

Håvard Fylling Haugen

# Investigating Operational Patterns for Three Different Modes of Operation in Shipping Using AIS Data

A data enrichment study for a more reliable operational status

Master's thesis in Marine Technology

Supervisor: Bjørn Egil Asbjørnslett

June 2019



Håvard Fylling Haugen

# Investigating Operational Patterns for Three Different Modes of Operation in Shipping Using AIS Data

A data enrichment study for a more reliable operational status

Master's thesis in Marine Technology  
Supervisor: Bjørn Egil Asbjørnslett  
June 2019

Norwegian University of Science and Technology  
Faculty of Engineering  
Department of Marine Technology

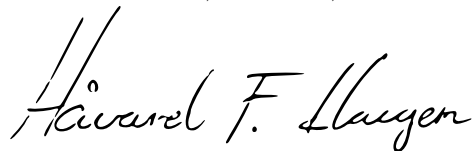




# Preface

This master's thesis represents the finalisation of my Master of Science (MSc) in Marine Technology with a specialisation in Marine System Design & Logistics at the Department of Marine Technology (IMT). The work has been carried out during the spring semester of 2019 at the Norwegian University of Science and Technology (NTNU) in Trondheim, and corresponds to 30 ECTs. This thesis encompasses a wide range of disciplines, and directed towards a general technical audience.

Trondheim, June 11, 2019

A handwritten signature in black ink, reading "Håvard F. Haugen". The signature is written in a cursive style with a large initial 'H'.

---

Håvard Fylling Haugen

# Acknowledgment

There are several people to whom I would like to express my sincere gratitude for making this thesis possible. Firstly I would like to thank Prof. Bjørn Egil Asbjørnslett at the Department of Marine Technology at NTNU, my supervisor, for competent guiding throughout this thesis. I would also like to thank The Norwegian Coastal Administration for supplying me with the AIS data. Further, I would like to thank MSc in Marine Technology, Bjørnar Brende Smestad, for inspiring consultations, and help with structuring the data. Lastly, my office mates for their open discussions and general advice.

H.F.H

# Summary

This thesis uses Automatic Identification System (AIS) data to analyse operational patterns for different modes of operation in shipping. Efficient ship operations have never been more relevant, and are important from both an economic as well as an environmental perspective. The aim of this thesis to understand real operational activity with the overall goal of increasing operational efficiency.

A comprehensive literature study is presented to investigate relevant AIS applications for the scope of the thesis. An introduction into the shipping industry is presented, where essential dynamics are captured, as well as operational characteristics for the different modes of operation. A thorough introduction to the AIS system and its reliability are provided. The AIS data used in this is provided by the Norwegian Coastal Authorities and is from the year of 2018. Port locations are obtained from Sea-web Ports. A comprehensive data preparation process is conducted, where the focus has been to prepare the AIS data for efficient operational analysis.

This thesis presents quantitatively that the included navigational status in the AIS messages is of limited quality, and that operational analyses could clearly benefit from a more reliable operational status. A substantial part of this thesis has been the development of a data enrichment process that establishes a vessel's true operational state. The presented method uses the vessel's speed and port location to determine the operational state, where it is distinguished whether the vessel is in port, waiting to enter a port, or underway sailing.

A case study is presented, where the new and more reliable operational statuses have been used to analyse operational patterns for three different modes of operation. Data for a representative selection of vessels were obtained, where container, bulk and LNG vessels were selected to represent respectively liner, tramp and industrial operation. Operational characteristics for the different modes have been investigated, where it was found support for most of the preconceived characteristics, especially with respect to loading conditions and port operations.

In conclusion, the extensive data preparation and enrichment process presented in this thesis has simplified the process for analysing operational patterns using AIS data. Furthermore, it was found support for that the mode operation has an impact in many aspects of a vessel's operation. Following the presented methodology could help ship operators improve their fleet's operational efficiency by having a better understanding of the current activity.

# Sammendrag

Denne masteravhandlingen bruker data fra Automatic Identification System (AIS) til å analysere operasjonsmønstre for ulike måter å operere i shipping. Effektiv drift av skip har aldri vært mer aktuelt, og er viktig sett fra både et økonomisk og miljømessig perspektiv. Denne oppgaven ønsker å forstå reelle skips operasjoner, med det overordnede målet om økt operasjonell effektivitet.

En omfattende litteraturstudie er presentert for å undersøke om AIS er brukt i litteraturen på relevante måter for denne oppgaven. En introduksjon til shipping bransjen er gitt, hvor de viktigste momentene og operasjonelle karakteristikk blir presentert. En grundig introduksjon til AIS- systemet, samt påliteligheten av systemet blir også gjennomgått. AIS dataen brukt i denne oppgaven stammer fra Kystverket, og havnelokasjoner er hentet fra Sea-web. En grundig prosess for å klargjøre dataen blir presentert, med hovedfokus på å klargjøre dataen for effektive operasjonelle analyser.

Denne oppgaven presenterer kvantitativt at navigasjonsstatusen inkludert i AIS-meldingene er av begrenset kvalitet, og følgende at operasjonelle analyser kan dra nytte av en mer pålitelig operasjonell status. En vesentlig del av denne oppgaven har vært utvikling av en prosedyre som etablerer et fartøys faktiske operasjonelle status. Den presenterte metoden bruker skipets fart og avstand til nærmeste havn for å bestemme denne statusen, hvor det skilles mellom om fartøyet er i havn, venter på å komme inn til en havn, eller i et seilas.

En case-studie er presentert, der de nye operasjonelle statusene har blitt brukt til å analysere operasjonsmønstre for de tre forskjellige måtene å operere på. Data for et representativt utvalg av fartøy ble etablert, hvor container, bulk- og LNG-skip ble valgt for å representere henholdsvis liner, tramp og industriell drift. Operasjonsmønstre for de forskjellige måtene å operere på har blitt undersøkt, hvor det ble funnet støtte for de fleste forutbestemte karakteristikk, spesielt med hensyn til lastekondisjon og havoperasjoner.

Som konklusjon har den omfattende data klargjøring og berikelse prosessen som blir presentert i denne oppgaven forenklet prosessen for å analysere operasjonsmønstre basert på AIS data. Det ble også funnet at operasjonsmetoden i shipping påvirker alle operasjonelle forhold for et skip. Følgende kan den presenterte metoden hjelpe skipsoperatører med å optimalisere drift ved å skaffe en bedre forståelse på nåværende tilstand.



# Contents

<b>Preface</b>	<b>i</b>
<b>Acknowledgment</b>	<b>ii</b>
<b>Summary</b>	<b>iii</b>
<b>Sammendrag</b>	<b>iv</b>
<b>List of Tables</b>	<b>viii</b>
<b>List of Figures</b>	<b>ix</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Background . . . . .	1
1.2 State-of-the-art . . . . .	2
1.3 Objectives . . . . .	3
1.4 Scope and Limitations . . . . .	4
1.5 Structure of the Report . . . . .	4
<b>2 Literature Review</b>	<b>5</b>
2.1 Introduction . . . . .	5
2.2 Relevant Articles . . . . .	5
<b>3 Shipping</b>	<b>9</b>
3.1 Introduction . . . . .	9
3.2 World Fleet . . . . .	9
3.3 Chartering . . . . .	10
3.4 Ports . . . . .	11
3.5 Modes of Transportation Shipping . . . . .	12
3.5.1 Liner Shipping . . . . .	12
3.5.2 Tramp Shipping . . . . .	12
3.5.3 Industrial Shipping . . . . .	13
3.6 Operational Characteristics . . . . .	13

<b>4</b>	<b>Data Foundation</b>	<b>15</b>
4.1	AIS Data . . . . .	15
4.1.1	Introduction . . . . .	15
4.1.2	Regulations . . . . .	15
4.1.3	Transmission . . . . .	16
4.1.4	Message Types . . . . .	17
4.1.5	Message Content . . . . .	17
4.1.6	S-AIS . . . . .	19
4.1.7	AIS Data Quality . . . . .	19
4.2	Vessel Database . . . . .	20
4.3	Port Locations . . . . .	21
<b>5</b>	<b>Method</b>	<b>22</b>
5.1	Data Preparation . . . . .	22
5.1.1	Data Decoding . . . . .	23
5.1.2	Data Filtering . . . . .	24
5.1.3	Data Structuring . . . . .	25
5.1.4	Data Validation . . . . .	27
5.2	Data Enrichment . . . . .	29
5.2.1	Port Locations . . . . .	29
5.2.2	Operational Status . . . . .	30
5.2.3	Leg Numbering . . . . .	33
5.3	Operational Analytics . . . . .	34
5.3.1	Area Segmentation . . . . .	34
5.3.2	Distances . . . . .	36
5.3.3	Loading Condition . . . . .	37
<b>6</b>	<b>Case Study</b>	<b>38</b>
6.1	Data Preparation . . . . .	38
6.1.1	Obtaining Data . . . . .	38
6.1.2	Data Structuring . . . . .	39
6.1.3	Data Validation . . . . .	40
6.2	Data Enrichment . . . . .	41
6.2.1	Port Locations . . . . .	41
6.2.2	Labelling Status . . . . .	43
6.2.3	Leg Numbering . . . . .	46
6.3	Operational Patterns . . . . .	47
6.3.1	Trade Flows . . . . .	47
6.3.2	Loading Condition . . . . .	48
6.3.3	Container Redistribution . . . . .	51
6.3.4	Port Times . . . . .	53
6.3.5	Waiting Times . . . . .	53
<b>7</b>	<b>Discussion</b>	<b>55</b>

<i>CONTENTS</i>	vii
7.1 Data Preparation . . . . .	55
7.2 Data Enrichment . . . . .	56
7.3 Operational Patterns . . . . .	58
<b>8 Conclusion</b>	<b>61</b>
8.1 Concluding Remarks . . . . .	61
8.2 Recommendations for Further Work . . . . .	62
<b>Bibliography</b>	<b>63</b>
<b>Appendices</b>	<b>67</b>
<b>A AIS Message Contents</b>	<b>I</b>
<b>B Code</b>	<b>IV</b>
B.1 database.py . . . . .	IV
B.2 main.py . . . . .	IX
B.3 db.py . . . . .	XIX
B.4 readCSV.py . . . . .	XXII
B.5 ais.py . . . . .	XXIV
B.6 machineLearning.py . . . . .	XXX
B.7 ais_Plotting.py . . . . .	XXXI
B.8 locationCheck.py . . . . .	XXXIX

# List of Tables

3.1	Average time in port in 2017 . . . . .	12
4.1	Reporting interval given operational status . . . . .	16
4.2	Most common AIS message types . . . . .	17
4.3	Representation of first digit in AIS ship type . . . . .	18
4.4	Ship Database Excerpt . . . . .	20
4.5	Port Database . . . . .	21
5.1	Structure of Raw S-AIS messages . . . . .	23
6.1	Original Database . . . . .	39
6.2	Database Size After Validation . . . . .	41
6.3	Number of Ports: Sea-web Ports vs. Clustering . . . . .	42
6.4	Number of Clustered Port Not Found in SeaWeb . . . . .	42
6.5	Accuracy of Port Locations: Sea-web vs. Clustering . . . . .	43
6.6	Messages with wrong navigational status . . . . .	44
6.7	Statistical Representation of Waiting Vessel Distance . . . . .	45
6.8	Number of changed Operational Statuses . . . . .	45
6.9	Sailing Distances: Laden vs Ballast . . . . .	50
6.10	Containerized trade on major Eas-West trade routes for 2018 (UNCTAD 2018)	51
6.11	Time in Port . . . . .	53
6.12	Waiting Times . . . . .	54
A.1	Information included in static AIS messages . . . . .	I
A.2	Information included in dynamic AIS messages . . . . .	II
A.3	Information included in voyage related AIS messages . . . . .	III
A.4	Information included in safety related AIS messages . . . . .	III

# List of Figures

3.1	World's Fleet Top 10 Ship Types . . . . .	10
3.2	World's Top 10 Ports . . . . .	11
5.1	Data Preparation Methodology . . . . .	23
5.2	Data Structuring Methodology . . . . .	26
5.3	Indexing Efficiency . . . . .	27
5.4	Visualisation of the DBSCAN algorithm . . . . .	30
5.5	Visualisation of Erroneous Status Reports . . . . .	31
5.6	Procedure for determining operational status . . . . .	32
5.7	The World's Oceans . . . . .	35
5.8	Ray Casting Visualisation . . . . .	36
6.1	Yearly Message Count for each Vessel . . . . .	40
6.2	Operational Status; Original vs. New . . . . .	46
6.3	Density Plot, Container Vessels . . . . .	47
6.4	Density Plot, Bulk Vessels . . . . .	48
6.5	Density Plot, LNG Vessels . . . . .	48
6.6	Draught Ratio Distribution . . . . .	49
6.7	Sailing Speed: Ballast vs Laden . . . . .	50
6.8	Trans-Atlantic Trade . . . . .	52
6.9	Asia - Europ Trade . . . . .	52

# Nomenclature

## Acronyms

AIS Automatic Identification System

COG Course over Ground

ETA Estimated Time of Arrival

GPS Global Positioning System

IMO International Maritime Organization

ITU International Telecommunication Union

LNG Liquefied natural gas

MARPOL International Convention for the Prevention of Pollution from Ships

MMSI Maritime Mobile Service Identity

MOU Mobile Offshore Units

S-AIS Satellite - AIS

SOG Speed Over Ground

SOLAS International Convention for the Safety of Life At Sea

SOTDMA Self Organizing TDMA

SQL Structured Query Language

TDMA Time Division Multiple Acces

TEU Twenty-foot Equivalent

VHF Very High Frequency

VLCC Very Large Crude Carrier

# Chapter 1

## Introduction

### 1.1 Background

International trade has the later years had a significant increase in volume. The main contributors for this increase are the added volumes from the developing countries, which as a group almost have doubled their trade since 2009 (UNCTAD 2018). Maritime transportation is the dominating mode of transportation, especially in intercontinental trade. According to ITF (2017), maritime transportation accounts for around 80% in volume, and over 70% in value of the global world trade.

