case. The curve now consists of two lines where each line contains its own source strength $\lambda(s_{ab})$ and $\lambda(s_{bc})$. The number of panels can be adjusted. For n panels the integral reads:

$$\phi_p = \sum_{j=1}^{n} \int_j \frac{\lambda(s_j)}{2\pi} \ln r_{pj}(s_j) ds_j. \tag{6.31}$$

Now assume the source strength $\lambda(s)$ is constant along each panel. This leads to the source strength at panel j being denoted as $\lambda(s_j) = \lambda_j$ and the distance $r_{pj}(s_j) = r_{pj}$. We then write the integral equation above as:

$$\phi_p = \sum_{j=1}^{n} \lambda_j \int_j \frac{\ln r_{pj}}{2\pi} ds_j. \tag{6.32}$$

Figs. (6.6 - 6.8) shows a step by step visualization of the approximations done above.

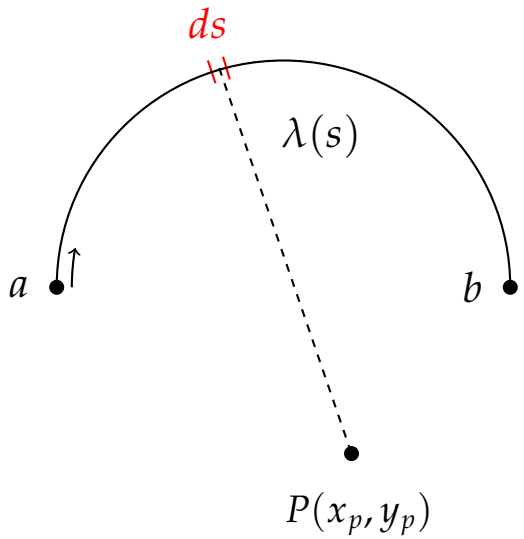

Figure 6.6: A part of a solid body (black line) with a source strength $\lambda(s)$ is shown here. The body induces a small potential $d\phi$ from ds at point P. Integrating from a to b grants the total contribution of the potential ϕ along the periphery of the surface. The arrow indicates the integration direction.

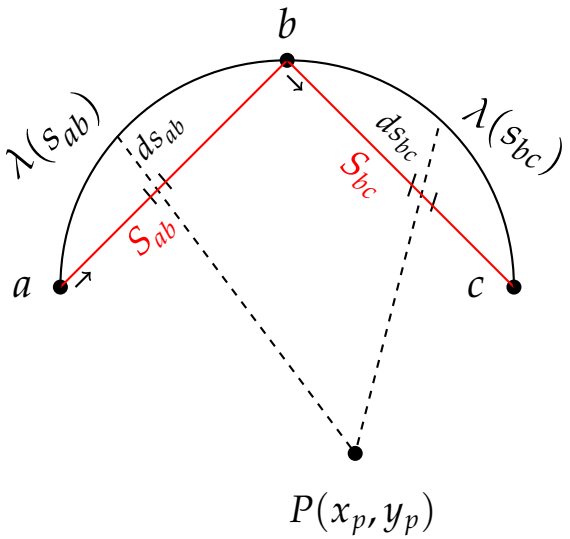

Figure 6.7: Here, we show the next step of approximating the geometry a body. By splitting the body into panels (red) gives two source strengths from $a-b$ and $b-c$ on the body. S_{ab} and S_{bc} are the panel lengths. A panel set of approximating the body is shown. More points, leading to more panels can be added for a better approximation.

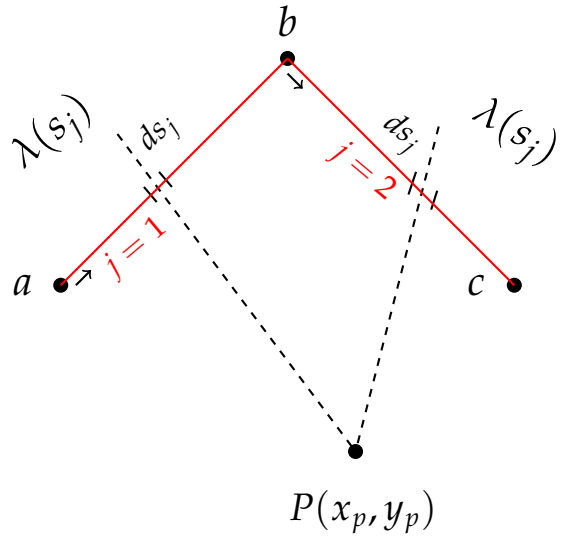

Figure 6.8: Here, we have assigned each panel (red) a constant source strength $\lambda(S_j)$ for $j = 1, 2, 3, ..., n$, leading to n source strengths for n panels. S_j is the jth panel length.

To sum up, the goal is to build a flow around arbitrary objects. The geometry was defined using the boundary and control points which are known values. The angles are needed when calculating the velocities and solving the integral term from eq. (6.14). This will be done in the next section. We found that a point object induces a small potential ϕ at a point P in space. Further, we found an expression of the total potential $\phi(P)$ at point P. Assuming the source strength parameter λ is constant along each panel, allowed us to break up an arbitrary shape into n panels consisting of straight lines. The source strengths λ are located at the control points of each panel. When we combine this with a freestream velocity, we have all the contributions needed in order to calculate the flow around an object of arbitrary shape. In the next section we dive further into the details on how to set up the equations needed for solving the panel source strengths and the velocities.

6.4 Calculating the velocities around an arbitrary shaped object

At the surface of an object the normal velocity component of a flow is zero. This condition will be used to set up a system of equations. Further, to calculate the velocities the tangential component will be used. Finally, the potential can be solved at every grid cell for the defined area of interest given by:

$$\phi_p = V_\infty \cos(\alpha)x + V_\infty \sin(\alpha)y + \sum_{j=1}^{n} \lambda_j \int_j \frac{\ln r_{pj}}{2\pi} ds_j. \tag{6.33}$$

The two first terms are the freestream velocity vectors in x and y direction, respectively. Taking the derivative of the potential ϕ_p in the x and y direction the velocities can be found:

$$V_{x,p} = \frac{\partial \phi_p}{\partial x}, \tag{6.34}$$

$$V_{y,p} = \frac{\partial \phi_p}{\partial y}. \tag{6.35}$$

The normal velocity deal with the physical part of the solution. They can be expressed as:

$$\vec{V} \cdot \vec{n} = 0. \tag{6.36}$$

With the source strengths λ placed at the control points of each panel the normal velocity component will be used to solve for the source strengths. The velocity potential at

panel i can be written using eq. (6.33):

$$\phi_p = V_\infty \cos(\alpha)x + V_\infty \sin(\alpha)y + \sum_{j=1}^{n} \lambda_j \int_j \frac{\ln r_{ij}}{2\pi} ds_j, \tag{6.37}$$

where we sum up the contribution of every source strength λ_j induced at panel i, with $i = 1, 2, 3, ..., n$ and $j = 1, 2, 3, ..., n$ running up to the number of panels n. The normal and tangential velocities at panel i can be expressed from the relation between the velocity potential as:

$$V_{n,i}^{Total} = \frac{\partial \phi_i}{\partial n_i} = V_\infty \cos(\alpha)\frac{\partial x_i}{\partial n_i} + V_\infty \sin(\alpha)\frac{\partial y_i}{\partial n_i} + \sum_{j=1}^{n} \frac{\lambda_j}{2\pi} \int_j \frac{\partial}{\partial n_i} \ln r_{ij} ds_j = 0. \tag{6.38}$$

By looking at the first two terms in the normal direction, the freestream velocity components are denoted:

$$V_{n,i}^{freestream} = \frac{\partial \phi_i}{\partial n_i} = V_\infty \cos(\alpha)\frac{\partial x_i}{\partial n_i} + V_\infty \sin(\alpha)\frac{\partial y_i}{\partial n_i}. \tag{6.39}$$

The change in x-direction with respect to the normal derivatives can be written as:

$$\frac{\partial x_i}{\partial n_i} = \cos(\delta_i) \quad , \quad \frac{\partial y_i}{\partial n_i} = \sin(\delta_i). \tag{6.40}$$

The setup can be seen in fig. (6.9). Inserting eq. (6.39) into eq. (6.40) and using the trigonometric identity $\cos(a \pm b) = \cos(a)\cos(b) \mp \sin(a)\sin(b)$ we end up with simpler term for the freestream velocity:

$$V_{n,i}^{freestream} = V_\infty \cos(\beta_i). \tag{6.41}$$

Recall, β_i is the angle from angle of attack α to δ_i $\beta_i = \delta_i - \alpha$. See fig. (6.5). Putting eq. (6.41) into eq. (6.38):

$$V_{n,i}^{Total} = \frac{\partial \phi_i}{\partial n_i} = V_\infty \cos(\beta_i) + \sum_{j=1}^{n} \frac{\lambda_j}{2\pi} \int_j \frac{\partial}{\partial n_i} \ln r_{ij} ds_j = 0. \tag{6.42}$$

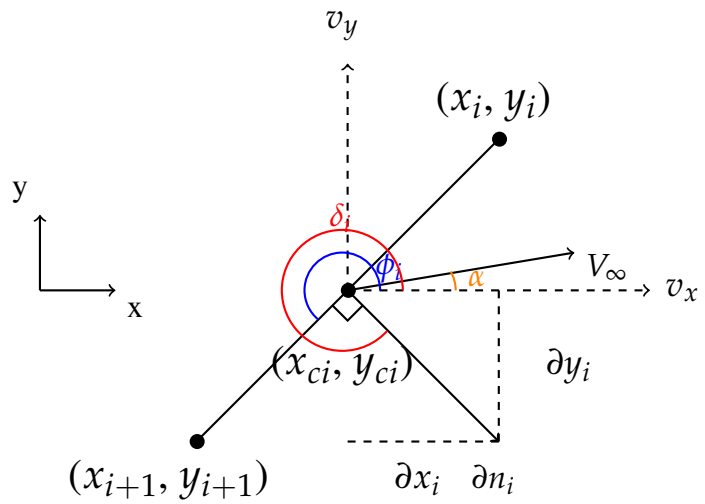

Figure 6.9: Panel i with its corresponding normal components ∂n_i can be related to ∂x_i and ∂y_i which can further be expressed with trigonometric functions. Note, we assume that the normal component points outwards.

6.5 Normal vector geometry integral

In this section we look at the integral part of the total normal velocity induced at panel i. We recall

$$V_{n,i}^{Total} = \frac{\partial \phi_i}{\partial n_i} = V_\infty \cos(\beta_i) + \sum_{j=1}^{n} \frac{\lambda_j}{2\pi} \int_j \frac{\partial}{\partial n_i} \ln r_{ij} ds_j = 0. \tag{6.43}$$

Let us focus on the integral that needs to be solved and express the terms with the use of variables mentioned above. Let the integral be denoted by I_{ij}:

$$I_{ij} = \int_j \frac{\partial}{\partial n_i} \ln r_{ij} ds_j. \tag{6.44}$$

The logarithmic term can be manipulated by the chain rule $\frac{d \ln(f(x))}{dx} = \frac{1}{f(x)} \frac{df(x)}{dx}$. Applying this to eq. (6.44) gives:

$$I_{ij} = \int_j \frac{1}{r_{ij}} \frac{\partial r_{ij}}{\partial n_i} ds_j. \tag{6.45}$$

The distance r_{ij} which is the distance between panel j to panel i, is defined as:

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{6.46}$$

Taking the derivative of the distance w.r.t to normal component yields the equation:

$$\frac{\partial r_{ij}}{\partial n_i} = \frac{(x_i - x_j)\frac{\partial x_i}{\partial n_i} + (y_i - y_j)\frac{\partial y_i}{\partial n_i}}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}. \tag{6.47}$$

From fig. (6.9) and the eqs. in (6.40) we recognize the derivatives in the latter equation as $\cos(\delta_i) = \frac{\partial x_i}{\partial n_i}$ and $\sin(\delta_i) = \frac{\partial y_i}{\partial n_i}$. Substituting eqs. (6.47):

$$\frac{\partial r_{ij}}{\partial n_i} = \frac{(x_i - x_j)\cos(\delta_i) + (y_i - y_j)\sin(\delta_i)}{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}. \tag{6.48}$$

Gathering eq. (6.48), eq. (6.46) and inserting this into eq. (6.45) the equation now reads:

$$I_{ij} = \int_j \frac{(x_i - x_j)\cos(\delta_i) + (y_i - y_j)\sin(\delta_i)}{(x_i - x_j)^2 + (y_i - y_j)^2} ds_j. \tag{6.49}$$

Further, we want to express the integrand in terms of s_j as this is the integration variable. We find that x_j and y_j

$$x_j = X_j + s_j\cos(\phi_j)\ , \quad y_j = Y_j + s_j\sin(\phi_j). \tag{6.50}$$

Here, we represent the point on the jth panel as a parameter representation, where X_j and Y_j are starting points for the panel, and s_j is the length of the jth panel. ϕ_j is angle between the x-axis and the panel j. The angle ϕ_j can be derived from the identity $\cos(\delta_i) = -\sin(\phi_i)$ and $\sin(\delta_i) = \cos(\phi_i)$ where the relation of δ_i and ϕ_i are $\delta_i = \phi_i + \pi/2$ as seen in fig. 11. Inserting this into I_{ij} the integral now reads:

$$I_{ij} = \int_j \frac{(x_i - x_j)(-\sin(\phi_i)) + (y_i - y_j)\cos(\phi_i)}{(x_i - x_j)^2 + (y_i - y_j)^2} ds_j. \tag{6.51}$$

We have been able to express $\frac{\partial r_{ij}}{\partial n_i}$ in terms of the angle ϕ_i, the control point at the ith and jth panel (x_i, y_i) and (x_j, y_j), respectively. In order to solve the integral we express the control points using the parameters representations in eq. (6.50) and integrate over the panel j. For the complete derivation, see Appendix B. The calculated geometry integral is stated and expressed as:

$$I_{ij} = \frac{A}{2}\ln\left[\frac{S_j + 2S_jB + C}{C}\right] + \frac{D - AB}{E}\left(\arctan\left[\frac{S_j + B}{E}\right] - \arctan\left[\frac{B}{E}\right]\right). \tag{6.52}$$

The terms A, B, C, D and E are:

$$\begin{aligned} A &= \sin(\phi_i - \phi_j), \\ B &= (X_j - x_i)\cos\phi_j + (Y_j - y_i)\sin\phi_j, \\ C &= (x_i - X_j)^2 + (y_i - Y_j)^2, \\ D &= (X_j - x_i)\sin\phi_i + (y_i - Y_j)\cos\phi_i, \\ E &= \sqrt{C - B^2}. \end{aligned}$$

Finally, the I_{ij} is solved and the normal velocity component can be computed as follows:

$$V_{n,i} = V_\infty \cos\beta_i + \sum_{j=1}^{n} \frac{\lambda_j}{2\pi} I_{ij} \tag{6.53}$$

This sums up the normal vector computations. The normal geometry integral has been expressed in terms of panel s_j thus making it possible to integrate and finding the contributions of every panel λ_j induced onto panel i. Next, we set the system of equations into matrix form.

6.6 Solving the system of equations

With the I_{ij} integral solved, the next part is to solve for a source strength at each panel. Recall, the normal component is set equal to zero at every control point (x_{ci}, y_{ci}), $i = 1, 2, 3..., n$ from the boundary conditions.

$$V_{n,i}^{Total} = V_\infty \cos(\beta_i) + \sum_{j=1}^{n} \frac{\lambda_j}{2\pi} I_{ij} = 0, \tag{6.54}$$

$$\sum_{j=1}^{n} \frac{\lambda_j}{2\pi} I_{ij} = -V_\infty \cos(\beta_i). \tag{6.55}$$

When $j = i$ the I_{ij} has been found to be $\frac{\lambda_i}{2}$, see Appendix A. The equations above now reads

$$\frac{\lambda_i}{2} + \sum_{\substack{j \neq i \\ j=1}}^{n} \frac{\lambda_j}{2\pi} I_{ij} = -V_\infty \cos(\beta_i), \tag{6.56}$$

and this is inserted into a matrix representing a set of linear equations, giving

$$\begin{bmatrix} \pi & I_{12} & I_{13} & \dots & I_{1n} \\ I_{21} & \pi & \dots & \dots & I_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ I_{n1} & I_{n2} & \dots & \dots & \pi \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \vdots \\ \lambda_n \end{bmatrix} = -2\pi V_\infty \begin{bmatrix} \cos\beta_1 \\ \cos\beta_2 \\ \cos\beta_3 \\ \vdots \\ \cos\beta_n \end{bmatrix}, \tag{6.57}$$

where we have multiplied the matrix equation with 2π. The matrix equation can be solved numerically, granting the source strengths $\lambda_1, \lambda_2, ..., \lambda_n$ of all n panels. The source strength distribution is now solved making it possible to solve for the remaining flow in a defined area.

6.7 Streamline geometry integral

The geometry integral part of the normal and streamline components have a lot in common, but the main idea is to make it clear which equations are needed and in what order to obtain streamlines around a body. With the sources strengths solved for, the velocities at every grid cell can be calculated and are the components needed to visualize the streamlines. We integrate the integral from eq. (6.33) the same way we did for the normal geometry integral. Instead of taking the derivative w.r.t to the ith normal vector $\frac{\partial}{\partial n_i}$, the derivative is taking w.r.t to x and y, $\frac{\partial}{\partial x_i}$ and $\frac{\partial}{\partial y_i}$ for finding the velocity components in the x and y-direction, respectively. As expressed eq. (6.34) and eq. (6.35). The integral has the solution:

$$V_{x,p} = V_\infty \cos(\alpha) + \sum_{j=1}^{n} \frac{\lambda_j}{2\pi} \left[\frac{-\cos\phi_j}{2} \ln\left[\frac{S_j + 2S_jB + C}{C} \right] + \frac{A_{xp}}{E} \left(\arctan\left[\frac{S_j + B}{E} \right] - \arctan\left[\frac{B}{E} \right] \right) \right], \tag{6.58}$$

where $A_{xp} = x_p - X_j + \cos\phi_j B$.

$$V_{y,p} = V_\infty \sin(\alpha) + \sum_{j=1}^{n} \frac{\lambda_j}{2\pi} \left[\frac{-\sin\phi_j}{2} \ln\left[\frac{S_j + 2S_jB + C}{C} \right] + \frac{A_{yp}}{E} \left(\arctan\left[\frac{S_j + B}{E} \right] - \arctan\left[\frac{B}{E} \right] \right) \right], \tag{6.59}$$

where $A_{yp} = y_p - Y_j + \sin\phi_j B$.

To sum up, analytic solutions of the Laplace equation around arbitrary bodies can in general not be done and the requirements for the SPM is knowing the shape of the body which is approximated by a set of panels (piecewise straight line segments) which can be varied for more accurate results. At the center of each panel a small source strength flow is placed and used for deflecting an oncoming stream. The velocity components at the center of the panel are equipped with boundary conditions on the normal velocity. The boundary conditions give rise to a geometric integral where the source strength values are needed. The normal geometric integral is first solved finding the source strength values at the center of each panel. At last, the contribution of every panel strength gives rise to the remaining velocities to be solved in an area, thus giving rise to visualizing streamlines. In the next chapter, how to represent the earth's shape onto a two dimensional plane and calculate distances on the earth's sphere is presented.

Chapter 7

Map projections

The knowledge of the Earth's shape has been known by cartographers for the past 2000 years [18]. The data we will be working with is logged from the Earth's sphere. How to convert a spherical surface onto a rectangular surface will be the focus in the following section. We also mention how and what types of projections can be used for representing a map. These projections and properties are needed when visualizing the AIS data and SPM. The Haversine formula is introduced and used to calculate distances between AIS data.

7.1 Map selection

The data used in the thesis are logged from the surface of a sphere i.e. the Earth. When it comes to visualize the data i.e. AIS data, this is done on a flat surface. Some background is needed in order to find the appropriate representation. This section presents an introduction to maps and the different ways to represent maps in two dimensions. As we know the Earth has a spherical shape, but in geography class the Earth was usually shown to us on a two dimensional map. This is known as a map projection. Flattening the Earth onto a rectangular plane and also attempting to maintain properties such as geometry, area size and distance, we would not succeed (see fig. 7.1). An example is trying to peel an orange and make its shape rectangular. Converting from three to two dimensions leads to distortion [18].

Figure 7.1: Here, the Earth flattened out without any distortion. Linking together the pieces comes at the price of distortion. Figure gathered from [19].

When deciding which projection to use, it is useful to consider the quantity wanting to preserve, as information regarding the map is lost in the conversion. Below we state some projection properties which are commonly considered:

- **Equal-area** preserves the area-size of objects. This is useful when wanting to see the relative size of multiple objects.
- **Conformal** preserves the angle between points. This is important when navigating at sea and when using large- scaled maps.
- **Equidistant** maintains the distance between points. Preserving this quantity comes in handy when wanting to compare several distances.
- **Compromise** considers some distortion from all properties, but attempting to balancing it.

There are three types of methods commonly used when it comes to map projection: cylinder, cone and plane.

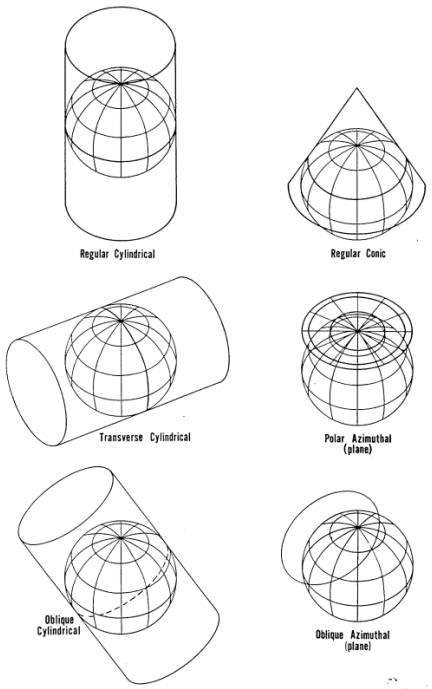

Figure 7.2: Different projections are made from a spherical surface to a flat surface. Drawing straight lines which are orthogonal to the point on the sphere onto the curved plane constitutes a map projection. Figure gathered from [18].

7.2 Geographic coordinate system

A coordinate system is a necessary tool when working with positional data. This gives insight when determining distances in multiple dimensions and from the use of mathematics grants the possibility of calculating positions from a given reference point or relative to given points. When determining positions on a spherical shaped object, e.g. the Earth, one must consider the curvature and how distances are measured. The measures of longitude and latitude have been implemented and defined in order to enable a structured way to define the globe [20]. These are angular measurements and there are several ways of defining latitude and longitude values: geocentric, astronomical or geographic [21]. The latter is most widely used.

Latitude is also known parallels are measured from north to south direction with an interval of 180°. The equator is located at 0°. The Earth is not a perfect sphere. This affects the relative lengths of latitudes. 1° of latitude amounts to 110.567 km at Equator and

111.699 km at the poles.

Longitude, also known as meridians is measured from west to east with an interval of 180°. The prime meridian, which is where 0° is defined, runs through Greenwich, London The relative distances are equal for 1° of longitude and equals 111.32 km [21]. Longitude and latitude coordinates make up a geographic coordinate system. Positional information can be determined in terms of the prime meridian and the Equator.

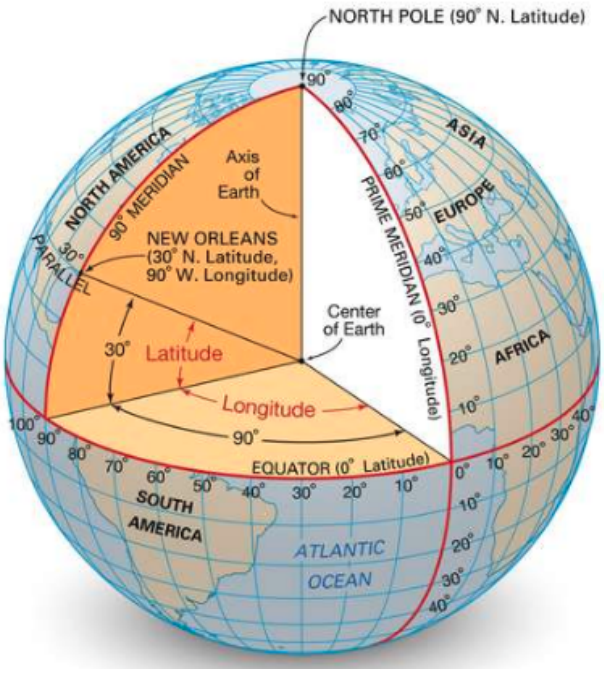

Figure 7.3: How angles of longitude and latitude are defined from the Earth's center is shown here. A unit of latitude change relative to the distance from the poles. Figure gathered from [21].

7.3 Calculating distances on a sphere

When calculating distances on a sphere, specifically the Earths sphere the Haversine formula eq. (7.3) can be applied. The Haversine formula calculates the distance the between two points. The points have longitude and latitude values.

$$a = \sin^2\left(\frac{lat_1 - lat_0}{2}\right) + \cos\left(lat0\right)\cos\left(lat1\right)\sin^2\left(\frac{lon_1 - lon_0}{2}\right), \qquad (7.1)$$

which is used to express:

$$c = 2 \arcsin(\sqrt{a}). \tag{7.2}$$

The distance between two points on the Earth can be calculated with follwing equation:

$$Haverdist = cr, \tag{7.3}$$

where the points have the coordinates (lon_0, lat_0) and (lon_1, lat_1). r is the Earth's radius. The Haversine formula will be used to calculate distances between AIS data which needed for the route extraction algorithm, see next chapter.

Chapter 8

AIS data

A concise introduction of the data used in the thesis is presented. We list up what is contained within an AIS message. A motivation as to why AIS data was introduced in the first place is also mentioned. However, its limitations need to be specified in order to gain insight in the work needed to process and use the data. Calculating vessel velocities will be carried out in order to visualize the vessel's and their corresponding heading and magnitude.

8.1 The development AIS data

The development of AIS data commenced after the environmental catastrophe of the Exxon Valdez oil spillage in 1989 [22]. In 2004 the International Maritime Organization (IMO) made it mandatory for all vessels with a weight over 300 gross tons to install an AIS transceiver [23]. The huge amounts of data being logged has made analyzing AIS data a useful tool within the maritime industry. Being able to predict arrival and waiting times at ports, turnaround times and make accurate decisions in advance related to risks is a benefit for the parties involved.

AIS data is a mandatory self-reporting system installed on maritime vessels originally developed to avoid collisions. It sends out information about a vessels characteristics in real time. This can be seen by other vessels in the vicinity or the data is used by coastal authorities for ship traffic monitoring. The transmitter the AIS system used to communicate is based on Very High Frequency (VHF) bands. The receiver at the other end, typically other ships within the vicinity or vessel traffic services (VTS), use the information for safety measures. Additional receivers may include; buoys and satellites (S-AIS). The data is divided in two types: static and dynamic. Static remain the same after being registered

AIS characteristics		
Variables:	**Specifications:**	**Data type:**
IMO number	International Maritime Organization identification system	Static
ShipType	The vessels classified ship type	Static
SOG	Speed over ground	Dynamic
Longitude	Geographic coordinate horizontal	Dynamic
Latitude	Geographic coordinate vertical	Dynamic
COG	Course over ground/heading	Dynamic
Time Stamp Position	Time of logged data	Dynamic

Table 8.1: The information contained in an AIS message consists of these vessel characteristics and are used as input for the route extraction.

for the first time, dynamic can change and is updated. AIS characteristics are listed in table 8.1.