Christiansen et al. (2004) presents a way of categorizing maritime transportation by distinguishing between three modes of operations, namely liner, tramp, and industrial shipping. Even if these modes are operated with a different set of perspectives and goals, all stakeholders benefit from efficient operations. Today it is not only from an economic perspective efficiency is essential. Maritime transportation is indeed the most energy efficient mode of transportation, but with the increasing need to lower the worlds greenhouse gas emissions, maritime regulators have introduced regulations that aim to reduce emissions from shipping. IMO states in MARPOL Annex VI, Chapter 4, that ships must improve energy efficiency with 10% by 2020, 20% by 2025, and 30% by 2030 (IMO 2018). To achieve these goals, we are not only in need of technological innovations, but also improve overall operational efficiency.

The current digitization megatrend has forced companies in all line-of-business to adapt. Historically the shipping industry has been deeply conservative, and it's only in the later years the business has seen potential benefits of bringing the industry into the digital age. Since 2004 IMO have required that most vessels should have installed an Automatic Identification System (AIS). As a result, over 165 000 ships currently are transmitting AIS signals, making it one of the most successful maritime technology deployments of all time (exactEarth 2015).

AIS was originally developed as a ship to ship anti collision system, but in the later years it also has been used as a basis for tracking and analysing ship traffic. With the launch of AIS

receiving satellites it's now possible to track most vessels more or less in real time, and with improving technology the amount of available data is increasing, making it important for the players in this industry to utilize the possibilities in this data to keep staying competitive.

Efficient ship operations have never been more relevant and are important both from an economic, and an environmental perspective. An essential part of achieving this efficiency is the possibility to analyse current operations and operational patterns. With the increasing amount of data available from AIS, it is now possible to carry out quantitative analyses on ship operations. The operational patterns and trends could provide several new angles for players in the maritime industry. Shipowners and shipyards could use a better understanding of real ship operations as decision support when designing and building new ships. Ship operators could with updated operational patterns, further optimize operations. It could also be useful from an environmental perspective, where access to real operational patterns could help deciding the best environmental measures for specific vessels.

## 1.2 State-of-the-art

AIS is a relatively new technology, and following the amount of literature with relevant applications of AIS signals is limited. However, after the launch of the first AIS receiving satellite, many academics saw potential in using this data with new perspectives. The background for the deployment of the AIS system was to contribute to collision avoidance. Naturally, maritime safety is a dominating part of AIS application in literature. Tu et al. (2018) investigates how AIS could contribute to route estimation, collision, and path planning. Another vital aspect of maritime safety is maritime spatial planning. Fiorini et al. (2016) uses AIS data to visualise ship routes and densities, Metcalfe et al. (2018) maps spatial distribution of different vessels types, and Tixerant et al. (2018) uses AIS data to analyse interaction between various marine activities, and how these can contribute to avoiding potential conflicts.

With the increasing focus from the industry, as well as stricter regulations, we have seen an increase in number of papers that uses AIS data with an environmental perspective. Schill & Browning (2015) explores the current applications of AIS data in environmental protection. Goldsworthy & Goldsworthy (2015) develops a model that use AIS data for calculation of ship engine exhaust emissions. Chi et al. (2016) presents a methodology for real-time monitoring of a vessels energy efficiency and emissions.

Another frequent topic in the literature related to AIS is traffic patterns and shipping networks. Spiliopoulos et al. (2018) investigates how global trade patterns could be extracted from AIS data, Wu et al. (2017) finds global vessel and traffic density, while Jia et al. (2017) uses AIS to obtain geographical, directional and total transport volume. Adland et al. (2017) investigates the reliability of trade volumes derived from AIS data

Operations and economics studies that use AIS data are also a substantial part of the literature. Smestad et al. (2017) use heuristics to determine ship type without the need for



an external database. Millefiori et al. (2016) estimates a seaports operational region. Jafarzadeh & Schjøberg (2018) uses AIS data to obtain ships operational profiles to identify ship types that could benefit from hybrid propulsion. Another side of operations and economics is sailing speed. Adland & Jia (2018) investigate which parameters influences sailing speed, while Assmann et al. (2015) investigates if speed is correlated with high freight rates and low bunker prices.

### **What Remains to be Done?**

From the literature study, it became clear that AIS has grown to become a popular subject in research. Within the scope of operational patterns, previous research has mainly focused on one part of a vessel's operation, where either the voyages or the port operations have been investigated. Most papers also only include one shipping segment in their scope.

A vital part when carrying out operational analytics using AIS data is the vessels operational status. The AIS message includes a navigational status report, but from the literature study, it became clear that this status is of limited reliability, however, few papers addresses this.

Thus, the contribution of this thesis to the existing literature is twofold: First, a methodology for establishing new and more reliable operational statuses for each message is developed. Second, this new operational status is used to compare operational patterns for different modes of operation in shipping, where the entire voyage is included.

## **1.3 Objectives**

The main objective of this thesis is to establish a procedure for investigating operational patterns, using AIS data as a basis for quantitative analyses, and examine the differences for three different modes of operation.

To address this primary objective, some sub-objectives need to be addressed. AIS data must be obtained and structured for efficient analysis, and its reliability must be evaluated. A method for determining a ship's actual operational status must be established and evaluated. Methods for analysing operational patterns must be explored and implemented. The methods will be tested on a representative selection of ships from the three modes of operation, and expected operational patterns and characteristics will be investigated.

Thus, the contribution of this thesis to the existing literature is twofold: First, a methodology for establishing new and more reliable operational statuses for each message is developed. Second, this new operational status is used to compare operational patterns for different modes of operation in shipping, where the entire voyage is included.

## 1.4 Scope and Limitations

The main limitation is related to the available data. The work uses S-AIS data supplied by the Norwegian Coastal Authorities, where the only available data available was from the year of 2018, and information about single vessels could not be made public. However, the latter did not affect the objective of this thesis.

The presented work also uses port location to determine a vessel's operational status. The port database available was Sea-web Ports, where the accuracy of the port location are found to be somewhat limited.

The scope of the operational patterns analyses has been to explore and compare expected operational characteristics for the different modes of operation.

## 1.5 Structure of the Report

The remainder of this thesis is organised as follows:

Chapter 2 intend to highlight the papers deemed most relevant for the scope of this thesis, where the methodologies used in the rest of the thesis is presented.

In chapter 3, a brief introduction to maritime transportation is given, where the focus has been to capture the most essential dynamics in the industry and highlight the operational characteristics for different modes of operation.

The data foundation used in the thesis is presented in chapter 4, where the fundamentals of AIS data and its content is presented, as well as the vessel and port databases accesed.

Chapter 5 presents the methodologies used in the case study. First, the complete procedure for preparing the obtained data is introduced. Second, a data enrichment process to establish a more reliable operational status presented. Finally, the basic methodologies for analysing operational patterns are presented.

A case study is conducted in Chapter 6, where the methodologies are tested on a representative selection of ships for the different modes of operation. The method for establishing a new operational status is developed and validated before the operational patterns are compared. The methods and results are discussed throughout the chapter.

Chapter 7 gives a higher level discussion of the work conducted and its results, before a conclusion and recommendations for further work is presented in chapter 8.

# Chapter 2

## Literature Review

Many articles regarding the application of AIS data have been studied. This chapter presents the ones deemed most relevant to the scope of this thesis. Numerous articles regarding shipping operations have also been under review, where the essential takeaways are presented throughout the thesis and in Chapter 3.

### 2.1 Introduction

The papers studied when writing this thesis have created a foundation for getting a deeper understanding of the AIS system, and how this data has been utilized in research until today. As AIS is a relatively new technology, there are relatively few studies using this data. However, the later years there has been an increasing interest from the academic research community in the possible application of this data, especially after the launch of AIS receiving satellites. The general methodology for the literature search has been to explore and review articles with a bit wider the scope than this thesis' actual objective. If the articles either have cited or been cited by other articles, these have also been reviewed. This procedure should ensure that both the source, as well as improved applications for all methodologies, is found.

### 2.2 Relevant Articles

AIS was initially developed to prevent collisions at sea and thereby increasing the safety of maritime traffic. Naturally, safety is also a hot topic in the AIS literature. Tu et al. (2018) presents a comprehensive study on how AIS data could be used for intelligent maritime navigation. The study explores the opportunities this data could have within safety, and present potential methodologies within each of these opportunities. Safe maritime navigation is categorised into three different aspects, namely route estimation, collision prediction and path

planning. Several methods for anomaly detection is introduced, and the possible application of these to find ships with anomalous characteristics. Speed anomaly would be most relevant for the scope of this thesis. They present a normalcy box for defining normal operational speed in different parts of a port region. A comprehensive survey of the available data sources for AIS data is presented and evaluated, where they find that the commercial databases are more extensive, especially when it comes to static messages.

Today's environmental problem and the increasing focus on reducing greenhouse gas emissions have inspired several studies where AIS data is used with an environmental perspective. Schill & Browning (2015) investigates how AIS, and especially S-AIS, can be used to address some of the environmental challenges of modern shipping. They suggest that nearly half of the pollution at sea is caused by, either ship accidents, or deliberate discharge of crude oil and other refined products. The study finds that that AIS could be used to identify these polluters by monitoring if ships are deviating from their predefined route. The reliability of S-AIS data is extensively discussed, especially from a receiving perspective. The two detection methodologies used by AIS receiving satellites are presented, respectively On-board satellite processing (OBP) and Spectrum de-collision processing (SDP). By investigation of data from the two different methods, they find that SDP is a superior method, especially in areas of high density. Another critical aspect of the environmental studies is a ships emission.

Goldsworthy & Goldsworthy (2015) develops a model for calculation of ship engine exhaust emissions in ports and extensive coastal waters. The model uses the engines loading factor for prediction of emission. The loading factor is found through the speed to design speed factor, and calculated by using AIS data and access to a ship's design specification. A method for determination of operation modes, based on the distance to shore and speed, is also presented. Smith et al. (2015) give updated figures on shipping emissions from 2007 to 2012 in The Third Greenhouse Gas Study. This is the first study from IMO that utilize AIS data to give better estimation off shipping emissions. The study include a presentation of potential source of errors included in the messages. When calculation emissions the article suggest that vessels moving with speed less than 3 knots area assumed to be at anchorage, and all messages below 3 knots are classified as anchor/port phase to estimate emission in these states.

With the launch of AIS receiving satellites, researches could analyse global traffic, following this is a large part of the AIS literature. Spiliopoulos et al. (2018) presents a four-step approach to extract global trade patterns from big data. The method extracts trade routes from raw AIS data, but the methodology is built on distributed processing (parallel processing) and could following be hard to set up. When pre-processing the data, it is suggested that message type 5 is prone to error and inconsistencies as it is manually inputted by the ship's crew. Wu et al. (2017) suggest there are mainly two different methods for mapping global vessel density and traffic density from AIS data, respectively grid-based and vector-based. They suggest that the most used method is the grid-based, but that this method is most suitable for smaller areas due to high computational cost. Their findings also indicate that the messages include many erroneous MMSI numbers. Several vessels share the same MMSI, and other messages had obviously the wrong MMSI such as 111111111 and 123456789.

Global ship traffic and movements could also be analysed with an economic perspective. Jia et al. (2019) present a methodology for estimating a vessels payload using AIS data only. This information has been difficult to obtain until now, as this not usual is public information. In the study, they use the draught parameter from AIS data, as well as a ships design specification as a basis for estimating the payload of bulk ships. Adland et al. (2017) investigates the reliability of trade volumes derived from official customs data. By conduction a case study on crude oil export, they find good alignment with estimated and official customs data. However, they suggest that a vessel's draught not necessarily is a perfect indicator of the payload.

Another part of ship traffic is a vessel's operational concerns, such as vessel speed. Adland & Jia (2016) uses AIS data for empirical testing of different determinants of vessel speed. Regression models based on technical, operational and macroeconomic variables are established. The results suggest that operational variables play a vital role in short-term vessel speed, while macroeconomic variables only have a marginal impact. To distinguish if a leg is in ballast or laden, they utilise the draught ratio, which is the ratio between current draught from an AIS message and the vessels design draught. For a VLCC a leg with draught ratio between 25% and 65% is characterised as ballast, and legs with draught ratios between 80% and 100% is characterised as a laden leg.

Assmann et al. (2015) investigate the claim made by both academics and by the industry that vessel speed is correlated with high freight rates and low bunker prices. By using AIS data from VLCCs, they find some support for the theory, however much lower than expected. Furtherer, they observe that speed optimizing is more common on backhaul trips than on laden trips. The result suggests that there are potential gains from more adaptations of slow steaming. Adland & Jia (2018) uses AIS data to investigate dynamic speed choice in bulk shipping. They find that owners do not appear to adjust vessel speed based on market condition, but vessel-specific variables and operational factor show some explanatory power. The result supports the previous studies and finds that macroeconomic variables have less impact on sailing speed than maritime economics should suggest

There are several more important aspect to a vessels operation than speed. Jafarzadeh & Schjølborg (2018) uses AUS to investigate which ship types could benefit from having hybrid propulsion. They find that offshore and passenger ships show the most dynamic operational profile, and following operates with significant time with engine loading factor away from diesel design specification. The operational profile presented is mainly the engine loading factor, which is derived from speed data in AIS signals. Millefiori et al. (2016) uses AIS data to estimate a seaports operations region. They suggest that a seaport's operational region not are staying static over time, but rather evolve as marked changes. The region is estimated by using an unsupervised machine learning methodology that is implemented on AIS data to define the extended operational area. Gao et al. (2016) uses AIS data to analyse the real operational activity of container ships. The study focuses on an entire ship voyage, including port operations, with the goal of improving navigation efficiency. The results can be used to improve the operational efficiency of container ships. Their methodology suggests that the navigational status of a vessel in an AIS message is manually inputted, and following

prone to error. They present a method for determining navigational status given a ship's speed and position. By using distance to berth, in addition to speed, the presented method distinguishes between three different navigational states, namely sailing, waiting or cargo handling.