The transmitter or receiver could cause errors leading to an AIS message or several not being registered. This causes the data set to contain gaps in the time and spatial parameters between the AIS messages. Moreover, a satellite is able to cover a larger area than AIS receivers which can lead to interference between ships [24].

In this thesis, AIS data limited to a time window and restricted to an area around the Trondheimsfjord was provided by DNV GL. The AIS data will help gain practical insight and perspective on the route done by actual vessels. As AIS data is installed by many different types of vessels, it could be interesting to examine if different vessel types take routes depending on the environment.

8.2 Preprocessing AIS data

AIS data provides several variables at a given timestamp. These variables are:

- **International Maritime Organization (IMO) number:** Introduced in 1987 to enhance "maritime safety, and pollution prevention to facilitate the prevention of maritime fraud" [23]. Each vessel gets assigned a unique number used for identifying vessels. Vessels over 100 gross tons are required to carry an IMO number with the exceptions to vessels engaged solely in fishing, vessels without mechanical means of propulsion, pleasure yachts, vessels engaged on special service, hopper barges, hydrofoils and hovercrafts, floating docks, vessels of war and wooden vessels [25].

- **Longitude and latitude:** make up vessel positions in a geographic coordinate system. Their units are angles and make up unique points on a sphere.

- **Heading:** is based on directions on a compass. Used for finding vessels' course measured in angles where 0°and 360° is true north, 90° true east, 180° true south and 270° true west.

- **Speed Over Ground (SOG):** is a vessel speed in magnitude relative to the earth's surface. SOG is measured in knots, ($1\ knot = 1.852\ km/h$).

- **Ship type:** specifies the vessel characteristic: cargo ship, passenger ship, tug boat, tank ships, coast guard ship, pleasure craft, high-speed craft, towing vessel, fishing boats, sailing vessels, ekranoplans and other.

- **Length:** the vessel's maximum length.

- **Time stamp position:** is the registered time of the data.

Data structure

The key elements in a tidy data set is being consistent, easy to access and work with and exploit. Another element to take into consideration is how the computer will be able to extract values. This will help when it comes to analyzing the data [26]. Each row represents an observation and each column represents a variable. According to [26] this is the preferred method in organizing data. Below is an image of how the AIS data is structured from file:

	IMONumber	TimestampPosition	Longitude	Latitude	Heading	SpeedOverGround	ShipType	Length	unix
1	9536521	2018-06-04 00:00:04	10.270005	63.47954	271	8.3	cargo_ships	89	1528070404
2	9536521	2018-06-04 00:10:05	10.21974	63.4794083333333	268	8.1	cargo_ships	89	1528071005
3	9536521	2018-06-04 00:20:06	10.1665833333333	63.47936	268	9	cargo_ships	89	1528071606
4	9536521	2018-06-04 00:30:14	10.108645	63.4789383333333	269	9.3	cargo_ships	89	1528072214
5	9536521	2018-06-04 00:40:24	10.05038	63.4776783333333	274	9.1	cargo_ships	89	1528072824
6	9536521	2018-06-04 00:50:24	9.99481333333333	63.4795516666667	289	8.9	cargo_ships	89	1528073424
7	9536521	2018-06-04 01:00:04	9.94693	63.489025	307	8.2	cargo_ships	89	1528074004
8	9536521	2018-06-04 01:10:06	9.91273833333333	63.5058183333333	325	8.4	cargo_ships	89	1528074606

Figure 8.1: An overview of raw AIS data. Each row represents observations of a vessel at given time with characteristics of the vessel. See section 8.1 for detailed description column representation

As seen from fig. (8.1) the structure of the AIS data is up to standard with the guide lines from [26]. Next, we calculate the velocity components of each observation by using two columns from the data set.

Calculating velocities

Velocities are useful units to calculate. This grants the possibility to visualize a vessels direction and heading. Here, we will see which vessels travel to and from Trondheim. Vessels may take a detour, traveling through Trondheim to then converge to Trondheim after a while. From fig. (8.1) the velocities can be found by using columns: $Heading$ and $SpeedOverGround$. The velocities can be found with these two equations:

$$v_{lon} = SOG \cdot \sin(heading) \tag{8.1}$$

$$v_{lat} = SOG \cdot \cos(heading) \tag{8.2}$$

v_{lon} and v_{lat} are the velocity components in the corresponding longitude and latitude direction, respectively. SOG is the vessels magnitude of velocity and $heading$ is the vessels angle in degrees. Note, $heading$ is measured from north, i.e. north $= 0°$ [27].

Route extraction

Visualizing the data can be done in many ways. Plotting every data point can be messy and misleading, as the data set consists of many data points which is the case here. Also, we are only interested in looking at a subset of the data, that is, vessels travelling toward Trondheim, as the goal will be located at Trondheim. Some information about how one route emerges and what classifies a route is needed. The data is not sorted in terms of a given destination, but by looking at the timestamp column in the data set, there is often a trend of logged data from a certain time interval, more specifically the change in dates.

The first and last data point of a given date could indicate the start and end of a route. Note that this will only not be possible for the complete data set. We will use the Haversine formula mentioned in section 7.3 to keep track of the vessel position at each time stamp. A simple route classification will be conducted. The vessel routes will be classified into routes travelling *To*, *From* or *Trough* Trondheim. By checking the start and end point of a route, routes having a larger distance at the end point than start point is classified as *From*, and else *To*. In addition, some vessels could be travelling through Trondheim. This is solved by calculating the distance at each time stamp. Vessel with an increase in distance for more than a given amount of time stamps, is classified as *Through*. The route extraction algorithm will be used to compare the vessel routes to the SPM. More details on this in the implementation part.

Part II

Implementation

Chapter 9

Analytical solutions

The framework and tools needed for visualizing and calculating the flow field in a given region is presented together with a route extraction algorithm for classifying routes. All programs written uses the Python programming language. This chapter is dedicated to the analytical solutions to the Laplace equation. We set up a class instance for calculating the velocity components of the different flow characteristics derived in the theory section. Chapter 10 introduces Basemap, a library for converting data to 2D maps using map projections. This is needed for the visualizing the SPM and AIS data. A route classification based on raw AIS data is carried out in Chapter 11. This will be used to compare the flow fields with actual vessel trajectories. In Chapter 12, we apply the SPM in Trondheimsfjorden by placing source strengths at the coastlines and calculating the strength coefficients. Also, a sink potential is added at Trondheim representing the goal.

9.1 Elementary flows

The solutions of Laplace equation give rise to elementary flow types. Using the superposition principle leads to different flow fields. A class instance for each flow characteristic is created and initialized with a flow strength λ and location of origin in 2D space $(x0, y0)$. Each method takes in a defined grid calculating the corresponding velocities at each grid cell. The velocity contributions are then added together for the final flow-field which can be visualized. An example of how to initialize a flow at point $(x0, y0)$ with strength coefficient λ ($lamda$ in the code) is shown in the code snippet below.

```
flow = Flow.Velocity(lamda, x0, y0)
u_source, v_source = flow.source(X, Y)
```

A class object is created from the program $Flow.py$ containing the class $Velocity()$. Note,

lambda is a reserved word in Python. Next, the flow characteristic is specified. The velocity components u, v are returned for a defined grid X, Y. X and Y are matrices of size nxm where n is the number of grid cells defined in the x-direction and m the number of grid cells in the y-direction. The total contribution of each individual flow are added together creating the final flow field. See flow chart for class instance in fig. (9.1). In a real world case, traversing through environments contain obstacles e.g. islands or reefs. By using the fundamental flows, we will look at how these elementary flows can be used to represent obstacles e.g. islands. We assume islands are circular shaped. It has been found in the literature that doublets can represent islands from [10] and [28]. However, the source flow has a flow characteristic moving radially away from its center point, which makes it a potential candidate for object representation and will be tested here.

Each individual flow type needs to define its corresponding flow strength coefficient. The strength coefficients can be regulated and adjusted affecting the streamline paths. For the analytical case, we must attempt to simulate an environment with fundamental flows and creating a flow field resulting in streamlines with a defined start and end point (or goal). Often, multiple streamlines lead to the goal. From a motion planner perspective, picking a suitable streamline to follow is based on the vessels characteristics. For the object representation, the flow strength must be large enough for the flow to actually flow around the object, as the potentials are defined from a point. The flow representing objects is located at the center of each objects. We will examine how different flow strengths for the start, end and obstacle representation will affect the streamlines. The program is executed as follows: From the *Flow.py* program we initialize a class object $Velocity()$ given the parameters flow strength and location (x_0, y_0). Next, we specify the flow type source, sink or doublet which is a method from the class. Each method requires as input a defined grid. The $np.meshgrid$ from the numpy library in Python is commonly used. The method returns the velocity components u, v in the x and y- direction, respectively. With every flow type defined we add together every contribution at each individual grid cell and plot the resulted flow field with standard matplotlib plotting tools such as $streamplot()$.

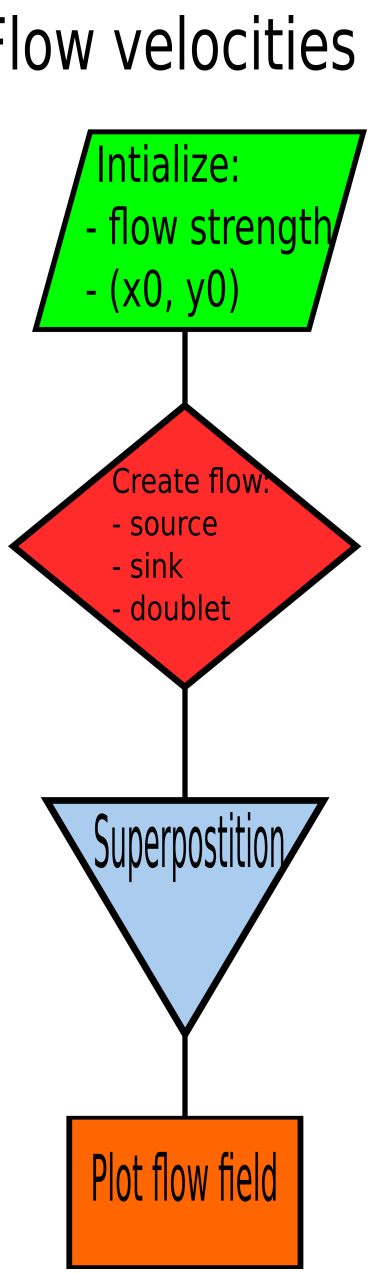

Figure 9.1: Flow chart for class instance $Velocity()$ for calculating the velocities at every grid point. This can create trajectories based on streamlines of the flow. Each flow type is assigned a flow strength, and point of origin $(x0, y0)$. A source and sink flow are commonly used to represent a start and end point, respectively. A doublet or source can be used to represent obstacles.

Chapter 10

Basemap

10.1 Creating a map

Creating a map in Python is done with the Matplotlib Basemap Toolkit. Basemap can plot geospatial data onto different map projections [29]. The necessary requirements for Basemap :

- Matplotlib 1.0.0 (or later, download)
- Python 2.6 (or later, including Python 3)
- Matplotlib 2.2 LTS requires Python 2.7 or later
- Matplotlib 3.0 requires Python 3.5 or later
- NumPy 1.2.1 (or later)
- Array support for Python
- PROJ4 Cartographic Projections Library

If using anaconda environment, Basemap can be installed by the following command in the terminal [30]:

```
conda install -c anaconda basemap
```

When creating a map, a regular Python class object instance is made. The arguments passed into the Basemap object are the boundary points, projection type and resolution. Plotting the map is done using Matplotlib's standard plotting commands. An example of initializing a Basemap class instance is shown in the code below:

```
from mpl_toolkits.basemap import Basemap
m = Basemap(llcrnrlon = minlon, llcrnrlat = minlat , urcrnrlon = maxlon
    , urcrnrlat = maxlat,
            resolution = 'i', projection = 'cyl', lat_0 = lat0, lon_0 =
    lon0)
m.drawmapboundary(fill_color = 'white')
m.fillcontinents(color = 'lightgrey', lake_color = 'white', zorder=1)
```

An object m is created and defined with the parameters: lower left corner longitude ($llcrnrlon$), lower left corner latitude ($llcrnrlat$), upper right corner longitude ($urcrnrlon$) and upper right corner latitude ($urcrnrlat$). How to define these values is mentioned in more detail in the next section. Basemap allows to choose 4 different resolutions: $'c'$ (crude), $'l'$ (low), $'i'$ (intermediate), $'h'$ (high) and $'f'$ (full). There are 24 projection types to choose from, here a cylinder projection types is used. The center of the map can be set with lon_0 and lat_0. Maps can be formatted using attributes in Basemap. Two attributes are shown in the code above, $drawmapboundary()$ which draws a line around the defined area, $fillcontinents()$ as the name suggests, fills the continents with a color of choice making it possible to see land areas. There are many more attributes and we will need several more for collecting the coastlines from the map. The necessary attributes are mentioned below and the remaining can be found in the documentation [11].

10.2 Setting map boundaries

Given the AIS data we now can limit our map to the maximum and minimum longitude and latitude values of the data. The range of longitude and latitude degrees are (-180, 180) and (-90, 90) respectively. The prime meridian is located at 0° of longitude which runs through Greenwich, England. Negative longitude values represent the western hemisphere and positive values the eastern hemisphere. For degrees of latitude, the 0° is located at equator. Negative latitude values represent the southern hemisphere and positive values the northern hemisphere. The maximum and minimum longitude and latitude values can be found using the standard max and min functions in Python:

```
minlon = max(-180, min(df['Longitude']))
minlat = max(-90, min(df['Latitude']))
maxlon = min(180, max(df['Longitude']))
maxlat = min(90, max(df['Latitude']))
```

The dataframe df contains the vessel characteristics. Chapter 12 explains in more details how to set up a Pandas dataframe df. The maximum and minimum longitude and latitude values are used as input in the Basemap class instance. The projection chosen is cylindrical making the representation rectangular preserving the distance in the projection. See fig.

(10.1) below.

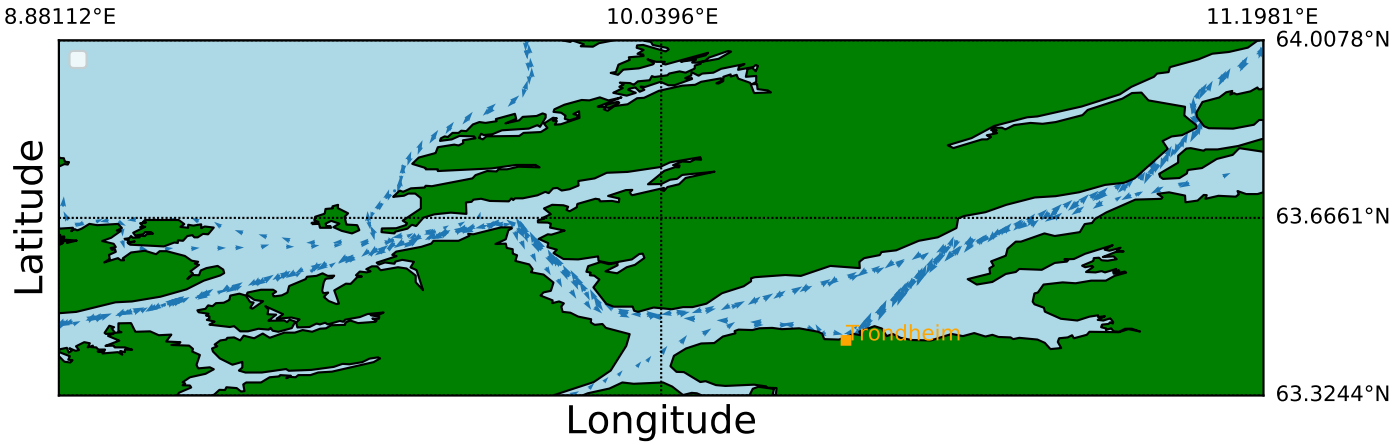

Figure 10.1: An overview of Trondheimsfjorden defined from the max and min points from the data set is shown. All tugboats in the AIS data set which is logged from 4/6/2018 to 13/10/2019 in the fjord of Trondheim. The blue arrows indicate the vessels direction of travel. The resolution in Basemap is set to *high*. Note how the vessel follow certain paths regardless of direction.

Trondheimsfjorden will be the area in which we will apply the SPM. There appears to be a trend of routes states occurring in the AIS data as seen in fig. (10.1). We will examine the Timestamp column in the data set and look to see if there is any correlation between the Timestamp and a route. This is conducted in the next chapter.

Chapter 11

Application of AIS data

This chapter is dedicated to extracting AIS data with Pandas. A route classification algorithm is carried out, classifying the routes in three states, vessels travelling *To*, *From* and *Trough* Trondheim. The vessel routes will be compared with streamlines produced from the SPM.

11.1 Creating routes with Pandas

Pandas is a Python library intended for data analysis. A Dataframe (df) type object lets a user manipulate a data set by indexing, slicing, subsetting, merging, accessing rows and columns to only name a few operations. Pandas requires Python 3.6.1 or higher and can be installed with anaconda [31]:

```
conda install -c anaconda pandas
```

or if using Python Package Index (PyPI) [32]

```
pip install pandas
```

Pandas supports different types of files format. Here, the file type will be comma-separated-value, more commonly known as a csv file and can be read in with Pandas as:

```
1 import pandas as pd
2 df = pd.read_csv('Routing/aisToTrondheim.csv', index_col = '
    TimestampPosition', parse_dates = True)
```

The csv file *aisToTrondheim* is located in the folder "Routing". Pandas allows for several ways to index the data. We choose to index on the column "TimestampPosition" column. This is read in as an argument when reading the file.

New columns can be added by treating the column as if it already exists in the df. We create a new column in the df and calculate the distance between the end point Trondheim and every registered positional coordinate eq. (7.3). Calculating distances on a sphere is done with the Haversine formula eq. (7.3) in the *convert.py* program:

```
df['Heading'] = np.radians(df['Heading'])
df['VelocityLongitude'] = df['SpeedOverGround']*np.sin(df['Heading'])
df['VelocityLatitude'] = df['SpeedOverGround']*np.cos(df['Heading'])
df['DistTrondheimHaversine'] = convert.distsphere(df['Longitude'],
    trond_lon, df['Latitude'], trond_lat)
```

New columns are added to the df, *Velocity Longitude*, *Velocity Latitude* and *DistTrondheimHaversine* and *Heading* use the existing ones to convert from degrees to radians.

The next step is to examine the data. The data set has dimension $71694x12$. The TimestampPosition column in fig. (8.1) will be used by comparing the interval of each timestamp to sample routes from the data set. Each column represent a variable for specific timestamp. Working with all that data becomes too complex making it difficult to extract information when visualizing. In addition, since there are many different vessels types e.g. with different size this will affect how each vessel type traverses. A subset of the data is drawn out for a clearer view. We create a new df, df_{subset} consisting only of one single vessel type:

```
df_subset = df[df['ShipType'].isin(['tugboats'])].copy()
```

The subset df_{subset} now consists of one vessel type (in this case tugboats) from the main data set. The objective now is to extract routes from the data subset as the only order in the data set is their respective timestamp. From the timestamp column it can be seen that on the same day the data contains a fixed interval of logged AIS data varying from seconds to minutes. One could argue that this is the same vessel being tracked.

A variable holding all unique dates in the subset is needed for separating the data even more. This assumes that every data point holding the same date constitute one unique route in the subset. The number of data points vary from date to date. We need to keep track of the number of data points in each route. At last, for each set of routes made, we want to find vessels travelling to Trondheim. Each defined route does not necessarily share the same end point. There are various reasons for this, it could be due to a malfunction or vessels are not necessarily bound to travel to the same destination. We choose to

separate each route into three categories: vessels travelling *To*, *Through* and *From* Trondheim. This classification is of interest when using the flow field, i.e. we want every vessel to converge to the same point, making it possible to compare the vessel trajectories and streamlines created from the SPM.

For each route we check if the distance is decreasing or increasing and make the assumption: If the last absolute distance value is larger than the first, we conclude that the vessel is travelling *From* the destination. And vice versa for a vessel travelling *To*. Vessels travelling through is measured by counting the number of times the distance increase occurs. If the count is larger than a given number (in the program defined as *count*), we conclude the route is travelling *Through*. By making this classification, we can easily discard routes not of interest. Below we show the implementation:

```
for i in range(unique_dates.shape[0]):
    route = df_subset.loc[str(unique_dates[i])]
    last_dist = route['DistTrondheim'][-1]
    first_dist = route['DistTrondheim'][0]

    if last_dist < first_dist:
        df_subset.loc[route.index, 'Direction'] = 'To'
        count = 0

        # check route going to if starting to deviate from dest.
        for j in range(route.shape[0] - 1):
            next_dist = route['DistTrondheim'][j + 1]
            prev_dist = route['DistTrondheim'][j]
            if next_dist < prev_dist:
                #dist decreasing
                count = 0
            else:
                #dist increasing
                count += 1
            if count == 3:
                #Ship could be travelling through
                df_subset.loc[route.index, 'Direction'] = 'Through'
                break
    else:
        df_subset.loc[route.index, 'Direction'] = 'From'
```

To access the classified routes, $df_{ToTrondheim}$ collects all vessels classified as *To* from the *Direction* column and gather the corresponding variables needed to plot and visualize:

```
df_ToTrondheim = df_subset.loc[df_subset['Direction'] == 'To',
                               ['Longitude', 'Latitude', '
   VelocityLongitude', 'VelocityLatitude', 'DistTrondheim']]
```

$df_{subset}.loc$ locates every data points with the $'Direction'$ with string values equal to $'To'$. We can easily access addtional information from that timestamp index such as $'Longitude', 'Latitude', 'VelocityLongitude', 'VelocityLatitude', 'DistTrondheim'$ which is stored in the dataframe $df_{ToTrondheim}$. With the use of Pandas we have been able to extract subsets of data which in this case makes it less chaotic when visualizing the data. We have been able categorize and produce routes from the data. In addition, a basic classification scheme determining the vessel's direction and end points is provided.

Chapter 12

Application of SPM in Trondheimsfjorden

In order to calculate the velocities using the SPM in Trondheimsfjorden, source potentials are located at the coastlines for the flow to be deflected. The coastlines will represent the panels in the SPM. The necessary coastline components for computing and implemented the SPM are then acquired. Basemap, with its built-in utilities, can collect the coastlines as line segments. This is explained in more detail below together with the functions used to calculate the velocities and streamlines. From the previous chapter, we found that some routes are travelling toward Trondheim. We will attempt to create streamlines from the SPM converging at Trondheim. An additional sink potential is located at Trondheim to help each streamline to converge towards Trondheim. This will be the final step.

12.1 Computing the source strengths

The first step for applying the SPM is to obtain the coastlines from the map. The data is gathered from a geographical data set [33]. This object holds information about the coastlines which can be retrieved further by the class attribute:

```
1 coast = m.drawcoastlines()
2 coordinates = coast.get_segments()
```

coordinates holds a list of arrays with the start and end points of a line segment

$(x_0, y_0), (x_1, y_1), ..., (x_n, y_n)$. Each set of points make up the polygonal representation of each connected land area. These points will be the boundary points (x_b, y_b) which is needed to calculate the control points (x_c, y_c) which is where the source potentials λ are located, see fig. (6.3). The necessary components needed to compute the source strengths λ are now obtained. Note, we assumed from the SPM that each λ are constant, but can vary in strength. Given the boundary point's the control points (x_c, y_c) can be calculated with eq. (6.21) and (6.22), respectively. The angles ϕ, δ and β are acquired using first eq. (6.23), then eqs. (6.24) and (6.25). The angles are stored in arrays with dimension $nx1$ where n is the number of panels.

From the boundary points, we have calculated the necessary components needed for the geometric integral. Solving the system of equations for the source strengths λ is one of the last steps to solve for the flow field and streamlines. First, a code snippet is presented showing how to access the control points (x_c, y_c), the angles β, δ and ϕ and the panel lengths *panels*:

```
XC, YC = FlowBoundary.control_points(XB,YB)
panels, phi = FlowBoundary.panels(XB, YB)
delta, beta = FlowBoundary.angles(phi, AoA)
```

From the program *Flowboundary.py*, functions for calculating the components needed for computing the source panel strengths are given here. AoA is the angle of attack.

For solving the geometric integral for the normal velocity component eq. (6.44), an empty matrix, with dimension nxn where n is the number of panels, is set up. The I_{ij} geometric integral is calculated by the terms from eq. (B.19), where A,B,C,D and E are listed below . S_j is the length of panel j in the notation of. For all $j \neq i$ eq. (6.57) fills every element of the matrix, leaving us with the diagonal elements (see code snippet below). As stated before it was found that when $j = i$ the I_{ij} has the value of $\frac{\lambda_i}{2}$.

To solve a set a linear equations, the numpy module $np.linalg.solve$ is used. The matrix must be square, have full rank i.e the columns or rows must be linearly independent [34]. First, the diagonal elements of the matrix need to be filled with π from eq. (6.57) for the case when $j = i$ which means physically when we calculate the total contribution of potential induced at itself. With the matrix produced from the python program *sourcepanel.py* where the function $solveGeometricIntegrals()$ is implemented, we use the array of β values found from $flowboundary.py$ and thus solve $I\lambda = b$ where I is the geometric integral values, b is the array β multiplied with $-2\pi V_\infty$. The full matrix

equation can be seen in eq. (6.57).

```
I = sourcepanel.solveGeometricIntegrals(XC, YC, panels, phi, beta, XB,
    YB)
np.fill_diagonal(I, np.pi)
b = -2*np.pi*Vinf*np.cos(beta)
lamda = np.linalg.solve(I, b)
```

Computing the geometric integral I and furthermore solving the system of equations from the matrix equation $I\lambda = -b$

12.2 Computing velocities

After computing the source strengths in a defined area we can compute the velocities components which will be used to visualize the streamlines of a fluid. First, we define a number of grid cells (nx, ny), more accurate results require more grid cells which leads to a higher computational load.

In order to help the streamlines converge to the wanted location, namely Trondheim, we place a sink potential at the desired goal. After calculating the velocities with the defined grid points we use the superposition principle, adding the contributions from every source panel and additional potentials to every grid cell.