Smestad et al. (2017) develops heuristics for determining ship type based on data included in AIS signals alone. By comparing the results from the heuristics with data from Clarksons Ship Register, the accuracy of the heuristic could be evaluated. Several ship types were identified with high precision, where e.g. Panamax bulk carriers could be determined with 98% accuracy. The methodology shows how parameters included in the AIS signal, such as speed, draught and length, could be used for estimating ship type. Rhodes et al. (2005) investigates how learning algorithms can be used in maritime situation monitoring. They present a two-step method for creating normalcy boxes. The normalcy boxes can be used to detect anomalies, which is an essential objective in maritime monitoring. Both an unsupervised clustering algorithm, and a supervised algorithm, are used for the creation of these normalcy boxes. Two case studies are presented, where one is the generation of a normalcy model in the New York Harbor area, using historical AIS data as input. The normalcy model in the case study presents normal operations speed for different area and operation in port.

# Chapter 3

## Shipping

This chapter gives a brief introduction to maritime transportation. Only the most essential part of shipping is presented as shipping is such a complicated business. The goal of the chapter is to capture the most important dynamics from the industry, and establishing a background for the expected operational characteristics.

### 3.1 Introduction

International trade is heavily dependent on maritime transportation and is by far the dominating method of transporting large quantities of goods over long distances. Maritime transportation accounts for around 80% in volume and over 70% in value of global world trade (ITF 2017). Shipping is often categorised into different segments. The most common is to categorise by either sailing distance or mode of operation. Short-sea shipping is the transportation of cargo and passengers mainly along the coast or shorter voyages, while deep-sea shipping is the transportation on intercontinental routes or ocean crossings. When categorising on the mode of operation, it is common to distinguish between three modes, namely liner, tramp, and industrial shipping (Fagerholt 2018).

### 3.2 World Fleet

The world merchant fleet consisted of 94 141 vessels on January 1, 2018 (UNCTAD 2018). The composition of the world's top 10 ship types are presented in Figure 3.1. The majority of the world's cargo fleet can be categorized into three main categories, respectively, bulk, container, and tanker. From the figure, we can observe that these types make up a large part both concerning the number of ships and capacity. There is however one category that stands out with a high number of vessels. This category is the general cargo ship, and it includes often smaller ships mainly involved in short-sea shipping. This vessel type often

carries various kinds of cargo and following is hard to categorise. It's worth noting that even this is the type with the highest number of vessels, it is one of the smaller concerning DWT.

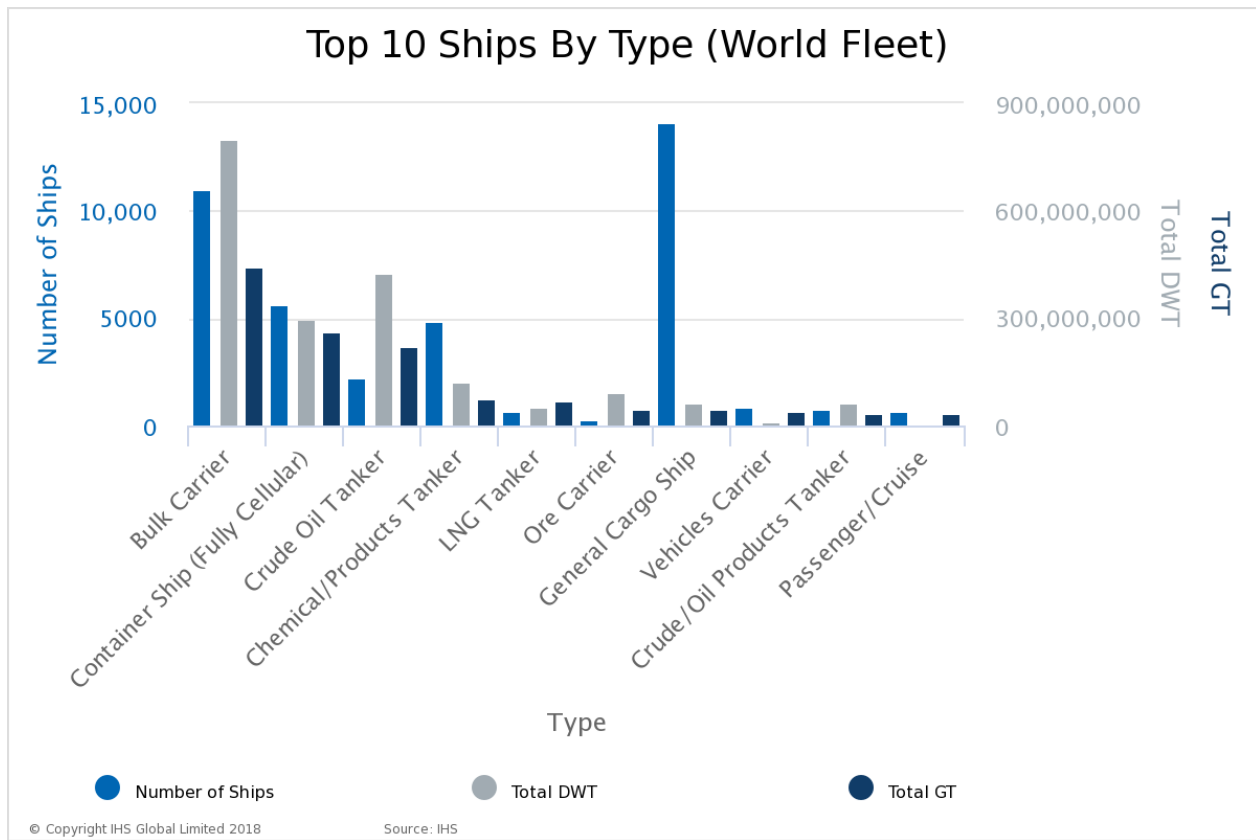


Figure 3.1: World's Fleet Top 10 Ship Types (IHS 2019b)

### 3.3 Chartering

Shipping today is made out of four closely related markets, all trading with different commodities. Sea transportation is traded in the freight market. Second-hand ships are traded in the sale and purchase market, new ships in the newbuilding market, and scraping of ships occur in the demolition market (Stopford 2009). The freight market can be defined as the place where the buyers and sellers of shipping services come together to strike a deal (Grammenos 2010). Shipowners are connected to a charterer through a broker. The contract between the shipowner and charterer are called a charter or charterparty.

The charter market can be divided into four different sectors, each with different characteristics. The voyage charter is when a charterer hires a ship to transport cargo from one port to another. Most of the operational responsibility lies with the shipowner. The contract of affreightment is when the shipowner agrees to carry a series of cargoes at an agreed rate, or in other words somewhat like a series of voyage charters. Ships are often not named in the



contract, so owners with a large fleet has an advantage (Grammenos 2010). A time charter is when a charterer is given operational control and responsibility of a vessel for a specific time. The shipowner is only responsible for ownership and management of the vessel (Stopford 2009). The bare-boat charter is when the shipowner operates more or less like an investor, and the charterer has full operational control and responsibility of the vessel.

### 3.4 Ports

Ports are an important player in global trade and logistics. In 2017 752.2 million TEUs were handled by container ports, where Asia is a dominating market, making up two-thirds of this volume UNCTAD (2018). However, in the later years, it's the bulk handling terminals that have had the most significant gain in throughput. The worlds ten largest port, based on million tons throughput, are presented in figure 3.2.

Rank	Port	Cargo throughput		Percentage change
		2016	2017	2017-2016
1	Ningbo-Zhoushan	918	1 007	9,7
2	Shanghai	700	706	0,8
3	Singapore	593	626	5,5
4	Suzhou	574	608	5,9
5	Guangzhou	522	566	8,5
6	Tangshan	516	565	9,6
7	Qingdao	501	508	1,4
8	Port Hedland	485	505	4,3
9	Tianjin	549	503	-8,4
10	Rotterdam	461	467	1,3

Figure 3.2: World's Top 10 Ports (UNCTAD 2018)

Ports today are subject to heightened competition, and port and terminal performance are now more critical than ever. This is especially the case in the container industry, where the market now is dominated by large shipping alliances. Following these alliances, shipping companies have invested in larger ships than ever before. We can now observe new dynamics between port and shipping companies, where ports are competing for fewer services by larger vessels, operated by large alliances. These new dynamics have, according to UNCTAD (2018), given shipping companies higher bargain power against ports. From the ports' perspective, it's now, more than ever, crucial to adapt and have efficient operations, to win port calls.

This increased focus on port performance could be the reason we observe that the average port time for all ships has improved in the last years. Where in 2017 the average time spent in port was 31.2 hours, down from 33.6 hours in 2016 UNCTAD (2018). The average port times for different shipping segments is presented in Table 3.1.

Table 3.1: Average time in port for different segments in 2017 (UNCTAD 2018)

Vessel Type	Days in port	Total arrivals
Container ships	0.92	447 626
Tankers	1.30	301 713
Gas carriers	1.10	64 603
Bulk carriers	2.68	236 407
Dry cargo and passengers ships	1.02	3 995 242
Total	1.31	5 054 591

## 3.5 Modes of Transportation Shipping

There is mainly two common way of categorizing the organization of the shipping market. Among others, Stopford (2009) and Branch (2014) organize the market into three main types of goods transported, namely Bulk, Specialized, and General Cargo. Where others, such as Fagerholt (2018), distinguish between three modes of operation: liner, tramp, and industrial. This thesis uses the latter categorization.

### 3.5.1 Liner Shipping

Liner shipping is when ships follow a fixed schedule, and the ships operate between specified ports. The schedule and pricing are advertised well in advance, in many ways similar to a bus line. Liner companies either own their ships, charter them, or a mix of these. The cargo is often within final and semi-final product, and also referred to as general cargo. The most important types of ships that operate in liner are container ships and ro-ro ships. Cruise ships also follow this model of operation, but not often referred to as liner operation (IMO 2016).

### 3.5.2 Tramp Shipping

In tramp shipping vessels follow the available cargo. Tramp ship are often engaged in contracts of affreightments, with optional spot ones (IMO 2016). The freight rate is decided by supply and demand for the specific shipping service. The cargo is often raw materials, such as oil and coal, and is commonly referred to as bulk cargo. Bulk cargo splits into two main types, wet bulk, and dry bulk. Similar for both is that the ship only carries cargo from one or two shipping users. Compared to liner shipping, tramp shipping carries cargo with low unit value. As the cargo carried, in general, is of one type the loading and discharging are often confined to a few ports.

### 3.5.3 Industrial Shipping

Industrial operators are operators that both control the fleet used under transport, and own the cargo that is transported. The operator could either own their vessel or have chartered the vessel on a time charter. The operational goal is to minimize the cost related to the shipping of their goods. Christiansen et al. (2004) suggest that in the later year there has been a shift from Industrial Shipping to Tramp shipping, where companies are focusing more on their core business. However, we still find many companies operating their own ship where, e.g. some of the major companies within oil and gas either operates their own ships or have vessels on long term charters, to retain control of the distribution of their products.

## 3.6 Operational Characteristics

One of the objectives in this thesis is to compare operational patterns of the three different modes of operation. To compare the different modes, we need a representative selection of ships from each mode to establish a quantitative foundation for analyses. As suggested earlier, the container segment is an obvious segment to represent liner shipping. Similarly, bulk ships represent the tramp segment, while LNG vessels represent the industrial shipping segment. From maritime theory, we expect to see many both similarities and differences in operational characteristics for the three modes of operations.

We expect some general differences from the different segments. While Container ships have a predetermined selection of ports to visit, bulk ships do not have a fixed schedule and follow the available cargo around the globe. Following we expect to see more static trade pattern from container vessels, while for bulk vessels we expect more sporadic trade routes. LNG vessels are operated mostly as Industrial shipping, following these ships often act as shuttles between the companies supply ports and the customer.

As suggested by amongst others Adland & Jia (2018) the vessel's loading condition has an impact on its operation. The different segments have a different operational pattern for different loading conditions. A voyage with cargo onboard is often referred to as laden leg, while a leg without cargo is referred to as a leg in ballast. Due to the mode of operation, we expect both bulk and LNG vessels to have an equal number of laden and ballast legs. However, as bulk ships are trying to maximise their revenue, they try to minimise the sailing distance in ballast condition by finding new cargo to transport close to the destination of the last shipment (Christiansen et al. 2004). As LNG operates as industrial shipping, the operation is based on minimising the cost related to the transport of the company's cargo. Following it is not so easy for an LNG ship to reduce the distance of the voyage when in ballast, as the vessels often must return to the same port. Assmann et al. (2015) finds that speed optimisation of a vessel is more common on backhaul trips than on laden trips. We following expect to see a larger variance in sailing speed for voyages is ballast. Especially for bulk ships where the idle time is minimised, and the next cargo is not available at the time of previous delivery (IMO 2016).

The later year we have witnessed consolidation through mergers of shipping companies and organisation in global shipping alliances (UNCTAD 2018). This results in that the need for ballast legs without any paying cargo aboard is not a concern for liner shipping (Grammenos 2010). However, due to an imbalance in trade container vessels often have a high load factor for revenue-generating containers in one direction, and a lower load factor in the opposite direction, due to container redistribution. This could have an impact in the vessels draught if the number of empty containers is significant, but should not have an effect on the vessels sailing speed.

Today there are three global shipping alliances dominating liner trade today, and together they make up 93% of the global East-West trade (UNCTAD 2018). By forming these alliances, the liner companies have strengthened their bargain positions against ports, and following container vessels have increased priority. Gao et al. (2016) suggest however that container vessel are fined if arriving late to a port. Because of this, many liner operations tend to arrive early and wait at anchor outside a seaport due to potentially extended voyages due to weather. For container vessels, it is following expected that many ships have a short wait before entering a port.

For LNG vessels, that operates as industrial shipping, we expect the vessels could enter the port without any waiting mainly because of two factors. First, in industrial shipping the operators also control at least the supplying harbour, so potential periods of waiting can be minimized. Secondly, LNG is a specialised cargo, an is following in need of special port facilities. Compared to LNG and container vessels the bulk ships tend to have significant wait times before entering a port. However, some bulk operators have arrangements that give them priority in port and could following lower the wait time for some vessels. We also expect some vessels that are waiting for the cargo to arrive, and following spend more time in port.

The three different segments will use different time in ports. Table 3.1 present the average port time spent by different shipping segments. Stopford (2009) suggest that containerization had a significant impact on world trade and its efficiency. We expect container vessels to have efficient port operations, with a low variance of time spent in port. The table also suggests that an LNG carrier should spend a somewhat longer time in port than a container vessel, while bulk carriers spend significantly more time than the other two segments.