Finally, we compute the velocity components at every defined grid cell in the area of interest. This is done with eqs. (6.58) and (6.59). Note that from the latter equations the sink potential located at the end point is not included. Therefore, the final equation for computing the velocities at a point P due to the potential induced at that point reads:

```
XGeom, YGeom = sourcepoint.solvePointGeometry(XP, YP, panels, phi, XB,
    YB)
```

```
vx[i, j] = Vinf*np.cos(AoA) + np.dot(lamda, XGeom.T)/(2*np.pi) +
    vx_sink[i, j]
vy[i, j] = Vinf*np.sin(AoA) + np.dot(lamda, YGeom.T)/(2*np.pi) +
    vy_sink[i, j]
```

Source panel method

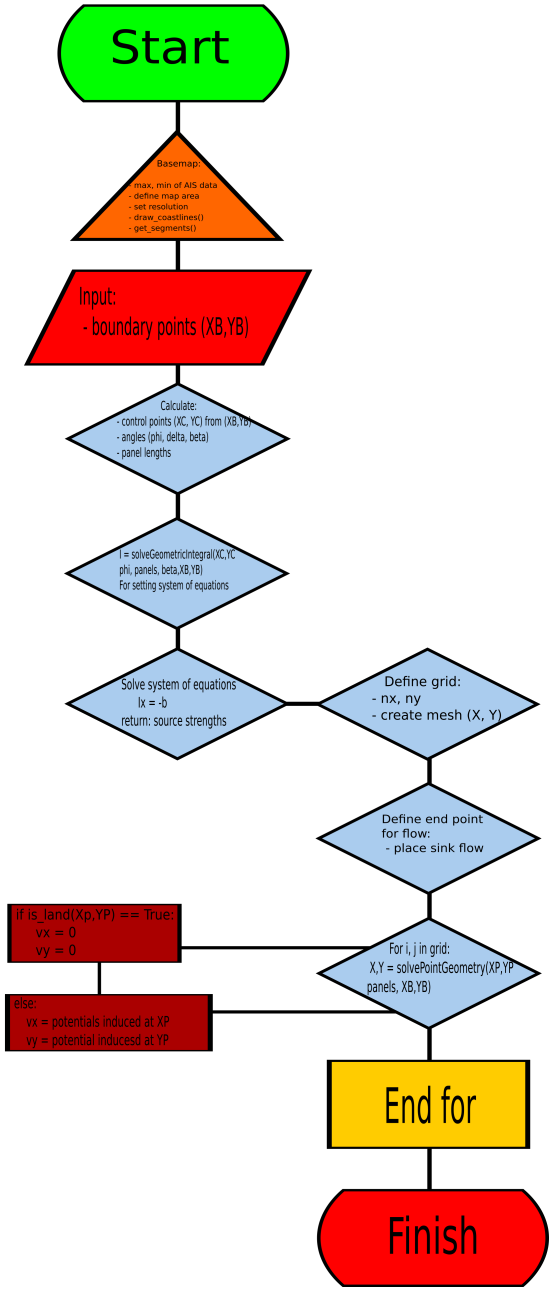

Figure 12.1: Initializing a Basemap class object gives access to several attributes, $get\_segments()$ is used to access the boundary points which is then used calculate the control points, angles and panel lengths in $Flowboundary.py$. This is used to calculate the normal geometry integral I_{ij} from $solveGeometryIntegral()$. The system of equations grant the source strengths at the control points. A grid is defined, and a sink potential located at Trondheim. The last step is to loop through every grid cell and check if each grid cell is located on land where we give the velocity component is set to zero. If the grid cell is on sea, we calculate the contribution on every source and sink potential induced at that point.

To sum up: a map is created from the max and min longitude and latitude values of the data set, setting the boundaries of the map area. We obtained the coastlines from Basemap' attribute $get\_segments()$ making it possible to set out the control points (x_c, y_c) at mid point of each coastline segment (or panel) and calculate the angles needed for solving the source panel strengths λ. Furthermore, the unknown source strengths λ give rise to n equations for n panels. The source strengths are solved for by representing the n equation in matrix form eq. (6.57). An intermediate step was to set out a sink potential flow making flow more likely to converge toward the goal, Trondheim. With source strengths computed and the sink set out, the final task was finding the velocity at a point P induced by the source strengths yielding the velocity components of the flow at every point in the defined grid.

Part III

Results

Chapter 13

Analytical flow solutions

Our main assumption and approach for creating a ship route simulator is based on the assumption that the path a vessel will follow, is similar to the streamlines in a fluid flow. This part presents two solutions, the first applying the analytical solutions in areas where circular objects e.g. islands are present using the Laplace equation to produce flows. Next, by increasing the object complexity, we apply the SPM in Trondheimsfjorden and set the end goal in Trondheim. We examine how the SPM is able to generalize in producing streamlines converging at Trondheim from the possible passages in the fjord. The route extraction for AIS data sets up the vessel's trajectories toward Trondheim which is used to verify and compare the fluid flow. The goal is to obtain streamlines which converge in Trondheim from different areas on the map.

13.1 Analytical flows around circular objects

For circular objects and geometries we found that doublet or source flows are good candidates for imitating flows around circular objects. Note, not every choice for a streamline is optimal. Some streamlines lead too close to the object in front and could in worst case lead to a collision with the object. This becomes an instance of optimizing with respect to safety or economy. Travelling closer to objects could lead to a shorter path to the goal, but increasing the risk of collisions. The object geometry needs to be simple enough, as the analytical solution is only able to produce simple flow characteristics. Having a wide selection of streamlines to choose from is convenient, as different vessel types will seldom choose the same route. From figs. (13.1) and (13.2) we show streamlines produced from the use doublet and source, respectively.

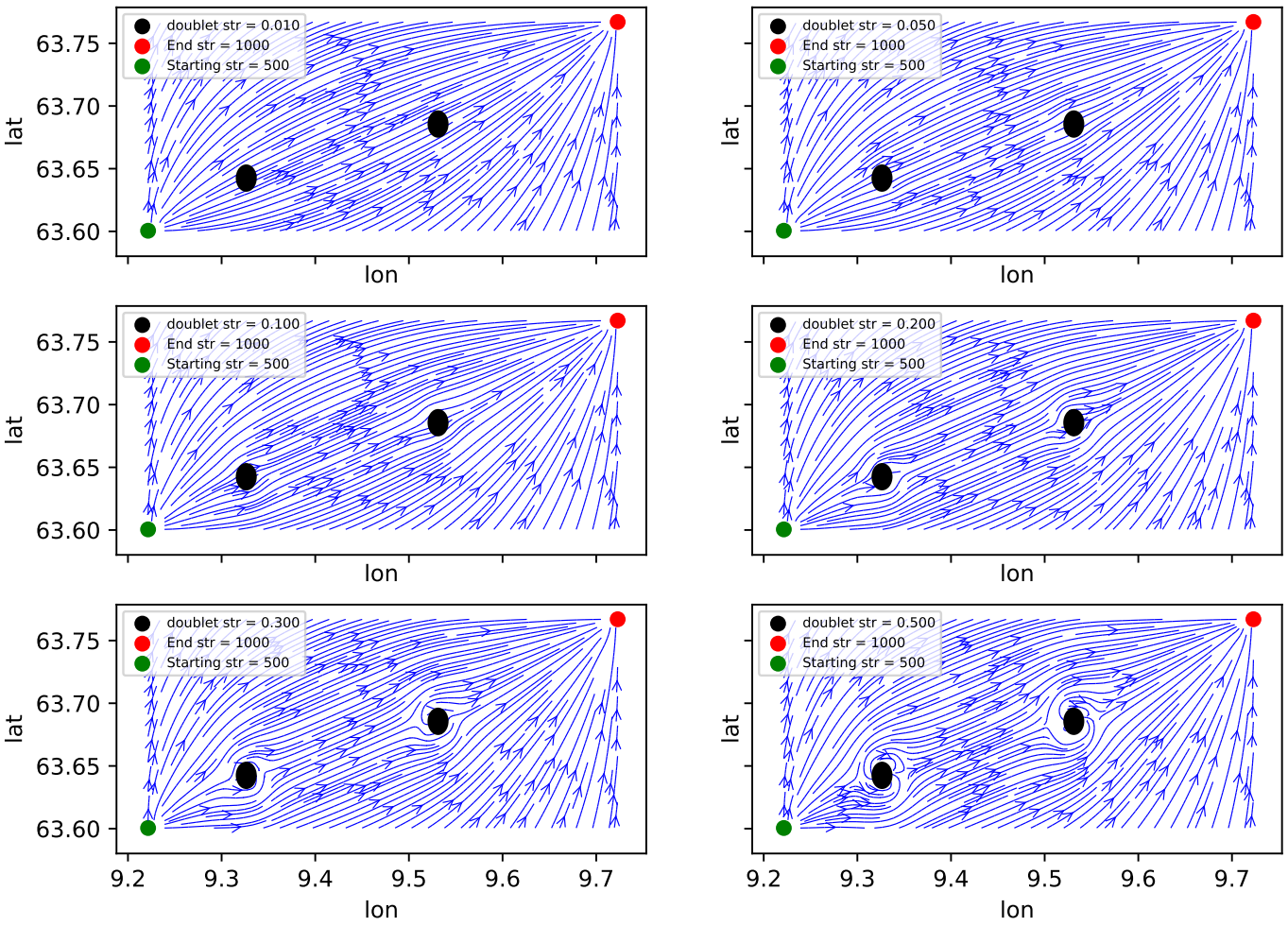

Figure 13.1: Source and sink flow are set as start (green marker) and goal (red marker), respectively. The doublet flow is set out at the center of each island (black circle). Different doublet strengths are set for each subplot with increasing strength coefficients starting, with the lowest at top left and largest at lower right. The arrows indicate the direction of travel. Note the lower right image, as the flow around each island is not a realistic representation of a vessel trajectory.

From fig. (13.1) in the top left image, we see that the doublet strength is not strong enough to create a deflection around the objects placed in flow, but with an increase in strengths as shown in the two images in the middle of fig. (13.1) the doublet flow pattern starts to emerge and shape the flows imitating the flow around the objects. The streamlines

close to the object tend to bend away slightly from the object which indicates that the strengths are more correctly adjusted. However, the lower right image suggests the doublet strength being too large causing streamlines close the objects to converge in the back end of the objects, leading to a collision. This is the main critic of the doublet method. The paths created from the starting point (green) to the goal (red) are many, making possible to traverse on the left or right side of the object which is possible for all strengths coefficients expect for the lower right case having $\lambda_{doublet} = 0.5$. However, when representing larger objects, increasing the strength coefficient would be useful as seen in the lower right, as the deflection is larger and the streamlines converging at the backside of the islands would not be a possible choice, as larger islands will cover those specific streamlines. It is also necessary to point out the strength coefficients for the starting source flow and goal sink, as these will also change the streamlines profile. These were set to a large constant value relative to the island strengths. The reason being that this assures a convergence at the goal and that the starting source potential is not dominated by other potentials set out in the area. Next, we use the source type flow to represent objects.

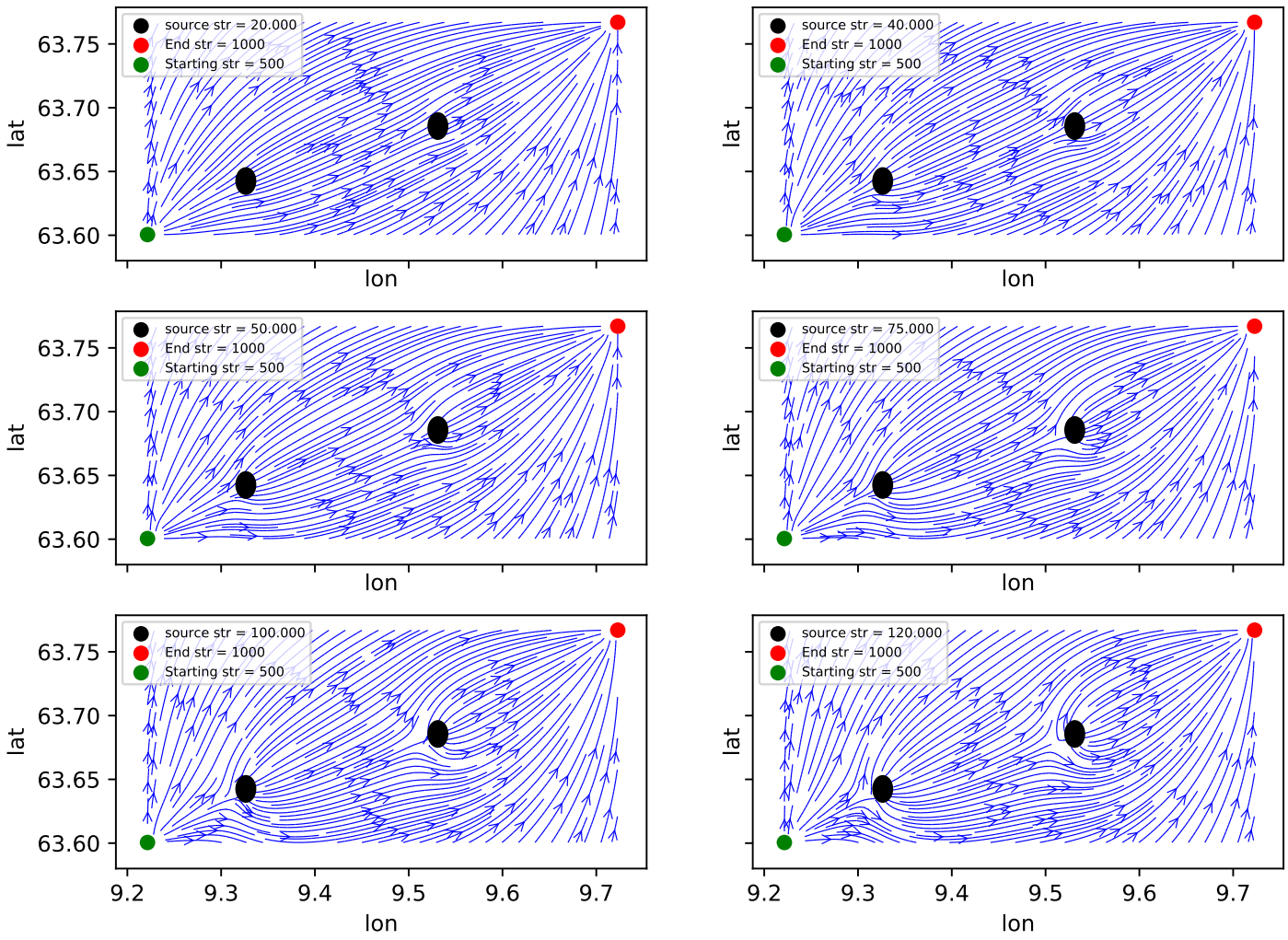

Figure 13.2: Different source strengths for islands (black circle) can be seen here together with a starting point (green marker) and goal (red marker). The flow changes together with the strengths. The source type flow is placed in the center of each island (black object). The arrows indicate the direction of travel. Large bends occur for larger strength coefficients.

As for the source flow case, representing objects with source potentials has some similarities to the doublet potential, see fig. (13.2). In the upper left image the source strengths placed on the objects are not large enough to cause a deflection. In the upper right and middle images, the flow slowly starts to create a deflection around the objects. Also, for

the lower right the deflection is exaggerated compared to the size of the island. This was similar to the doublet case as well. By comparing the doublet flow in fig. (13.1) and source flow in fig. (13.2) we also notice a difference in the way the deflection is caused from the source and doublet flow characteristics. The effect of the source flow characteristic around the objects create streamlines which seem to only bend away from the object, not being affected after passing the objects. This is not the case for the doublet flow characteristics where we see from fig. (13.1) that the streamlines tend to deflect and curve around at the front and end of the objects.

The two flow characteristics used to represent objects e.g. islands in an area could be used for different vessel types. By using the doublet flow representation, smaller and faster vessels seem to have these types of trajectories, as their ability to turn faster the shortest route would be most preferable path for these types of vessels. For larger vessels where the route is mainly determined by the shortest amount of turns, the source flow characteristics could be a good candidate for these vessel types, as they would most likely be interested in large turns together with the smallest amount of turns in a selected route. The strength coefficient has a big effect on the streamlines produced. It is unclear what the appropriate strength coefficient value should be. There was not found any information regarding this in the literature. It would seem there exists an interval of values which could create realistic streamlines which could substitute vessel trajectories.

Chapter 14

SPM

14.1 Verifying the SPM in Trondheimsfjorden

We investigate how the SPM performs in the following section. Trondheimsfjorden consists of several passages leading toward Trondheim making it possible to enter from the northwest or northeast side. We show several cases with different parameters selected, the sink potential located in Trondheim λ_{sink}, freestream velocity V_∞ and angle of attack AoA. Collision-free streamlines toward the goal is the desired output. In addition, we limit the map, examining the passage only possible to enter from the northeast side of Trondheim. From these cases we discuss how the SPM performs in the different parts of the fjord. First, we examine two cases with different parameters including the whole area of Trondheimsfjorden in figs. (14.1) and (14.2).

From fig. (14.1), the produced streamlines from the SPM in Trondheimsfjorden is shown. The uniform flow was set to be positive for all $X < Trondheimlon$ and negative else in order to assist the flow toward Trondheim. Certain streamlines show a lack in the ability to follow the coastlines leading to many streamlines crashing into the coastlines or being lead into bays trapping the flow. From the lower left narrow passage, streamlines tend to bend, ending up in the bay in the lower left not producing any streamlines toward Trondheim. In the upper mid, streamlines crash into coastlines resulting in streamlines also not able to be as a possible path toward Trondheim. As for the upper right, streamlines only seem to be affected by the freestream velocity. Note also, that every area does not contain streamlines (mid and top right). This is caused by an error in streamline function in matplotlib not able to produce streamlines in those areas, as there appears to be non existing values in those areas. We expect the streamlines to be deflected more at the coastlines by the source potentials, but this is not the case in every coastal area. This would suggest the strength

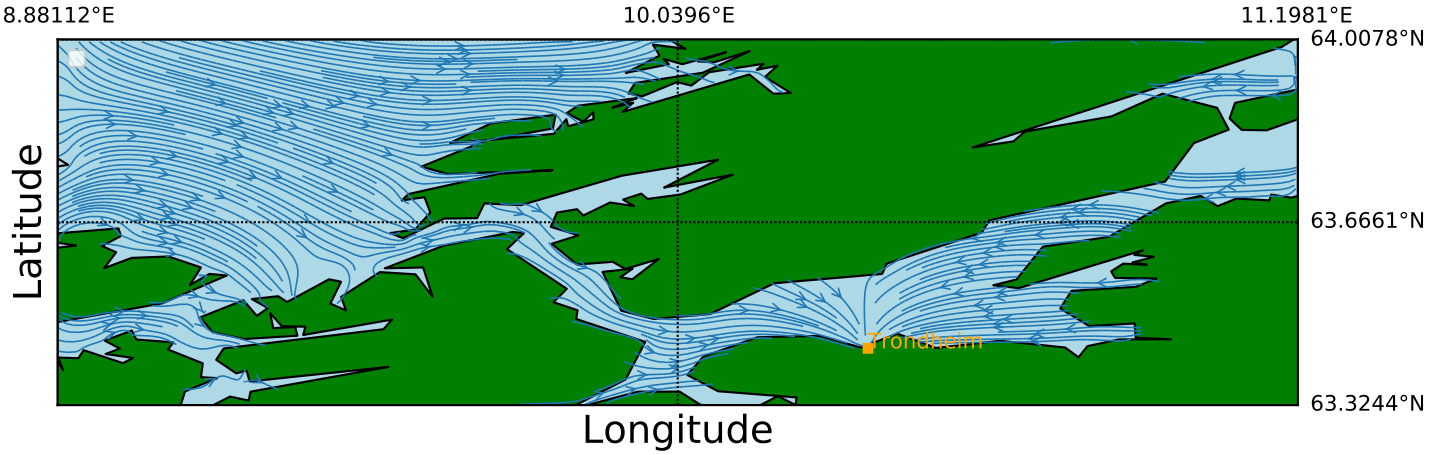

Figure 14.1: Streamlines produced from SPM with a sink flow located at Trondheim together with an uniform flow. The uniform flow is set to be positive for all grid cells $X < Trondheimlon$ and negative for all other points. X is a matrix holding the x coordinates in the area. The angle of attack, $AoA = 0$ gives a uniform flow in the x-direction and no contribution in the y-direction. The strength coefficient of the sink is set to $\lambda_{sink} = 1$.

contributions from other potentials are larger than potentials at the coastlines, neglecting the source potentials at the coastlines. However, the passage from the middle relative to Trondheim produce streamlines leading all the way toward Trondheim, but only from a certain point at the far left side. With this, one could argue that the SPM has the potential of creating a streamline toward the desired goal given a starting point. Note, the starting point can found by tracing the streamline to its origin.

The SPM in certain areas tend to have a sink potential rather than source which should be the case. The normal component located at the control points was assumed to be pointing outward relative to the object. However, it not unlikely that the normal component has been reversed leading to the normal component pointing inwards. This would explain the sink behaviour in those areas.

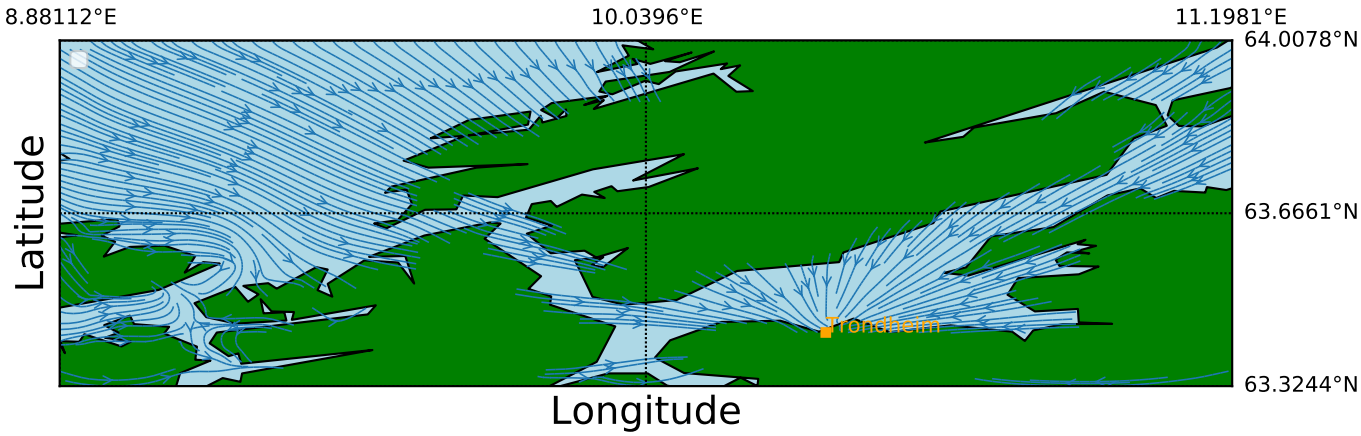

Figure 14.2: A change in the uniform flow contribution for SPM compared to fig. (14.1) is performed. The parameters are set to: $AoA = -\frac{\pi}{2}$, $\lambda_{sink} = 0.1$ and $V_\infty = 0.01$. Note, the change in streamlines produced compared to fig. (14.1).

In fig. (14.2), a trend of straight streamlines often neglecting the coastlines strengths appears to be the dominating factor. Even though the strengths AoA, λ_{sink} and V_∞ are set lower than in fig. (14.1), the dominating potential seems to be the sink located in Trondheim. This explains the radial streamline profile. However, in the lower left area, as in fig. (14.1) a chaotic trend in the streamlines occur. It is unclear what the cause is, one possibility could be that the contribution from every coastline source potential has a different orientation. The neighboring velocity components would then be affected very differently, resulting in a significant difference in the flow direction.

Similar to the analytical case, from figs. (14.2) and (14.1), adjusting the parameters for V_∞, AoA and λ_{sink} impacts the streamlines significantly. Note, the strength coefficients solved at the coastlines are the same in both cases. From the two cases shown in figs. (14.1 and 14.2) the SPM changes a considerable amount by adjusting AoA, V_∞ and λ_{sink}.

Let us consider choosing a selected part of the map, creating a smaller and less complex object geometry. The streamlines will now be constrained to come in from one side, as for the general case (figs. 14.1 and 14.2) there are multiple. The map size is defined based on the maximum longitude and latitude values from a single route and longitude and latitude values of Trondheim. This sets up the appropriate map size when comparing the streamlines with a real route. This is will be conducted in the coming sections.

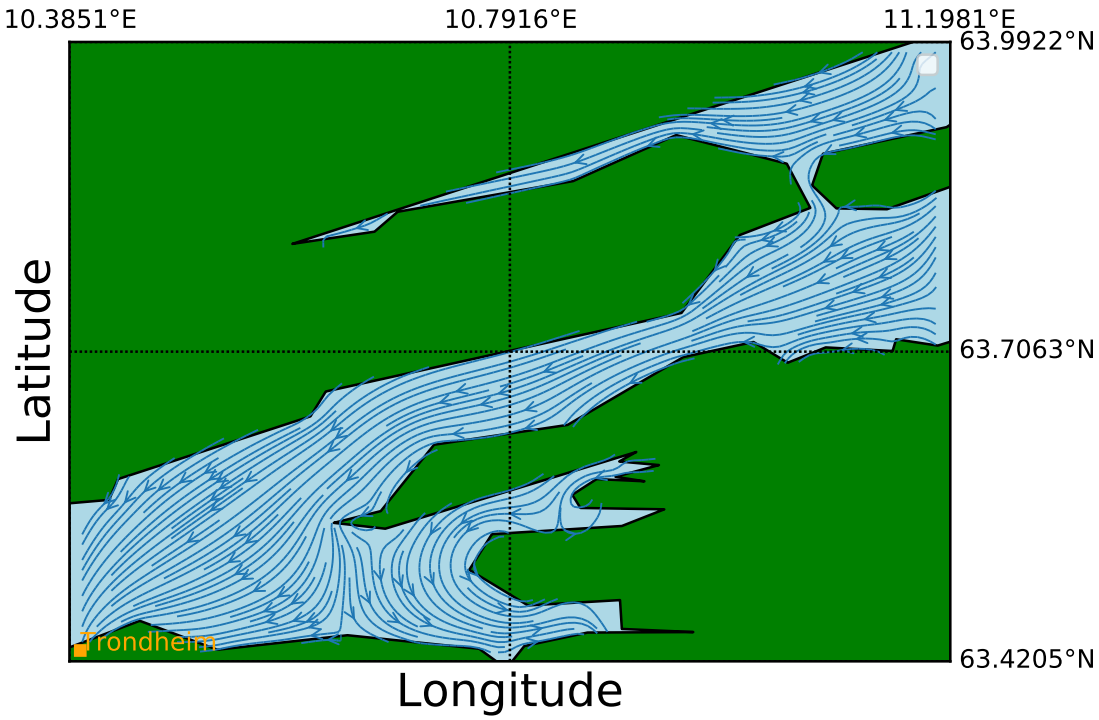

Figure 14.3: A close up of Trondheimsfjorden is shown. The angle of attack $AoA = \arctan(\frac{y_{goal}-y_{start}}{x_{goal}-x_{start}})$. (x_{goal}, y_{goal}) are the longitude and latitude coordinates of Trondheim, (x_{start}, y_{start}) is a coordinate set to top right area which is a possible passage. Note the streamlines in the lower mid bending toward the bay.

By limiting the map only looking at the east side of the fjord is provided in fig. (14.3). To assist the streamlines toward Trondheim the $AoA = \arctan(\frac{y_{goal}-y_{start}}{x_{goal}-x_{start}})$, as the streamlines leading toward Trondheim are limited to one angle of incidence. From fig. (14.3), the SPM is able to create several streamlines converging toward Trondheim. However, starting in the bay to the right of Trondheim would not be able to assist in the path to Trondheim. Note the separation of streamlines at the right of Trondheim leading to a change in the streamlines direction.

The resolution of Basemap determines the amount of boundary points being produced, thus the panel lengths. Setting a high resolution e.g. $high$ or $full$ leaded to numerical errors when calculating panels lengths with these selected map boundaries, more specifi-

cally the E term in the geometric integral eq. (B.19) was equal to zero. This resulted in the matrix I becoming singular, thus leading to source strengths not being solved. The E term becoming zero could be due to the distance between the control and boundary points being too small when increasing the resolution. The information lost from decreasing the map resolution is significant, especially in areas containing islands which are not included for a lower resolution setting, see fig. (14.4). In addition, the attribute $get\_segments()$ was not able produce correct coastline information on these two resolution settings. The highest possible resolution was $'i'$ (intermediate) being able to draw correct boundary points and giving the correct representation of the map. Some adjustments were still necessary. The area around Trondheim does not have continuous coastline and islands in the fjord. Going from different land areas, additional lines were drawn which also created several boundary points and control points in areas where no control points were supposed to be located. These were needed to be located and removed manually by plotting up the boundary points and removing the unwanted points from the array. Additional control points were created at the map boundaries for flow not able to escape. As with the analytical solution, parameters are able to be adjusted. This includes flow strengths such as sink strength at Trondheim and uniform flow which are not part of the coastline source strengths. This changes the streamlines and are as well as for the analytic solution very important to adjust correctly in order to get a good flow representations.