# Chapter 4

## Data Foundation

This chapter introduced the data foundation used in the rest of this thesis. A comprehensive introduction on the AIS system, its message content and reliability are discussed. The ship and port databases accessed are also presented.

### 4.1 AIS Data

#### 4.1.1 Introduction

Artificial Intelligence System, hereby AIS, is an automatic tracking system developed to prevent collisions at sea and thereby increase the safety for maritime traffic. The system is based on the maritime VHF radio. Transceivers are installed onboard ships, and these broadcast information to transceivers on other ship, AIS base stations and satellites (S-AIS). Vessels can use AIS to safer navigation by sending and receiving information about position and course to surrounding vessels. Although AIS is a relatively new innovation, the technology has advanced the later years, and many new applications for it have been found. Many marine authorities, including port authorities and coast guards, are now using AIS for better monitoring the current maritime situation.

#### 4.1.2 Regulations

The first regulation including AIS was the 2002 version of IMO's SOLAS conventions where it became mandatory for most vessels over 300 gross tonnage and passenger vessels on international voyages to be fitted with a Class A transceiver. With later amendments the current regulation states:

*All ships of 300 gross tonnage and upwards engaged on international voyages and cargo ships of 500 gross tonnage and upwards not engaged on international voyages and passenger ships*

*irrespective of size shall be fitted with an automatic identification system (AIS)* (IMO 2014).

In Norway, the Norwegian Maritime Directorate have introduced a somewhat stricter requirement for AIS than SOLAS. In addition to the requirements included in SOLAS there is also a demand for all European fishing vessels over 15 meters, and all MOU (Mobile offshore units), to be fitted with Class A transceiver (Norwegian Coastal Administration 2014).

### 4.1.3 Transmission

There are several types of devices with the purpose of sending and receiving AIS signals. Class A and Class B transceivers<sup>1</sup> are the most frequently used, where ships under the regulations stated earlier are required to be fitted with a class A transceiver. Class B transceivers are mainly used by vessels that voluntary have installed AIS and ships operating in countries with special regulations Ball (2012). Both of these devices rely on a SOTDMA system. Where TDMA is a communication system where signals not are transmitted continuously, but rather in specific time slots. This means that if a ship wants to broadcast information, this must happen inside one of these time slots.

SOTDMA was the solution for ensuring that communication happens in an orderly way. The transmitter first maps the current time slot, then announces of time which time slot it will use before it broadcasts information. Using two dedicated VHF channels, the AIS system is cable of sending 4500 of these time slots per minute. According to Eriksen et al. (2010) each of these slots can fit a 256-bit package, containing a fixed-length digital message.

Depending on the vessels operational status, the SOTDMA protocol will adjust the number of reporting time slots required. The different intervals is collected from IMO (2015) and presented in Table 4.1.

Table 4.1: Reporting interval given operational status

Vessel Operational Status	General reporting interval
Vessel at anchor	3 min
Vessel at 0-14 knots	10 sec
Vessel at 0-14 knots and changing course	3 1/3 sec
Vessel a 14-23 knots	6 sec
Vessel at 14-23 knots and changing course	2 sec
Vessel at >23 knots	2 sec
Vessel at >23 knots and changing course	2 sec

<sup>1</sup>Transceiver: A device that can both transmit and receive communications, in particular, a combined radio transmitter and receiver. <https://en.oxforddictionaries.com/definition/transceiver> (Accessed: 07.10.18)

### 4.1.4 Message Types

The different AIS transmissions have different purposes, and following also contains different information. The International Telecommunication Union (ITU) categories the different AIS transmissions into 27 different AIS message types (ITU 2014). The most frequently used types are type 1-5; these are presented in Table 4.2. The remaining message types mainly consist of messages related to the functionality of the system and special cases, where one example is message type 10, which requests current time and date.

Table 4.2: Most common AIS message types

ID	Name	Description
1	Position Report	Scheduled position report
2	Position Report	Assigned scheduled position report
3	Position Report	Special position report (response to interrogation)
4	Base station report	Position, UTC, date and current slot number of base station
5	Static and voyage related data	Scheduled static and voyage related vessel data report

All these types can be categorized into two main categories, namely static and dynamic messages. Dynamic messages transmit dynamic information, such as position and speed. This is automatically transmitted every 2-10 seconds, depending on the vessels operation (see table 4.1). Static messages transmit static and voyage related information. Some are programmed into the transmitting equipment at installation, e.g. IMO number, where the rest is provided by the crew, such as ETA and Destination.

### 4.1.5 Message Content

The different message types contain a different type of information. For the scope of this thesis, only message type 1 to 5 will be further used, both because these represent almost all AIS messages in the database, and because the information included in these messages is sufficient. The complete collection of information included in all message types are gathered from IMO (2015) and attached in Appendix A. The following is the critical information in the messages.

- **UNIX Time**

All AIS messages contain a UNIX time stamp. This is a system for describing time, where the number is equal to the number of elapsed seconds since 1 January 1970.

- **IMO**

The IMO number is a unique vessel identification number. This number is assigned to the ships hull, and is not changed during the vessels lifetime.

- **MMSI**

The Maritime Mobile Service Identity is a unique 9 digit number, that identify each

station that transmits on the VHF network. According to ITU (2015) the first digit states the type of transmitting device, where e.g. digit 2-7 is individual ships and 8 is from handheld VHF devices.

- **Coordinates**

Geographical coordinates, divided into latitude and longitude in degrees.

- **SOG**

Speed over ground (sog) is recorded from the ships GPS system, and is given in knots. It's important to keep in mind that a ship's resistance is mainly depended on the speed through water, not the SOG.

- **COG**

Course Over Ground (COG) is given in 1/10 degrees interval.

- **Navigational status**

The navigational status is to be manually inputted by the crew, where e.g. State "0" states the vessel is underway using engines, "1" is at anchor, and "5" is moored. See Appendix A for the complete list.

- **Ship Type**

The AIS ship type is a double digit number form spanning from 10 to 99, where the first digit states the type of vessel. For some vessels (Cargo and Tankers) the second digit also states the hazard level of the cargo, where the hazard level is spanning from 1 (major hazard) to 4 (Recognisable hazard) (MarineTraffic 2018). The first digit representation is presented in table 4.3.

Table 4.3: Representation of first digit in AIS ship type

First digit	Ship type
1	Reserved for future use
2	WIG (Wing In Ground)
3	Other vessels
4	High-speed carrier, or vessel <100 Gross Tonnes
5	Special craft
6	Passenger ships >100 Gross Tonnes
7	Cargo ships
8	Tankers
9	Other types of ships

There are also some vessel specific information such as current draught and breadth, as well as voyage related information such as Destination Port and Estimated Time of Arrival (ETA).

As stated earlier, message type 1,2 and 3 are dynamic messages, and type 5 is a static message. These two message type of messages must be analyzed in combination with each other, as dynamic messages do not contain vessel descriptive information. The only thing



connecting the two message types is the MMSI number, following this number will be used for connecting the operational conditions with the vessel-specific information.

#### 4.1.6 S-AIS

Skauen et al. (2013) states that a typical range for the coastal AIS network is about 40-50 nautical miles. This is mainly because of the world's curvature, and these signal only travels in a straight line. This meant that one was not able to track messages sent from outside this zone. Høyve et al. (2008) claims that AIS signals could reach as far as 10 000 km in the vertical direction, and by having AIS receivers on satellite one can overcome the range restriction found in surface bases AIS. In 2010 the first AIS satellite receiver was launched, opening the possibility to receive and track global AIS data.

By having space-based AIS receivers, the information received has increased drastically. One satellite can have global coverage after seven orbits, or about 11 hours Eriksen et al. (2010). This means that space-based AIS receivers face two additional challenges compared to the one on the surface. First is that because of the increased distance between transmitter and receiver, the signals are weaker. The second is that due to the increased range of the receiver, the possibility of interference problem is increasing. Especially areas of high traffic density could potentially cause issues according to Skauen et al. (2013).

The Norwegian coastal administration is currently possessing four satellites receiving AIS data. The first two was sent into orbit in respectively 2010 and 2014. Its information from these satellites that is mainly the source for the analysis in this thesis. In the summer of 2017 another two satellites was launched Norwegian Coastal Administration (2017).

#### 4.1.7 AIS Data Quality

Several factors could impact the quality of the AIS data. With the introduction of satellites receiving AIS signals, and the new possibilities this created, the quality of the AIS data became even more relevant. This section will focus primarily on the quality of S-AIS data.

As mentioned earlier, the AIS system was not designed for space-based receivers. Initially, the system was designed for receiving messages from within a limited range, with satellites receiving these signal, the range of messages available increased significantly. With this increased range, there is a possibility that two different messages are overlapping. This is especially the case when receiving from different SOTDMA cell, but same time slot (Høyve et al. 2008). This will especially be a problem in the area of high traffic, and with low orbiting rates, this could lead to gaps in the data.

It is not only interference and receiving problems that cause erroneous messages. As discussed earlier, it is mainly two different types of AIS messages, static and dynamic messages. While dynamic messages include parameters such as speed and location, the static message contains voyage and vessel related information. Smestad (2015) found some kind of inaccurate data

in several thousands of vessels. Where he found among other, vessels are longer than the world’s longest ship, and invalid IMO number. Leonhardsen (2017) find that the number of distinct MMSI is higher than the size of the world fleet. Wu et al. (2017) pointed out that it is not unusual that several vessels share MMSI number, and suggest that this is either because vessel change ownership or that it could be default MMSI for a specific model of AIS sender.

Both the static and dynamic messages contain some data to be manually inputted. Examples of this are draught, ETA and navigational status. Following human error is a prominent source of error in the ASI data. Some data is manually inputted during the installation of the equipment, while others, such as draught and navigational status, is updated by the vessel’s crew during the voyages. From the literature study in Chapter 2, it became clear that especially the navigational status is prone to error, as the crew tends to forget to update the vessel’s status, as suggested by among others Spiliopoulos et al. (2018).

## 4.2 Vessel Database

The AIS signal contains only limited information about the ship the message originates from. As Table 4.3 shows, it is only possible to distinguish between a limited selection of ship types with the ship classification included in the AIS message. Vessel-specific data is also limited, especially with regards to design specifications. There exist several providers of vessel specific information, where many of them are subscription based. In this thesis, Sea-web Ships is used to obtain vessel specific data. This is one of the more comprehensive databases, and contain detailed data on over 200 000 different ships. The database includes most design parameters for every ship. The online tool also allows filtering on ship type and extracting vessel lists to external filetypes, including CSV files. Table 4.4 present an excerpt of the ship database to visualise some of the available data, and how it is structured. The complete database includes several more parameters.

Table 4.4: Ship Database Excerpt

IMO/LR/IHS No.	...	Displacement	Draught	Flag	Length	MMSI	...
...							
9351608		68,719	12.621	Hong Kong	265.04	477815200	
9426805		85,394	13.5	Liberia	293.9	636017515	
9464704		71,025	12.62	Liberia	260.32	636018300	
9400576		0	12	China	294	412713000	
...							

From the table, we can observe that the information provided is of varying quality, where e.g. one of the ships are listed with a displacement of 0. The resolution of the data is also of varying quality. If we look at the draught, the information is given with different precision.

It is worth noting that this excerpt was done to highlight differences in the data quality and that the database is in general of high quality.

### 4.3 Port Locations

Ports are an essential part of the worlds shipping network, and following also a vital part of a ship's operations. There exist several methods for extracting port information and locations. In this thesis, IHS Markits's port and terminal guide is used. This is an online database that contains information about the world's ports, including location and facilities, and provides over 15 500 ports and terminals. The database includes the possibility to filter on port facilities and extract lists to external files. To visualise some of the available data, and its structure, an excerpt of the database is presented in Table 4.5. The except is for one port, with including terminals, the whole database also includes several more parameters about each port.

Table 4.5: Port Database

World Port Nr	Country	Latitude	Longitude	Name	...
PO1002	Albania	41°18 N	19°27 E	Durres, Albania	
PO1002	Albania	41°19.00 N	19°27.00 E	Durres Container Terminal...	
PO1002	Albania	41°19.00 N	19°27.00 E	Ferry Terminal, Albania...	
PO1002	Albania	41°19.00 N	19°28.00 E	Lindor Terminal,Albania ...	
PO1002	Albania	41°19.00 N	19°28.00 E	Western Terminal, Albania ...	
...					

From the table we can observe that the coordinates is in the format: *degrees* and *decimal* minutes. When looking at the precision of the data, the *minutes* seems to be given with four significant figures. Further investigations suggests this number is actually limited to two, as all of the ones with four have the last two figures equal to ".00". In other words, this means the precision of the port locations is limited to one whole minute, or approximately 1.8 km on the earth's surface.

# Chapter 5

## Method

This chapter presents the methodologies used in the later presented case study. First, a data preparation process for preparing AIS data for operation analytics is presented, where the focus has been on establishing the foundation for efficient operational analyses. Second, a data enrichment process is presented to establish a new and more reliable operational status. Finally, the methods used for analysing operational patterns are described. The limitations of the methods are discussed throughout this chapter.

### 5.1 Data Preparation

The data preparation process in this study prepares the available data for efficient analyses. Chapter 4 presents the data foundation used in this thesis. This chapter mainly describes the procedure for the preparation of AIS data from its raw original format to a database suitable for analyses. The general data preparation process for the AIS data is visualised in Figure 5.1. The port coordinates obtained from Sea-web Ports are also in need of decoding to be used in analyses with the AIS data.

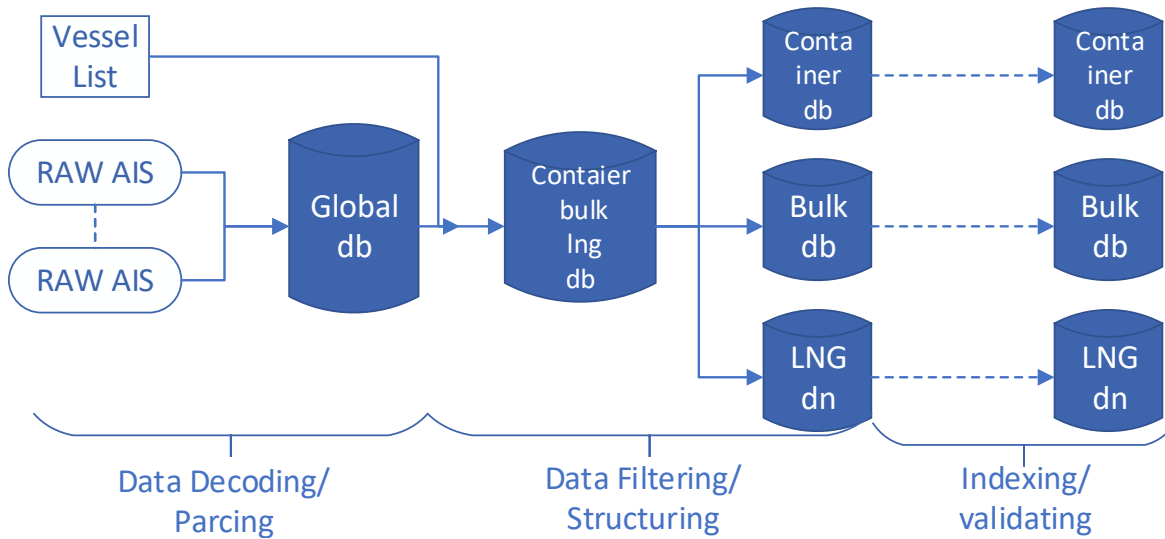


Figure 5.1: Data Preparation Methodology

### 5.1.1 Data Decoding

In computer science, encoding is the process of converting data into a specific sequence of characters for efficient transmission and storage<sup>1</sup>. Decoding data is the opposite, where this specific sequence is separated and converted into an understandable format more suitable for further analyses. Both AIS data and Port locations require decoding.

#### AIS

The data utilised in this study is granted by the Norwegian Coastal Authorities and contain data from the year of 2018. The data is structured as daily files that contain all AIS messages received that specific day. The structure of an arbitrary S-AIS message is presented in Table 5.1.

Table 5.1: Structure of Raw S-AIS messages (Leonhardsen 2017)

Arbitrary S-AIS Message
/s:ASM//Port=638//MMSI=c:1280622239*78/!BSVDM,1,1,,A,14cTbj0vAV16ctLelSOB>Aih0D01,0*54

The raw AIS messages are decoded by Bjørnar Brende Smestad, by using a Python script developed by Smestad (2015) that utilise an external AIS parser developed by Lane (2006). The

<sup>1</sup><https://searchnetworking.techtarget.com/definition/encoding-and-decoding>, (Accessed: 04.05.19)

decoded messages were then structured in an SQLite database with all S-AIS messages stored in a decoded format. The database holds every S-AIS message received by the Norwegian Coastal Administration for the year of 2018.

## Port Locations

Port locations also require decoding to be in a format suitable for further use. As mentioned in chapter 4.3, the port locations are given in the format *degrees* and *decimal minutes*. The AIS signal now contains coordinates in the form *decimal degrees*, and following it is beneficial to have the port coordinates in the same form. The general formula for converting into decimal degrees is:

$$\text{decimal degrees} = \text{degrees} + \frac{\text{minutes}}{60} \quad (5.1)$$

As presented in Chapter 4.3, an example of the format of the coordinates in the port database is:  $41^{\circ}18 N$ . When using computer programming, it is important to distinguish between different data types. The location is given as a combination of *digits* and *characters*, and following the datatype for the coordinates would be a string. It is important that the location is stored as a number, and following more accessible for analyses. By letting  $N$  and  $S$  correspond to respectively positive and negative numbers, and in a similar way  $E$  equal to positive and  $W$  to negative, the coordinates could be represented by only a number.

An algorithm is created for converting the coordinates into the form similar to the one found in the AIS signals. The algorithm works by splitting the string on  $^{\circ}$ , then converting the digits before and after into two numbers. Then depending on the last character it would return either the positive or negative of the sum of  $\text{degrees} + \text{decimal minutes}/60$ . The algorithm is found in Appendix B.4 under the name *decimalDegrees()*. This algorithm also removes duplicate positions so later analyses that utilise port location could run more efficiently.

### 5.1.2 Data Filtering

As presented in Chapter 4.1.5, the AIS message only contains limited information about the ship it originates from. The static AIS message (message type 5), contains a two-digit number specifying the ship type. Filtration on this ship type is not sufficient for the scope of this thesis, where it is necessary to filter on different shipping segment. As discussed in Chapter 2, Smestad et al. (2017) developed heuristics for establishing vessel-type based on AIS data alone. The result by only analysing speed, was that 92% of all container vessel sails with a maximum speed above 15.9 knots, while 92% of bulk carriers sails with a maximum speed less than 15 knots. By also including other parameters, such as length, breadth and draught more accurate heuristics was introduced, where for example, Panamax container vessel and Panamax Bulk vessels could be determined with respectively 90.3% and 98% accuracy.

Another way to find the vessel type is by accessing a ship database as presented in chapter 4.2. The MMSI number is the only parameter connecting the static and dynamic messages in AIS

signals, and following the only way of connecting these two. The Sea-web Ships database also include MMSI number in their database. Following, this parameter is a suitable parameter to utilize when filtering out ships of interest. Vessel list of different shipping segments can be downloaded from Sea-web, and used to filter out wanted AIS messages into a new database.

In SQLite, it is possible to attach a database to another. This is useful when establishing a new database with data from another database. One method for extracting only specific ship from the original database is to first insert the MMSI numbers into one table of the new database. Then attaching this database to the new, and inserting only the messages with MMSI number matching the one in the new database. The following SQLite is an example of the method.

```
INSERT INTO new.dyn
SELECT unixtime,cog,latitude,longitude,nav_status,sog,userid
FROM dyn WHERE (userid in (SELECT userid FROM new.ShipList))
```

Filtering could also be used to selecting the data that gives the best data foundation. In this thesis, it would be beneficial to have an as high messages resolution as possible. By counting the number of messages sent from each vessel, one could filter out the ship that gives the highest message count. The following SQLite query counts how many messages are sent by each ship and ordered in ascending order:

```
SELECT userid, count(*)
FROM dyn GROUP BY userid
ORDER BY count(*)
```

### 5.1.3 Data Structuring

The data could be structured in many different ways. For the analyses carried out in this thesis, it must be easy to distinguish from which shipping segment the different messages originates from. As presented in Figure 5.1, the data was split into separate databases, one for each shipping segment. The original dataset also distinguishes between the different message types. From Chapter 4.1.4, we know there are two main types: dynamic and static messages. Message type 1,2 and 3 are dynamic messages, while message type 5 are static. As the three dynamic messages include the same information, these could be combined into the same table. This will increase the resolution of the data, something that could be beneficial in this thesis. The Data structuring process is presented in Figure 5.2.

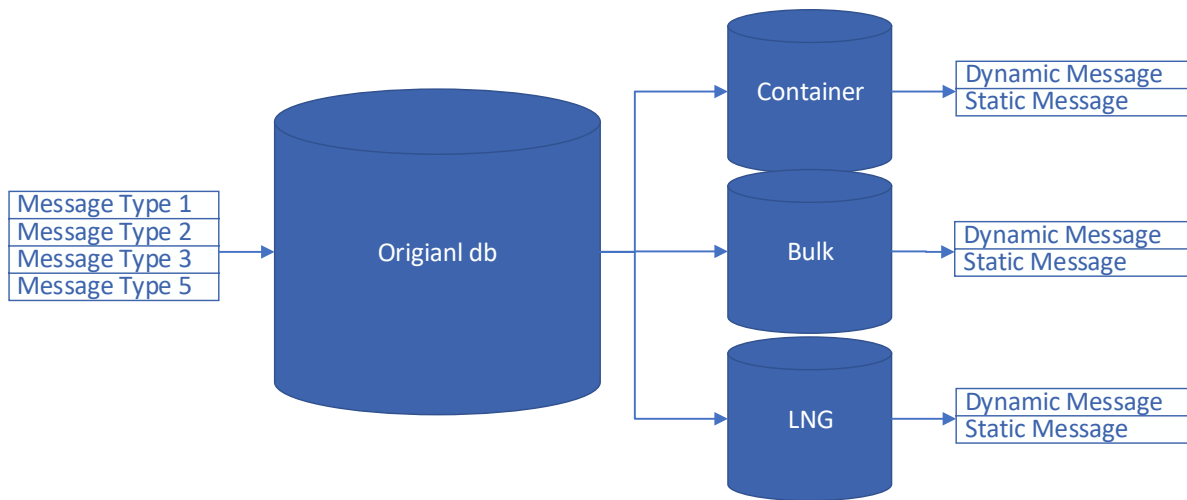


Figure 5.2: Data Structuring Methodology

## Indexing

When structuring the data, it is important to structure it in a way that efficient analysis could be carried out. As suggested by Smestad (2019), indexing can improve performance when searching for data in a database. When creating an index in SQLite, a new datatype is created. This data type contains pointers to the rows in the table. When querying an indexed variable, the algorithm does not need to search the whole table, only the specific location the pointers point to. In theory, the efficiency should improve with increasing cardinality (number of values subject to indexing). However, as the procedure creates a new datatype that stores these pointers, the size of the database also increases. To investigate the impact this could have when analysing AIS data; a simple test case was set up on a test dataset. The case was for the program to find the number of unique MMSI numbers in a dataset, before and after indexing. The algorithm was tested on increasing table size. Figure 5.3 presents the result.

From the figure, we can easily observe the high impact indexing could have, even for this test case where it just was a simple algorithm on a relatively small amount of data. The result also points to that this impact only will increase with increasing table size. An important thing to keep in mind is that for every variable indexed the database size increases. E.g. the last algorithm ran on about 6.3 million lines, and with indexing, the database increased from 258MB to 350MB. So it is crucial to only index the variables used in frequent searches to keep the database size within a reasonable size.



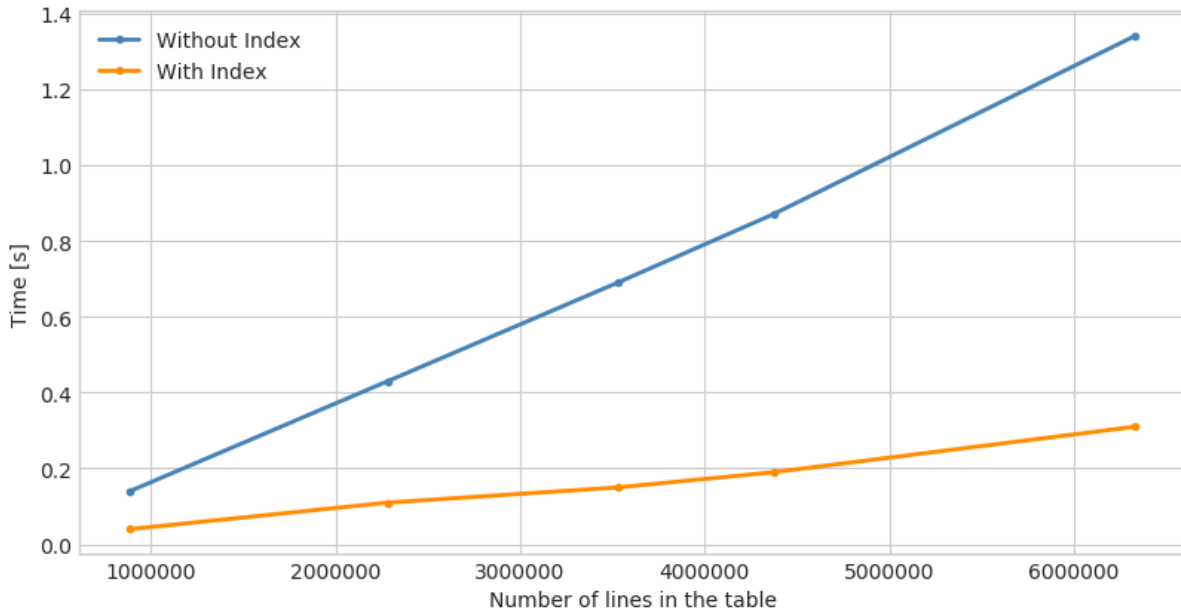


Figure 5.3: Indexing Efficiency

### 5.1.4 Data Validation

From the literature study presented in Chapter 2, it became clear that the AIS signals are prone to many types of errors, and requires validation before analyses could be carried out.

#### MMSI number

Wu et al. (2017) suggest that the MMSI number especially is prone to errors. Common erroneous reports include that the MMSI number does not follow the format as presented by ITU (2015), where the MMSI is a 9 digit number on the format  $mIDXXXXXX$ . The first three digits represent the maritime identification Digits (miD), and X is any figure from 0 to 9. Another possible source of incorrect MMSI number is that the equipment is still in factory setting mode, or have been restarted to factory default (Smith et al. 2015). Wu et al. (2017) find some MMSI number that is shared by more than 10 different vessels. These MMSI number often follows a valid format but could be number such as 11111111 and 123456789.

Another possible problem with filtering on MMSI number is that if the vessel change ownership during the time of the dataset, the MMSI number might change. Message type 5 includes both the MMSI number and the IMO number. The IMO number follows the vessels' hull and is never reassigned to another ship (IMO 2019). One method of assuring that the vessels analysed are the one of interest throughout the analyses, is to check if the IMO number corresponding to each MMSI number change in the dataset. A simple algorithm is to check if the IMO number change from the first to the last static message. Following is a pseudo code for this algorithm:

```
for for every unique MMSI number do  
  find first IMO number for that MMSI number  
  find last IMO number for that MMSI number  
  if first not equal last then  
    remove vessel from database
```

## Speed

Among others, Tu et al. (2018) suggests that the included messages could have false speed reports. They find the included speed over ground parameter could experience a jump in value for single messages due to noise. These incorrect messages could quickly be addressed by removing messages with speed reports outside the expected range. It is reasonable to assume that vessel in the shipping segment subject to analyses does not have a sailing speed above 35 knots.

## Ship Positions

Smestad (2015) suggest that the reported coordinates from AIS messages also could be erroneous. He finds that some messages include reports with coordinate without the range possible for coordinates. In other words that the messages could have a given longitude outside the defined range of  $-180^\circ$  to  $180^\circ$ , and latitude outside the defined range of  $-90^\circ$  to  $90^\circ$ . By visual validation, we could also ensure that we have false location reports, so no vessels have a location report suggesting the vessel sails on land.