14.2 Classifying routes

A classification is provided, making it possible to extract routes in a given state. From fig. (10.1) every data point of every registered tugboat is plotted. This is not an optimal setting when we only want consider vessels travelling to Trondheim. Fig. (14.4) shows how the route extraction scheme is able to classify routes. For this case, we would also like to point out that not all classification done are with the equal amount of significance or validity. The reason being the lack of data which could be caused by malfunction at either sender or receiver or the algorithm not being robust.

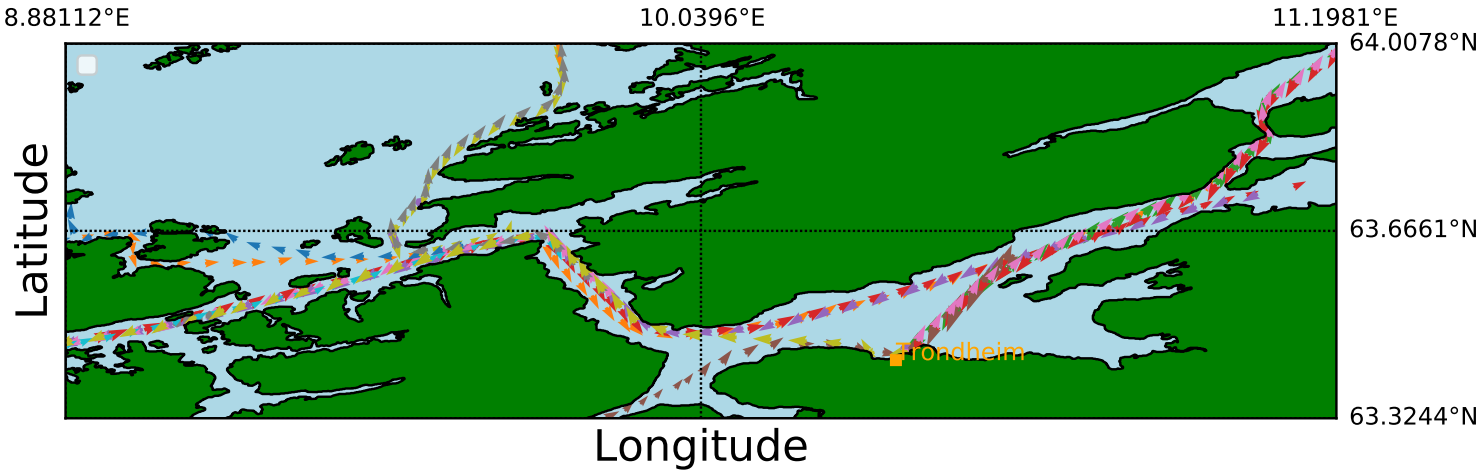

Figure 14.4: A subset of the data containing all tugboats are seen here. Each color represents a unique route. Some similar traversing is done going to and from Trondheim. Some tugboats are going straight past Trondheim which must be handled by route extraction. The resolution is set to $full$ for a complete overview of Trondheimsfjorden. Note, regardless of the direction the vessel's tend to follow the same paths. Islands are included in this resolution setting

With the classification implemented for AIS data we are not interested in vessels travelling *from* or *through* Trondheim, only *to* Trondheim. The projection style chosen is equidistant cylindrical. Preserving the distance between points and thereafter comparing them is what is mainly focused on here. As seen in fig. (14.5) at upper part we see only two AIS data points. Not all vessel paths end up in Trondheim, emphasizing that the data registered does not necessarily contain a complete amount of data points toward Trondheim. In addition, the amount of data points vary from path to path.

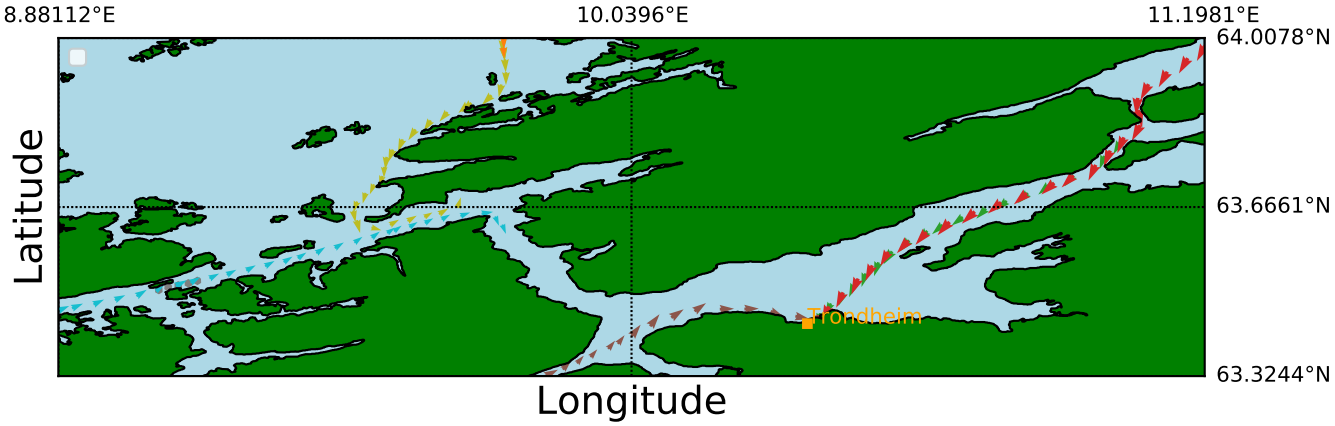

Figure 14.5: Here, all *To* Trondheim vessel are plotted. As we see, the amount of AIS data contained in the individual routes vary from route to route. Compared with fig. (14.4) all tugboats shown are travelling toward Trondheim.

To check the validity of the extraction algorithm presented would be useful. However, by comparing figs. (14.4 and 14.5), the paths stored in the *To* column from the classifying algorithm seems to be accurate enough for extracting routes. In fig. (14.5) each data points for each path there seems to be a logical route being drawn for each route with anomalies occurring.

14.3 Comparing SPM with vessel route

We examine the results from the SPM and compare them to AIS data. We test the case in same area as in fig. (14.3). From the program we are able to select one streamline, but several streamlines could be selected for a more thorough and complex comparison. The selected streamline is selected based on the first AIS data point. Within a given interval of the first AIS data point, many streamlines occur, therefore we select the one streamline having the lowest deviation to the vessel path. Deviations occur between the selected streamline and vessel route. But the AIS data following the streamline to some extent. In general island or reefs could be present, but are not shown with this resolution. This is likely the case and can be verified from fig. (14.5), just above and to the right of Trond-

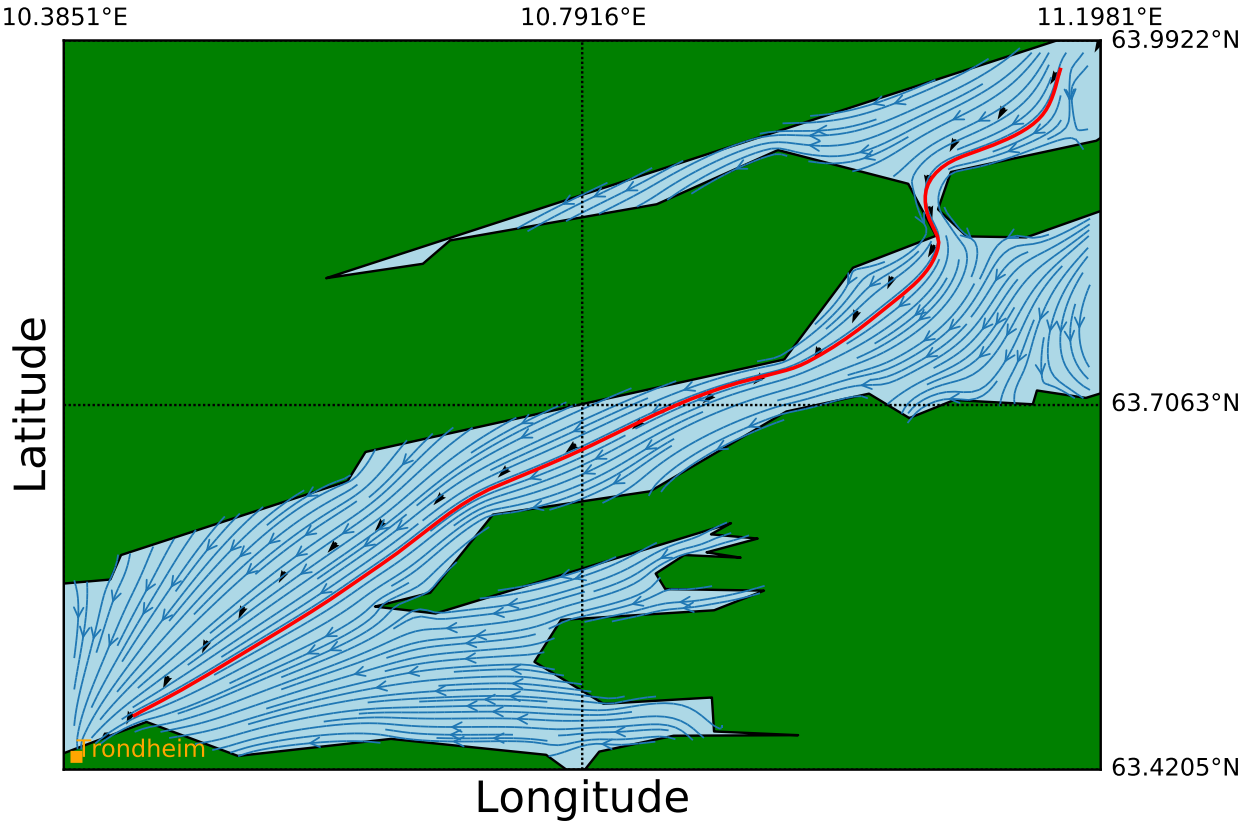

Figure 14.6: The SPM produced streamlines with AIS data is shown. A suggested streamline (red line) compared with an extracted route (black arrows). Note choosing different streamlines will not necessarily lead to Trondheim and the deviation occurring between the chosen streamline and the vessel before converging in Trondheim.

heim, an island is located. Unfortunately it was not possible to increase the resolution including the island.

The goal of this thesis was to simulate a vessel's trajectory with a streamline produced by the SPM. We have found that the SPM has some limitations and shortcomings. Regarding the resolution setting from Basemap, it was detected that not all resolution settings were able to run with the SPM. We saw that from fig. (14.6), a vessel deviates from a selected streamlines because of an island present. The island is shown with a higher resolution and can be seen in fig. (14.5). Also, a division by zero occurs in the calculations of the source strengths, leading to a singular matrix which is not able to be solved. The resolution usable for producing streamlines also has some limitations, as it was shown that the SPM in certain areas does not seem to be affected by the source strength at the coastlines. This could be caused by some bugs in the code, or the contributions from the sink strength in Trondheim or the freestream velocity strengths were too large, neglecting some source strengths.

The SPM methodology may need further modifications in order to be able to produce satisfactory results in the whole of Trondheimfjorden and be able to generalize. The freestream velocity V_∞, AoA and λ_{sink} needs to be examined for setting the correct strengths relative to the source strengths at the coastlines, as there seems to be a significantly difference in the output when adjusting the three parameters. However, from fig. (14.6), we find that the SPM has the potential of being used as a substitute for AIS data in certain areas.

Chapter 15

Conclusions and future work

In this thesis, an analytical and numerical solver for simulating marine vessel trajectories has been presented. With the assumption that vessels can be represented by the flow in a fluid, the motivation and intention for this approach was to investigate how the produced flows could help fill in lacking AIS data in a given area. In areas containing objects with simple geometries, an analytical solution was applied. We found that the source type flow could also represent obstacles, as obstacles often were described as doublets in the literature [10]. The two flow characteristics could be used to guide vessels depending on the vessel's characteristic. Parameter adjustments were necessary in order to create satisfactory flows around the objects.

The SPM was applied and tested on a real area, Trondheimsfjorden with the goal located in Trondheim. Coastal information was obtained and used as input (in the shape of boundary points) in the SPM, leading to a system of equations, thus solving the source strength coefficients located at the control points of each panel. Another finding was that the SPM was not able to generalize or perform well in the whole area of Trondheimsfjorden, as the streamlines in certain areas did not lead to the desired goal. It may have been caused by numerical errors due to the limited resolution of the coastline. This should be investigated further. Therefore, we narrowed down the area and found that the SPM was able to perform significantly better and we compared a selected streamline with an extracted route which correlated to some degree. The SPM clearly has some limitations and shortcomings which must be further examined, but the potential of being a substitute for missing AIS data is shown in this thesis.

15.1 Future work

- Parameter optimization: AoA, v_{sink} and V_∞ are free parameters, and as we have seen, affects the streamlines produced and no information about their size was found in the literature. A parameter range or optimal choice would be beneficial.
- Division by 0 in the normal geometry integral: The E term in the normal geometry integral tends to zero for small distances between the boundary points gathered from the map. This influences the model leading to a less accurate model output. However, [17] claims that two E terms could be used when calculating the source strengths; for $E^2 > 0$ and $E = 0$.
- Extraction of boundary points: When extracting boundary points from Basemap, additional points were sometimes created not being part of the coastline. The order of boundary points were obtained are not always in a ordered manner, leading to lines drawn from each land area and creating additional control points.
- Fluid flow trapped: In certain areas with bays the fluid gets trapped. A method for locating bays and setting a criteria preventing the fluid of getting trapped would very useful, as this could occur in many areas of the world. This could also be an effect from the coastlines and boundary conditions. A thorough analysis of the boundary conditions may lead to a different fluid flow, which might prevent trapping of fluid.
- Create flows based on the AIS data: Another approach could be to add the velocity components from the AIS data to the SPM, thus being sure that flows produced will simulate the vessel trajectories. This will only apply for the areas where the data exists.
- Local SPM model instead of general: Contributions of potentials far away are not necessarily relevant for local flow solutions, we hypothesize that this could affect the output of the SPM. Instead of solving the complete set of coastlines in the area, but solve local land area.
- Setting the normal component to a small value: By increasing the normal component could lead to the source potentials at the coastlines to deflect the flow more which could contribute to a more obstacle avoidance.
- Test on several areas: How the model works in other areas to see how the model generalizes once the methodology has reached a more mature state would be useful to consider.