## 5.2 Data Enrichment

Data Enrichment is the process of adding value to and improve the quality of a data set<sup>2</sup>. This could both be done by improving the data quality itself, as well as accessing other data sources to improve quality and add value to the original data set. In this thesis, a data enrichment process will be carried out to improve the quality of the data, especially with respect to the operational status included in the AIS messages. The goal is to use this improved operational status in operational analyses.

### 5.2.1 Port Locations

An essential part of a ship operations is as earlier discussed the ports. The port location is essential to establish a ship's operational status. As presented in Chapter 2, Millefiori et al. (2016) suggest that a seaports region not remains constant over time, but evolve as the marked changes. They present a methodology for mapping seaports operational regions by using unsupervised clustering method on AIS data. When using a port database, it is hard to validate that this contains updated information. Another problem could be that it is hard to filter out wanted ports, and following end up with a more extensive port database than needed. With inspiration in the method presented by Millefiori et al. (2016), we see machine learning could be applied to locate ports. The machine learning methodology used to locate ports in this thesis is clustering.

### Clustering

Clustering is referred to as grouping of objects, with similar characteristics, into the same group, or cluster. Clustering is used within many fields, where machine learning and pattern recognition are the most prominent. In addition, it is one of the most common approaches in unsupervised learning. From the literature study in Chapter 2, we saw that Rhodes et al. (2005) used machine learning and clustering to define normal operating condition in the different port activities, and how this could be used for anomaly detection in maritime monitoring. For the scope of this thesis, clustering could be used to identify port locations. Several approaches and algorithms for the technique are available, where the most relevant clustering approach for port location is density based, as according to Seif (2018) the number of clusters doesn't need to be specified beforehand.

### Density Based Clustering

Density-based clustering algorithms aim to find areas of higher density than the remainder of the data set. The most popular method for density-based clustering is the DBSCAN

---

<sup>2</sup><http://www.consultparagon.com/blog/what-is-data-enrichment-improving-your-data-to-add-value>, (accessed: 21.05.19)

algorithm. This method finds clusters of points that are in close proximity, based on a specified search distance (ArcGIS 2018). The method is illustrated in Figure 5.4. One advantage of this method is the possibility to expand the cluster in any direction and following also is better to detect clusters of complex shapes. One drawback to this method is that the method struggles to find different clusters with different density.

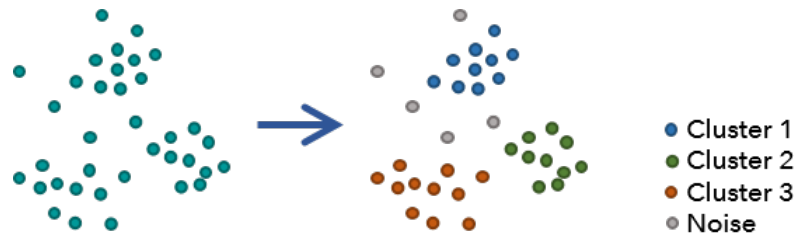


Figure 5.4: Visualisation of the DBSCAN algorithm (ArcGIS 2018)

In Python, the Scikit-learn library includes a variant of the DBSCAN method. This builtin function also allows for using the Haversine formula, as presented in chapter 5.3.2, so distances between points are more realistic. The builtin function don't include the functionality of finding each clusters's centre, so a function for calculation of this was created and can be found in Appendix B.6 under the name "*clusterCentre()*".

## Validation

To validate the port locations, the ports found using clustering will be compared to the one from Sea-web Ports. First, the distance from each clustered port to the closest port in the database will be evaluated. This will give an impression of the accuracy of the method, and could point to both locations that wrongly are labelled as a port, as well as ports that are located with low accuracy. Secondly, the distance from every message with navigational status equal to moored and speed equal to 0, to the closest port found both by clustering, and in the database, will be evaluated. This will potentially evaluate the accuracy of both methods, as well as indicating if clustering fails to locate some ports. By validating the result with these presented methods, a conclusion could be made for which port source will be utilised for the rest of this thesis.

### 5.2.2 Operational Status

One of the most important variable included in the AIS messages for operational analysis is the navigational status. This gives the current operational state of the vessel and is essential when analysing operational patterns. All dynamic messages include a variable that states the vessels navigational status. However, as presented in Chapter 4.1, this is manually inputted by the ships crew members. From the literature study, we see that among others Spiliopoulos et al. (2018) suggest that the information manually inputted by the vessel's crew is prone to

errors. This includes the static message (message type 5) as well as the navigational status included in the dynamic message types. A simple method for investigating to what extent these errors occurs is to examine if the messages include any obviously wrong statuses. If the vessel's status is set to moored we expect the speed to be zero, in a similar way if the ship is underway using engine we expect the speed to be over 0. We also expect that if a ship is lying at anchor the speed is close to 0. Figure 5.5 present a plot of AIS messages from a test dataset with obviously erroneous navigational statuses.

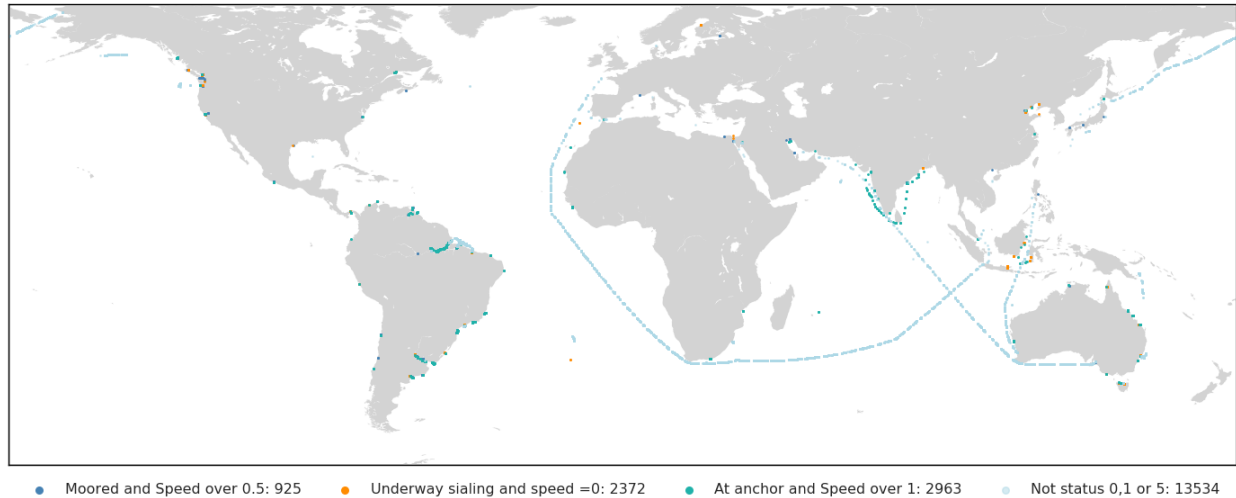


Figure 5.5: Visualisation of Erroneous Status Reports

From the figure, it is easy to observe that most erroneous reports happen close to a port, and could following be because the crew possible forget to update the status after a port visit or anchorage. We can also observe some voyages where the status is wrong throughout the voyage. The obviously erroneous status reports account for about 1% of the whole dataset, and 10% of the dataset with speed under 5 knots. The later is quite significant when analysing operational profiles a port and anchorage make out a considerable part of a vessels operation. Given that this is only the statuses that easily could be categorised as erroneous; the actual number of error could be higher, especially around ports operations. A new method for determining a vessel operational status could following improve the accuracy of operational analysis.

From chapter 4.1 we know the AIS signal includes a navigational status report where it is distinguished between different 15 statuses. For merchant vessels it is mainly three statuses of interest, namely: *underway using engine* (0), *at anchor* (1), and *moored* (5). The remaining are mainly for other vessel types as well as special cases such as *not under command* (2), and *aground* (6). For the scope of this thesis, where operational analyses are carried out, it is mainly three statuses of interest. We are interested in if the vessel is sailing, waiting to enter a port, or in port.

From the literature study, it became clear that some earlier studies have to some extent

established new operational statuses. Goldsworthy & Goldsworthy (2015) uses a combination of speed, shore closeness in addition to the included navigational status to validate the status. The method assigns the vessel as berthed if a message has speed less than 1 knots and within 1 km of the coast, and any other with speed to less than 1 knot to anchored. A key difference is that they are analysing edges and not single messages. Gao et al. (2016) also present a methodology for determining navigation status. The method distinguishes between three statuses, namely sailing, waiting in port or cargo handling. To determine the status the speed and closeness to berth are used, where all with speed greater than 3 knots is categorized as sailing. If the speed is under 3 and the position report is from within the berth, it is cargo handling, or else it is waiting. However, the presented method suggest that the distance that categorized if the vessel is within ports is presented as  $10 \text{ min} < 100 \text{ m}$ . If the minutes are referred to as geographical distance,  $10 \text{ min}$  corresponds to 20.4 km if we use the Haversine method, much more than the suggested  $100 \text{ m}$ .

For the scope of this thesis, as discussed earlier, it is sufficient to distinguish between three operating modes. We are interested in finding out if the ship is in a port, waiting to enter a port or sailing. The presented method is a development of the methods presented by Goldsworthy & Goldsworthy (2015) and Gao et al. (2016). The general methodology is first to validate the include navigational status, and if this is deemed wrong, it is assigned a new operational status by using speed and distance to port. The new methodology also captures vessels that are drifting and not anchored when waiting to enter a port. Independent if a vessel is anchored or drifting, the new operational status is set as a waiting vessel. The methodology is presented in Figure 5.6, where two parameters need to be determined. The distance to Berth threshold, labelled  $X$  in the figure, is dependent on the accuracy of the locations of the ports obtained. Further, the area of which a vessel could be categorised a waiting outside a port, labelled as  $Y$  in the figure, is investigated individually for each segment.

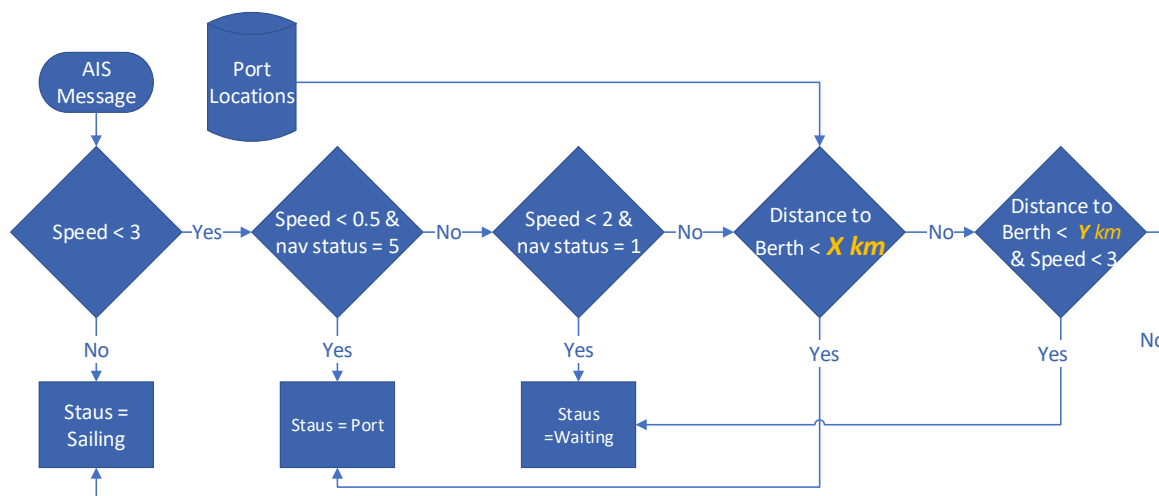


Figure 5.6: Procedure for determining operational status

The latter could be determined by investigating a statistical representation of the distance to port for vessels with navigational status equal to anchor and low speed. The result should point to the distance where the vessel could be categorised as waiting. This threshold is explored for the different segments, as operational differences could point to that this should be set individually for the different segments. With the resulting parameter, the method could be applied. By visually inspecting different ports area, and highlight both the old navigational status, as well as the new operational status, the methodology could be validated.

### 5.2.3 Leg Numbering

In operational analysis, it is essential to be able to distinguish between different voyages, port visits, and potential periods of waiting. Now, with a more reliable operational status, it is possible to categorise each leg, port visit and periods of waiting. By looping through the messages for each vessel, one could use the new operational status to label different legs, port visits, and periods of waiting. The labelling procedure for voyages is presented in the following pseudo-code. The same procedure could be used for labelling port visits and periods of waiting.

```
for for every vessel do  
  Leg Nr = 1  
  for for every message do  
    if status = sailing then  
      while status = next status do  
        Leg = Leg Nr  
        Leg Nr = Leg Nr + 1
```

## 5.3 Operational Analytics

The operational analytics chapter establishes the foundation of the methods used for analysing operational patterns for the different modes of operation. More specifically, the foundation for comparing the operational characteristics of the different segments, as presented in Chapter 3.6.

### 5.3.1 Area Segmentation

In operation analyses, it could be beneficial to include segmentation concerning the area of operation. This could give the possibility of finding potential operational differences given area of operation. It is normal to categorise the world ocean into five oceans, and this is a natural segmentation when analysing ship operations. One could also segment into more specific zones, such as a lands economic zone, and investigate if the following regulations in that zone could have an impact on how a ship operates. One way to establish regions is to create a geofence around that region.

#### Geo Fencing

Geofencing is when a virtual boundary, or fence, is defined around a geographical area in the real world. A geofence could have several different geometric shapes, but the most common is either polygons or circles. The methodology's goal is to determine if the message received is sent from within the defined boundary. There are many way geofencing is used today, where all from law enforcement to marketing have the later years utilized this <sup>3</sup>. In this report, geofencing will be used to define the world oceans. For the scope of this thesis, the Arctic waters are not of interest, so the segmentation will be done on the three remaining oceans. In Figure 5.7 the segmentation is visualized, where the oceans are presented, namely the Pacific, Atlantic and the Indian Ocean. Depending on the shape of the geofence, there are multiple ways one could calculate if the signal is from within a given geofence or not. If the boundary is a simple circle, the distance to the circle's centre would be sufficient. However, in this report, the world oceans are represented using polygons. One way to calculate if a point is inside a given polygon is the Ray Castings Algorithm.

---

<sup>3</sup><https://whatis.techtarget.com/definition/geofencing> (accessed: 20.09.18)



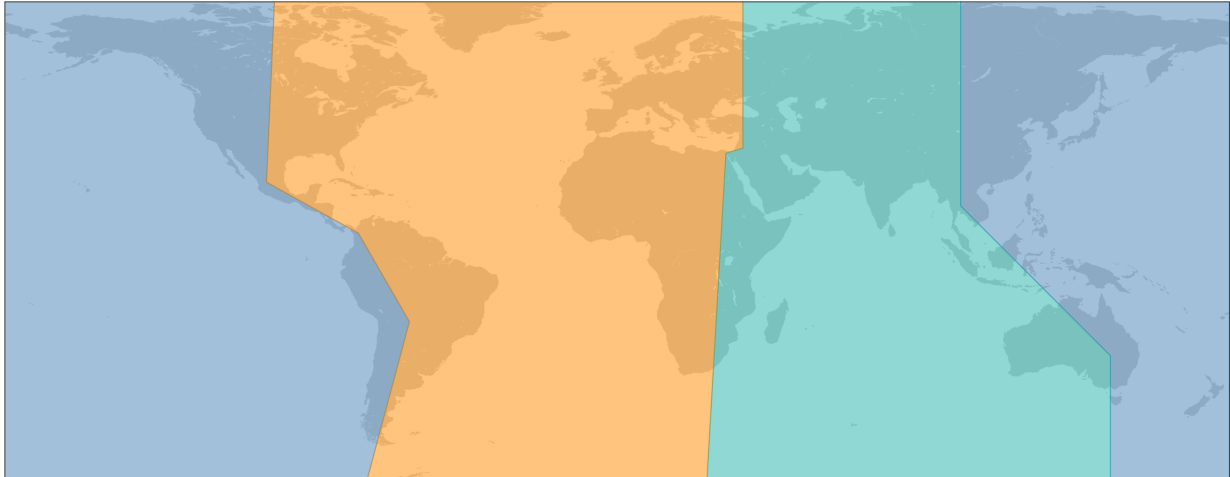


Figure 5.7: The World's Oceans

### Ray Casting Algorithm

Ray casting algorithm will determine if a given point is within the given polygon. The algorithm traces a horizontal ray from the given point and counts how many times this ray intercept with the polygon's edges. If the number of interceptions is odd the point is inside, if the number is even the point is outside. The following represents a pseudo code for the algorithm.

```

for for every point do
  inside = FALSE
  count = 0
  draw ray in horizontal direction
  for every edge of polygon do
    if ray intersect edge then
      count ++
  if count = odd then
    inside = TRUE
  
```

When using Ray Casting on AIS data it is important to have as efficient implementation of the method as possible. One method for improving its efficiency is, as suggested by Næss (2018), to only run the algorithm on points that already are within the polygons extremal values. In other words, if a point is from within a polygon, it's also from within the rectangle surrounding the polygon. Figure 5.8 present a visualisation of the methodology on a small dataset.

From the figure, we can observe that only checking the points within the polygons extremal values could have a significant impact on computational cost. The only points checked that not was inside the polygons was the ones in orange; without this improvement, even the black points have to be checked.

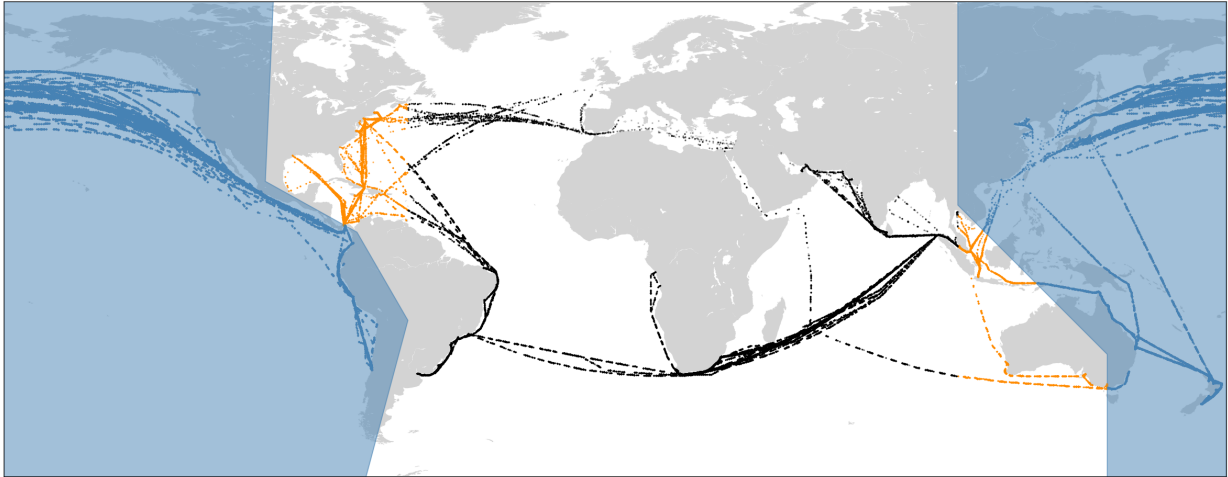


Figure 5.8: Ray Casting Visualisation

### 5.3.2 Distances

Several methods for calculating distances between points exists, where the *Euclidian* distance is the most used. This method builds on *Pythagoras theorem* and calculates the straight line distance between two points. In AIS messages the position is given in geographical coordinates. Coordinates are often given in degrees and minutes, following it is hard to relate a Euclidean distance calculated on coordinates. Another thing to keep in mind is the curvature of the earth, and especially with large distances, the Euclidean distance will not be accurate. In this thesis, the *haversine* formula is used. This utilises the great-circle distance, or in other words, the shortest distance between two points on the surface of a sphere, to calculate the distance between two points. The distance between two coordinates calculated with the haversine formula is:

$$d = 2R \cdot \arcsin \sqrt{\sin^2\left(\frac{\theta_2 - \theta_1}{2}\right) + \cos(\theta_1)\cos(\theta_2)\sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)} \quad (5.2)$$

Where  $R$  is the earth's radius,  $\theta$  and  $\lambda$  represents the latitude and longitude of the two coordinates respectively.

When using this formula to calculate the distance between two coordinates, we assume that the world's surface is that sphere. The method calculates the distance between two points on a perfect sphere. This is not the actual case as the radius of the world varies about 20 km between the equatorial radius and the polar radius. Given that in this thesis, the distances calculated are relatively close by, this assumption should give a good enough approximation of the real distance. In this thesis the radius used for calculations is set constant equal to the volumetric mean radius:  $R = 6371 \text{ km}$ <sup>4</sup> that is the most common when calculating distances

<sup>4</sup><https://nssdc.gsfc.nasa.gov/planetary/factsheet/earthfact.html> (Accessed: 19.04.19)

between two coordinates.

### 5.3.3 Loading Condition

Loading condition is an operational variable used in several shipping studies. In operational studies, it could be beneficial to segment voyages into the ones that could be characterised as in laden and the ones characterised as in ballast condition. One possible way to determine if a voyage is in ballast or laden is by investigating the draught ratio. The draught ratio could be defined as follows:

$$\text{draught ratio} = \frac{D}{D_d} \quad (5.3)$$

where  $D$  is the current draught, and  $D_d$  is the ships design draught. Adland & Jia (2016) uses the draught ratio to decide whether a VLCC is in ballast or laden condition. Further, they suggest that for a VLCC observation with draught ratio between 80% and 100% are classified as laden, and for observation between 25% and 65% could be classified as in ballast.

As this thesis investigates different shipping segments, this classification will not be applicable for all segments. For both industrial and tramp shipping it would be reasonable to expect that the ship is either close to fully loaded or in ballast condition. Where for liner shipping we would expect to have more of a dynamic draught ratio. However, due to an imbalance in trade as discussed earlier, we should expect that some of the legs would redistribute empty containers, but this could be harder to find in the draught ratio.

# Chapter 6

## Case Study

This case study addresses the main objectives of this thesis, where operational patterns are analysed for the different modes of operation. The case study consists of three main parts. First, a data preparation procedure is carried out for the obtained database. Secondly, the data enrichment process is developed further and evaluated. Lastly, the new operational status is used to analyse the operational patterns, as described in Chapter 3.6. The methods and results are presented and discussed throughout this chapter.

### 6.1 Data Preparation

A representative selection of AIS data must be obtained in order to carry out the case study where the operational pattern are compared between the three modes of operations. As discussed in Chapter 3.6, a shipping segment that represents liner operation is the container ship. Similarly, bulk ships represent tramp operations, and LNG represents Industrial operations. These shipping segments are used to obtain a quantitative data foundation for representation of the different modes of operations.

#### 6.1.1 Obtaining Data

The Norwegian Coastal Authorities granted the AIS data available in this thesis. The available data contains all AIS messages received by their own satellites during the year of 2018. To obtain a representative selection vessel to analyse it was chosen to access Sea-web and create ship lists with only ships of interest, rather than use the heuristics as developed by Smestad et al. (2017). From Chapter 4.2 we know that Seaweb includes the possibility to filter both on ship type and if the ship currently is in service or not. According to IHS (2019a), the number of in-service vessels that categorises as bulk, container and LNG carriers were respectively 11847, 5215 and 520.

Bulk and Container vessels come in all sizes and operate in both short-sea and deep-sea shipping, while LNG vessels is a more specific shipping classification. To ensure that the operational comparison is executed on as equal terms as possible, specific shipping segments within container and bulk shipping was selected. This will help to ensure the resulting operational differences is due to the mode of operation rather than vessel specific differences, e.g. that the vessels size impact the operating speed. For the same reason, only the newest vessels were selected. The Panamax class was chosen to represent container and bulk vessels. As the latest data available was from 2018, vessels built in 2018 or after was filtered out, so data is available throughout the whole year for every ship. As AIS data is as large as it is, ship list with the approximately 500 vessels was sent for extraction to limit the size of the data to be analyses. Bjørnar Brende Smestad created the database by using the methodology as presented in Chapter 5.1.1, with data from the Norwegian Coastal Authorities. The size of the returned database is presented in Table 6.1.

Table 6.1: Original Database

	Message Type 1	Message Type 2	Message Type 3	Message Type 5
Message Count	56 860 136	13 025	4 154 612	4 256 261
Fraction of Total	0.87	0.0002	0.06	0.07

## 6.1.2 Data Structuring

The structure of the obtained database was, as presented in Table 6.1, one table for each message type. These message types contain different information. As discussed in chapter 4.1.5, message type 1,2, 3 three contains dynamic information such as position and speed, while message type 5 contains static voyage related data, such as vessel information and destination. From table 6.1, we see that Message type 1 contain 87% of all messages. In this thesis, a high message frequency is beneficial, and why the three different dynamic message types were combined into one. For later analyses, it must be easy to distinguish between the different shipping segments, so the database was split into three different databases, one for each segment. The procedure is presented in chapter 5.1.3.

As mentioned, it is desirable to have a database with a high message frequency as possible. From experience, the number of messages sent from different vessel tends to vary. To ensure high message frequency throughout the year, a further filtration process is carried out. By using the query presented in chapter 5.1.2, the number of messages sent from each ship could be counted. The distribution of messages sent from different vessels the whole year is presented in Figure 6.1.

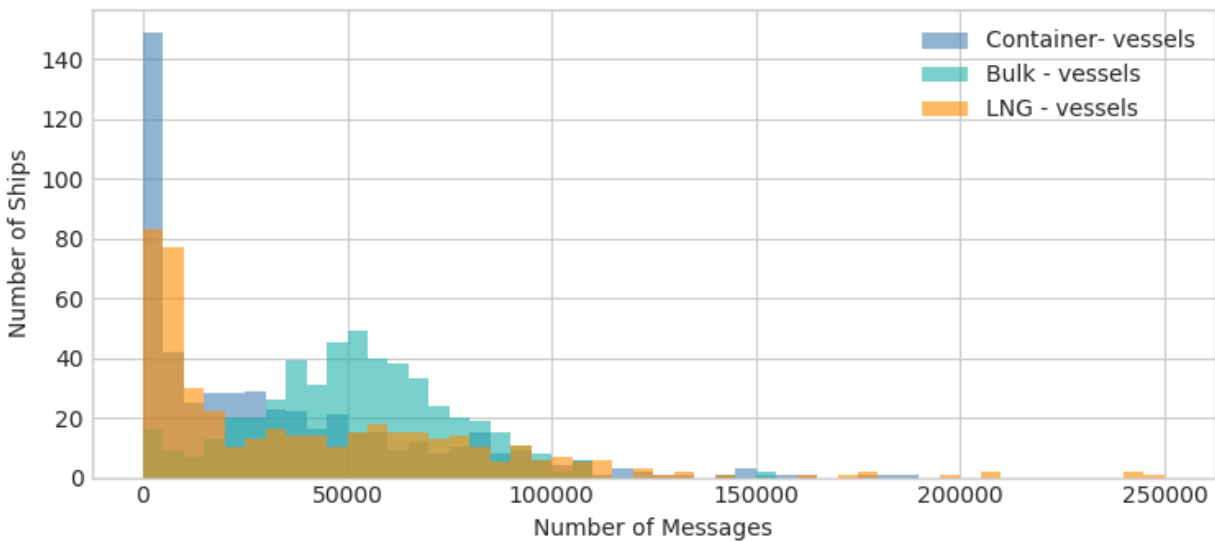


Figure 6.1: Yearly Message Count for each Vessel

The results suggest that it is a significant variation concerning the number of messages sent for each ship. Many ships have a message count of less than 20 000 during 2018 or an average of 54 messages per day, significantly less than the intervals presented in Chapter 4.1.3. The 200 vessels with the highest message count from each segment was selected for further analyses. This ensures that each vessel includes a minimum of 33 000 messages.

Smestad (2019) suggests that indexing the database could lower the time of analyses significantly. In chapter 5.1.1 an investigation of the impact indexing could have is presented. The result suggests this could reduce computational time significantly. However, this also increases the size of the database, and it is following beneficial to limit the variable indexed. The resulting databases are indexed on time (unixtime), and MMSI number (userid) as these are frequent variables used when filtering AIS data.