Appendices

Appendix A

Normal velocity component derivations

In this section, we derive the source panel strength magnitude from a single panel. From section 6.2 when calculating the normal velocity component, it was stated that the contribution from a panel on itself lead to a singularity in the distance variable r_{ij}. We show that the contribution from a panel on itself is $\frac{\lambda}{2}$. Suppose a panel located in vertical position at $(0, y)$ with size $[-L, L]$. The potential induced at an arbitrary point $P = (x, y)$ from a small element dl of the panel can be expressed as:

$$d\phi = \frac{\lambda dl}{2\pi} \ln(r) = \frac{\lambda dl}{2\pi} \ln(x^2 + (y - l)^2)^{1/2}, \tag{A.1}$$

where ϕ is the small potential induced at point (x, y) and λ the source strength coefficient. The small element dl is located at $(0, l)$. The total potential induced from the panel is found by integrating over the whole panel. Leading to the expression:

$$\phi(x, y) = \frac{\lambda}{4\pi} \int_{-L}^{L} \ln(x^2 + (y - l)^2) dl. \tag{A.2}$$

We find the velocity component in x-direction $u(x, y)$ by differentiating eq. (A.2) w.r.t x from the relation $u(x, y) = \frac{\partial \phi}{\partial x}$. Because of the way the panel is located in space, $u(x, y)$ corresponds to the normal velocity component of the panel. The velocity component gives:

$$u(x, y) = \frac{\partial \phi}{\partial x} = \frac{\lambda}{2\pi} \left(\arctan\left(\frac{y + L}{x}\right)\right) - \arctan\left(\left(\frac{y - L}{x}\right)\right). \tag{A.3}$$

Now, we let $x \to 0$ from negative and positive side.

$$u(0^-, y) = \lim_{x \to 0^-} \frac{\lambda}{2\pi} \left(\arctan\left(\frac{y + L}{x}\right)\right) - \arctan\left(\left(\frac{y - L}{x}\right)\right). \tag{A.4}$$

From the calculations done from the velocity was found to be:

$$u(0^-, y) = -\frac{\lambda}{2}, \qquad u(0^+, y) = \frac{\lambda}{2}. \tag{A.5}$$

Appendix B

Normal vector geometry integral

Here we calculate the normal vector geometry integral I_{ij}. Starting with the geometry integral expression from section 6.5:

$$I_{ij} = \int_j \frac{(x_i - x_j)(-\sin(\phi_i)) + (y_i - y_j)\cos(\phi_i)}{(x_i - x_j)^2 + (y_i - y_j)^2} ds_j. \tag{B.1}$$

For the sake of tidiness, the numerator will be considered first. Thereafter, the denominator will be considered. From the above equation the numerator reads:

$$(x_i - X_j - s_j\cos(\phi_j)(-\sin(\phi_i)) + (y_i - Y_j - s_j\sin(\phi_i)\cos(\phi_i)), \tag{B.2}$$

after inserting the relations $x_j = X_j + s_j\cos(\phi_j)$ and $y_j = Y_j + s_j\sin(\phi_j)$. Further we arrange the terms such that:

$$(X_j - x_i)\sin(\phi_i) + (y_i - Y_j)\cos(\phi_i) + s_j(\cos(\phi_j)\sin(\phi_i) - \sin(\phi_j)\sin(\phi_i)), \tag{B.3}$$

and use the trigonometric identity $\sin a - b = \sin a\cos b - \cos a\sin b$ on the last term:

$$(X_j - x_i)\sin(\phi_i) + (y_i - Y_j)\cos(\phi_i) + s_j\sin(\phi_i - \phi_j). \tag{B.4}$$

We write the numerator in terms of s_j:

$$s_j A + D, \tag{B.5}$$

where $A = \sin(\phi_i - \phi_j)$ and $D = (X_j - x_i)\sin(\phi_i) + (y_i - Y_j)\cos(\phi_i)$. The numerator is now simplified and can be integrated.

Moving on the denominator. From eq. (B.1) we write out the terms and insert the relations $x_j = X_j + s_j\cos(\phi_j)$ and $y_j = Y_j + s_j\sin(\phi_j)$:

$$x_i^2 - 2x_i(X_j + s_j\cos\phi_j) + (X_j + s_j\cos\phi_j)^2 + y_i^2 - 2y_i(Y_j + s_j\sin\phi_j) + (Y_j + s_j\sin(\phi_j)^2. \tag{B.6}$$

Now we gather all s_j terms with same polynomial degree. The denominator can then be written as:

$$s_j^2 + 2s_jB + C, \tag{B.7}$$

where $B = X_j\cos\phi_j + Y_j\sin\phi_j - x_i\cos\phi_j - y_i\sin\phi_j$ and $C = (x_i - X_j)^2 + (y_i - Y_j)^2$. The end term with the numerator and denominator is put back into eq. (B.1). We end up with the expression:

$$I_{ij} = \int_0^{S_j} \frac{s_jA + D}{s_j^2 + 2s_jB + C} ds_j. \tag{B.8}$$

Before integrating we need to complete the square in the denominator.

$$s_j^2 = 2Bs_j + C = s_j^2 + 2Bs_j + B^2 + C - B^2 = (s_j + B)^2 + E^2, \tag{B.9}$$

where $E = \sqrt{C - B^2}$. Now the integral reads:

$$I_{ij} = \int_0^{S_j} \frac{s_jA + D}{(s_j + B)^2 + E^2} ds_j, \tag{B.10}$$

which is on the form that can be integrated. Setting $u = s_j + B$ leads to the expression:

$$I_{ij} = \int_0^{S_j} \frac{(u - B)A + D}{u^2 + E^2} du. \tag{B.11}$$

The integral will be split up into two integrals by separating the numerator.

$$I_{ij} = A\int_0^{S_j} \frac{u}{u^2 + E^2} du + (D - AB)\int_0^{S_j} \frac{1}{u^2 + E^2} du. \tag{B.12}$$

Using substitution $\gamma = u^2 + E^2$ on the first integral again we obtain the solution as:

$$A\int_0^{S_j} \frac{u}{u^2 + E^2} du = \frac{A}{2}\int_0^{S_j} \frac{d\gamma}{\gamma} = \frac{A}{2}[\ln\gamma]_0^{s_j}. \tag{B.13}$$

Now we substitute u and γ

$$\frac{A}{2}\Big[\ln(S_j + B^2 + E^2) - \ln(B^2 + E^2)\Big], \tag{B.14}$$

$$\frac{A}{2}\ln\left[\frac{(S_j+B)^2+E^2}{B^2+E^2}\right]=\frac{A}{2}\ln\left[\frac{S_j^2+2S_jB+B^2+E^2}{B^2+E^2}\right]. \tag{B.15}$$

With the final touch of using $E^2 = C - B^2$ we end up with:

$$\frac{A}{2}\ln\left[\frac{S_j+2S_jB+C}{C}\right]. \tag{B.16}$$

The second integral in eq. (B.12) has the solution:

$$(D-AB)\int_0^{S_j}\frac{1}{u^2+E^2}du=\frac{(D-AB)}{E}\arctan\left[\frac{u}{E}\right], \tag{B.17}$$

as with the first integral we substitute back and end up with:

$$\frac{D-AB}{E}\left(\arctan\left[\frac{S_j+B}{E}\right]-\arctan\left[\frac{B}{E}\right]\right). \tag{B.18}$$

Let us sum up the most important parts of this integration below. The integration gave the solution:

$$I_{ij}=\frac{A}{2}\ln\left[\frac{S_j+2S_jB+C}{C}\right]+\frac{D-AB}{E}\left(\arctan\left[\frac{S_j+B}{E}\right]-\arctan\left[\frac{B}{E}\right]\right). \tag{B.19}$$

The terms A, B, C, D and E are:

$$\begin{aligned}
A &= \sin(\phi_i-\phi_j),\\
B &= (X_j-x_i)\cos\phi_j+(Y_j-y_i)\sin\phi_j,\\
C &= (x_i-X_j)^2+(y_i-Y_j)^2,\\
D &= (X_j-x_i)\sin\phi_i+(y_i-Y_j)\cos\phi_i,\\
E &= \sqrt{C-B^2}.
\end{aligned}$$

Appendix C

Software and hardware

The software used in this thesis involves Python on a Ubuntu operating system. The calculations done were performed on a computer provided by the University of Oslo. The computer used to run all simulations contains the specifications: 16GB of RAM, Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz processor using Ubuntu 18.04 operating system.

The code implemented in the thesis can be found on Github:
user: Thomassjaastad, repository: Master-thesis.

Appendix D

Bibliography

1. UN-Business Action Hub, United Nations. *Business.Un.Org* https://business.un.org/en/entities/13. 2020.
2. Wang, X., Li, J. & Zhang, T. A Machine-Learning Model for Zonal Ship Flow Prediction Using AIS Data: A Case Study in the South Atlantic States Region. en. *Journal of Marine Science and Engineering* **7,** 463 (Dec. 2019).
3. Pallotta, G., Vespe, M. & Bryan, K. Vessel Pattern Knowledge Discovery from AIS Data: A Framework for Anomaly Detection and Route Prediction. en. *Entropy* **15,** 2218 (June 2013).
4. Meijer, R. *ETA Prediction Predicting the ETA of a Container Vessel Based on Route Identification Using AIS Data* PhD thesis (2017).
5. Arhus, G. H. & Salen, S. R. Predicting Shipping Freight Rate Movements Using Recurrent Neural Networks and AIS Data. en, 78 (2018).
6. Besse, P., Guillouet, B., Loubes, J.-M. & François, R. Review and Perspective for Distance Based Trajectory Clustering. en. *arXiv:1508.04904 [cs, stat]*. arXiv: 1508.04904 [cs, stat] (Aug. 2015).
7. Hvamb, K. Motion Planning Algorithms for Marine Vehicles. en, 153 (2015).
8. Corman, M. *Dijkstra's Algorithm* https://medium.com/@madelinecorman/dijkstras-algorithm-1c6d93c0ea39. 2020.
9. LaValle, S. & Kuffner, J. J. Rapidly-exploring Random Trees: Progress and Prospects (1998).
10. Pedersen, M. D. & Fossen, T. I. Marine Vessel Path Planning & Guidance Using Potential Flow. en. *IFAC Proceedings Volumes* **45,** 188 (2012).

11. matplotlib. *Basemap api* https://matplotlib.org/basemap/api/basemapapi.html.

12. Gjevik, B. *Innføring i Fluidmekanikk* (University of Oslo, 2009).

13. Acheson, D. *Elementary Fluid Dynamics* (Oxford, 2003).

14. Anderson, J. D. *Fundamentals of Aerodynamics* 2nd ed. en
(McGraw-Hill, New York, 1991).

15. Kundu Pijush K., C. I. M. D. D. R. *Fluid mechanics fifth edition* p. 115
(Elsevier Inc., 2012).

16. Versteeg, H. K. & Malalasekera, W.
An Introduction to Computational Fluid Dynamics 2dn edition (Pearson, 2007).

17. Kim, J.-O. & Khosla, P. K.
Real-Time Obstacle Avoidance Using Harmonic Potential Functions. en, 28.

18. Snyder, J. *Map Projections- a Working Manual* 1987.

19. Šavrič, B. & Kennedy, Melita. *Map Projections*
https://storymaps.arcgis.com/stories/ea0519db9c184d7e84387924c84b703f. 2020.

20. Kher Aparna. *Latitude and Longitude*
https://www.timeanddate.com/geography/longitude-latitude.html. 2020.

21. Encyclopedia britannica. *Latitude and Longitude*
https://www.britannica.com/science/latitude. 2020.

22. Mambra, Shamseer. *The Complete Story of the Exxon Valdez Oil Spill*
https://www.marineinsight.com/maritime-history/the-complete-story-of-the-exxon-valdez-oil-spill/.

23. International Maritime Organization. *Regulations for Carriage of AIS*
http://www.imo.org/en/OurWork/safety/navigation/pages/ais.aspx.

24. Smestad, B. B., Asbjørnslett, B. E. & Rødseth, Ø. J.
Expanding the Possibilities of AIS Data with Heuristics. en.
TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation **11,** 93 (2017).

25. Aarsnes, M.
A Feasibility Study of Assessing Bunkering Operations Through AIS Data. en, 197
(June 2018).

26. Wickham. Tidy Data. *Journal of statistical software,* 24 (2014).

27. Applanix. *Course, Heading, and Bearing*
https://www.applanix.com/news/blog-course-heading-bearing/. 2020.

28. Palm, R. & Driankov, D.
Fluid Mechanics for Path Planning and Obstacle Avoidance of Mobile Robots: en. in *Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics*
(SCITEPRESS - Science and and Technology Publications, Vienna, Austria, 2014), 231.

29. matplotlib. *Installing Basemap* https://matplotlib.org/basemap/users/installing.html.

30. Anaconda. *Installing Basemap with Anaconda*
https://anaconda.org/anaconda/basemap.

31. Anaconda. *Installing Pandas with Anaconda* https://anaconda.org/anaconda/pandas.

32. Pandas. *Getting Started with Pandas*
https://pandas.pydata.org/docs/getting_started/index.html.

33. Wessel, Paul & Smith Walter H.F.
A Global Self-Consistent, Hierarchical, High-Resolution Geography Database
GSHHG(https://www.soest.hawaii.edu/pwessel/gshhg/).

34. Strang, Gilbert. *Linear Algebra and Its Applications, 2nd Ed*
(Academic Press Inc, 1980).