### 6.1.3 Data Validation

Before carrying the operational analyses, it is important to validate the data. Chapter 5.1.4 present several potential errors the data could have, and how these could be addressed. The first possible error could be the messages MMSI number. In this thesis, where the databases are created by filtering out ships found in ship lists obtained from Seaweb, we already know the messages in the database contain a valid MMSI number. However, by filtering on MMSI number, one potential error is still remaining. That is if an MMSI number is associated with different vessels during the time of the analyse. This is addressed by exploring if any MMSI numbers are associated with more than one IMO number in the static messages. When using the procedure as presented in chapter 5.1.4, the results was that one MMSI number was associated with two IMO numbers in both the LNG and the Bulk database. All messages

with these MMSI number was removed.

Other possible variables that could be erroneous is speed and position. Messages with speed and position reports without the expected range, as presented in Chapter 5.1.4, are also filtered out. Table 6.2 present the size of the dynamic messages before and after the validation process.

Table 6.2: Database Size After Validation

# of Messages	Container	Bulk	LNG
Before Validation	14 066 817	15 052 099	15 279 088
After Validation	14 048 102	14 871 299	13 470 968

The results suggest that container, bulk and LNG databases contain respectively, 0.1%, 1.2% and 11.8% erroneous messages. The literature study has suggested finding erroneous messages in size range similar to the ones found for Container and Bulk vessels. However, the result from LNG was higher than expected, as 11.8% is a substantial part of the database. With further exploration of the LNG databale, a large part of these erroneous messages are vessels listed with speed = 102.300003 knots and navigational status equal to "undefined" (status number 15). No trend is found when plotting the positions of these erroneous messages. The location reports seem to give correct information, while the speed reports are wrong. This could point to some kind of equipment failure, but no conclusion could be made as several vessels had similar erroneous reporting, and this only was the case for the LNG database. These messages were removed in the final databases.

## 6.2 Data Enrichment

With the database validated and prepared for analysis, the process of data enrichment can be initiated. The methodology is presented in chapter 5.2, where the goal is to establish new and more accurate operational statuses and use these to distinguish between each distinct voyage, port visit and waiting periods.

### 6.2.1 Port Locations

The first step in order to obtain new and improved operational statuses is to establish accurate and reliable port locations. As suggested by Millefiori et al. (2016), a port's operational region do not always stay constant, but evolves over time, and following lead to that the Port Databases contain outdated information. From chapter 4.3, we know the accuracy of the port locations is limited to the closest geographical minute, or in other words about 1.8 km in metric surface distance. These two things combined are the background for the investigation if port regions could be determined by AIS signal alone. Another potential

benefit this could lead to is that only ports that are visited by the vessels subject to analyses are included. Even if the Sea-web Ports includes the possibility to filter on facilities, this will give a broad spectre of ports, more than the ones actually visited by the ships included in the analysis. This will especially be the case for bulk and container ships. Including as few ports as possible is beneficial with respect to computational performance.

Chapter 5.2.1 presents a method for finding ports by using AIS data only. The proposed method applies a density-based clustering method called DBSCAN. This method is applied to the three different datasets to locate ports visited within each shipping segment. The clustering algorithm is only run on messages with navigational status set to moored and speed under 3 knots to increase performance and accuracy of the results. An implementation of the clustering method in Python is presented in Appendix B.6, under the name "cluster-DBSCAN()". Table 6.3 present number of port found for the different segments, both from the database, and using the clustering method presented in chapter 5.2.1. From the result, we see that the number of ports found by clustering is significantly less than the one found in Sea-web Ports. It is easy to see the potential computational benefit this could have.

Table 6.3: Number of Ports: Sea-web Ports vs. Clustering

	Container	Bulk	LNG
Sea-web Ports	7887	10675	1868
Clustering	238	278	71

With the resulting port location from the clustering method, the method could be validated, and the final port locations could be determined. As suggested in chapter 5.2.1, the result is validated in two ways. First, it is investigated if the clustering algorithm locates any ports not found in the port database. This is found by calculating the distance for every port found using clustering, to the closest port in the database. The results are presented in Table 6.4, where we see the number of clustered ports with a distance above 3 and 10 km to the ports from Sea-web Ports.

Table 6.4: Number of Clustered Port Not Found in SeaWeb

	Container	Bulk	LNG
Number with $d > 3$	15	10	9
Number with $d > 10$	9	5	8

This result can both suggest that the clustering method locates ports that the Sea-web ships database doesn't include, and that the clustering algorithm locate ports that not is an actual port. The result can following not be analysed individually. The second validation method is to calculate the distance from each message that unarguably is in a port, this is messages that have navigational status equal to moored and speed equal to 0. The distance is calculated both to ports found using clustering, as well as ports from the Sea-web Ports database. Table



6.5 presents the statistical results, where mean, standard deviation and percentiles are given as distance in kilometers.

Table 6.5: Accuracy of Port Locations: Sea-web vs. Clustering

	Database Ports				Clusterd Ports			
	Mean	STD	95%	98%	Mean	STD	95%	98%
Container	0.73	0.49	1.05	1.17	0.35	6.97	0.97	1.23
Bulk	0.89	1.23	1.45	2.00	0.57	14.64	1.28	1.77
LNG	1.59	3.8	4.8	5.02	0.53	21.85	0.46	0.48

From the result, we see that the mean distance is higher to the ports in the database than the one found using clustering. This is in line with earlier expectations, as the accuracy of the ports in the database is down to closest 1.8 km. By investigating the percentiles we see that clustering can locate ports more accurate for 98% of the cases for Bulk and LNG vessels, and in 95% of the cases for Container vessels. Especially for LNG vessels, the result sees to be in favour of the clustering method. A possible reason for this could be ship to ship transfers, where the navigational status is equal to moored. The location of a ship-to-ship transfer is not in the port database, and why we see a trend of a higher mean distance to the database ports for the LNG segment.

If the distance's standard deviation is taken into account, we see this is significantly higher for the clustered ports. Given that the results from the clustering method seem to give better results for a minimum of 95% of the messages, in combination both with the results presented in Table6.5, and that the standard deviation of the distance is higher, we can conclude that the clustering method fails to locate some ports.

Even if the clustering method seems to locate ports with higher accuracy, there is a problem that it failed to find some ports. The resulting accuracy from the Port Database was as expected somewhat inaccurate due to the resolution of the coordinates. However, these ports had a significantly lower standard deviation. The smaller standard deviation points to that no visited ports were missing if we disregard ship-to-ship transfers in the LNG segment. The labelling of operational status procedure requires the location of every port visited to give accurate results. Given that the clustering method tends to both locate ports that don't exist, and fail to locate ports that exist, the port location used in the rest of this thesis is from Sea-web Ports and not the ones found with clustering.

## 6.2.2 Labelling Status

As presented in chapter 2, several sources suggest that the only true reliable data for the AIS signals are the signals sent from dynamic equipment onboard the vessel. Spiliopoulos et al. (2018) suggest that the navigational status is manually inputted, and following prone to error. From Chapter 5.2.2, we saw that when investigating a test dataset, the result suggested that

about 10% of all messages with speed under 5 knots had obviously wrong operational status. The operational status is a key variable when conducting reliable operational analyses. Now, with final port locations, the procedure for establishing a new operational status can be carried out. The methodology developed is presented in chapter 5.2.2.

By investigating messages with obviously erroneous messages, we can get an impression of the impact a new operational status could have. The result is presented in Table 6.6, where status equal to 5, 0 and 1, suggests the vessel respectively is moored, sailing and at anchor.

Table 6.6: Messages with wrong navigational status

Segment	Status = 5 & Speed >0.5	Status = 0 & Speed = 0	Status = 1 & Speed >1	% of Msg, Speed <5	Status Not 0,1 or 5
Container	28 935	50 453	3 314	12.7 %	101 368
Bulk	2 730	20 621	5 374	4.3%	112 144
Lng	6 455	34 527	3 484	9.9%	189 663

This results support the result from the test dataset presented in Chapter 5.2, were a significant percentage of the messages with speed below 5 knots, had obvious wrong navigational statuses. Besides, we see that many messages include other navigational statuses, then the three of interest. With further investigations, there are two statuses dominating in all segments, that is *undefined* and *underway using sail*. The latter is most likely only a misunderstanding by the crew, as a vessel is often is referred to as sailing when it is on its way. In the AIS messages, however, there are separate statuses if the vessel is underway using the engine or using sail.

As presented in Chapter 5.2.2, two parameters must be determined before the labelling procedure could be carried out. The first is the limit as to where a vessel could be characterised as within a port. In chapter 6.2.1, the accuracy of the port locations was evaluated, where it was chosen to use the ports from the Sea-web Ports database. Even with port location with a resolution down to closest 1.8 km, we know from table 6.5 that least 95% of the messages were within 1.45 kilometre of the port locations, if we disregard the ship-to-ship transfers for LNG vessels. To ensure no vessel would be excluded, a range of 1.5 kilometres was selected for all segments in the labelling method.

The second parameter that needs to be determined is the range of where a vessel could be categorized as waiting, either at anchor or drifting, to enter a port. By investigating the distance to port for vessels at anchor, this range could be determined. Table 6.7 present a statistical representation of this distance for vessels at anchor for each of the three different segments.

The results suggest that the vessels in the different segment tend to wait with varying distance to a port. However, the waiting distance tends to be somewhat similar for Container and LNG vessels, while the result for the LNG vessels is significantly higher if we look at mean and standard deviation of the distance.

Table 6.7: Statistical Representation of Waiting Vessel Distance

Segment	mean	std	3%	70%	80%	90%	95%
Container	18.4	23.7	2.6	18.1	22.1	44.97	51.15
Bulk	14.37	14.84	1.2	16.6	19.8	25.34	31.6
Lng	305.4	719.8	6.1	48.0	69.5	1147.5	1753.7

It can be two reasons why the LNG vessels have a much higher mean and standard deviation. First, IGU (2019) suggest a low utilisations rate of LNG vessels during the first quarter of 2018, and following the increased distance could be vessels that are layup at anchor. Secondly, as discussed earlier, LNG sometimes uses ship-to-ship transfers and following not visit a port. It is latter that is more likely as these locations are not found in the Sea-web Port database.

With all this taken into account, we see that a distance of 50 kilometres should capture most waiting vessels, with an accuracy of over 70% for LNG vessels and 95% for Container and Bulk vessels.

With both parameters determined, the new operational statuses could be determined. A function for establishing new operational statuses was implemented and can be found in Appendix B.5, under the name *LableStatus()*. The function utilises the procedure as presented in Figure 5.6, with the determined parameters. The parameters used were the same for all three segments, and was respectively 1.5 kilometres and 50 kilometres, for  $X$  and  $Y$  as presented in Figure 5.6. The procedure was applied to all dynamic messages for each segment. Table 6.8 present the number of messages that changed it's operational state from the included navigational status.

Table 6.8: Number of changed Operational Statuses

	Container	Bulk	LNG
Number of messages with change status	348 753	209 284	271 899
Percentage of all messages	2.6%	1.4%	2.0%

From the result, we see the procedure changes operational statuses for up to 2.6 % of all messages in the dataset. This is significant as even if port operations are an essential part of shipping operations, most vessels spend significantly more time sailing than in port. Many of the changes done are related to messages with low speed, as the methodology labels all messages with speed over 3 knots as sailing. Following this method have a significant impact when carrying out operational analyses on AIS data.

As suggested in Chapter 5.2.2, the results from the methodology can be validated by visually inspecting ports area, where the old and new operational statuses are highlighted. Many port location was investigated. Figure 6.2 present an example, where all messages from container traffic around New Zealand is plotted for the year of 2018.

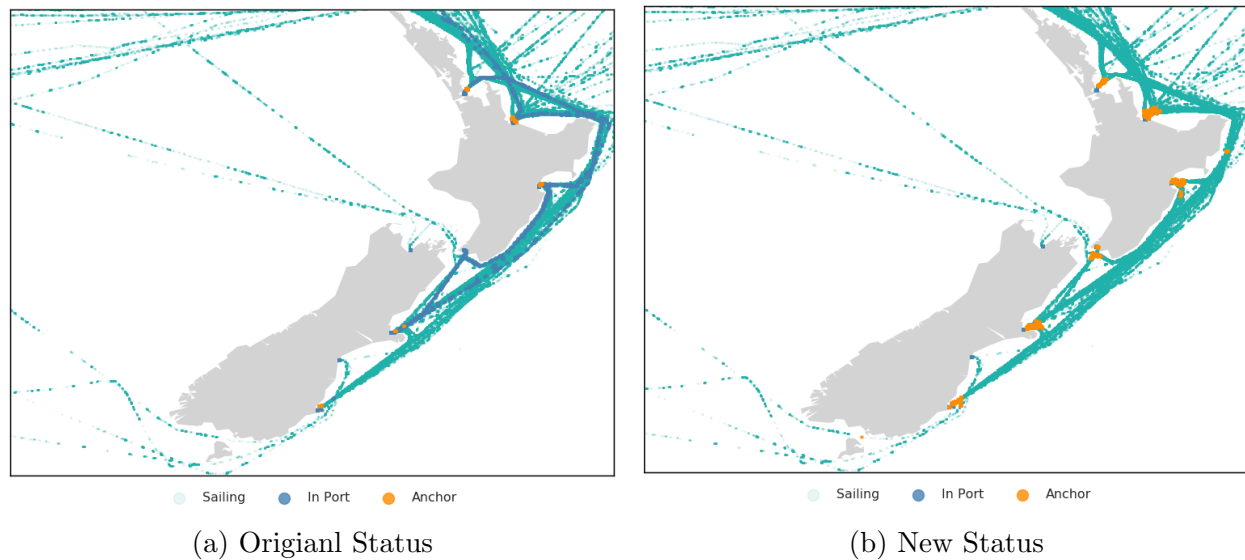


Figure 6.2: Operational Status; Original vs. New

From the figure, we can see the methodology gives promising results. In Figure 6.2a we see some voyages in blue, this is vessels with the original navigational status equal to *moored*. This is vessels that not updated their navigational status after a port visit. From Figure 6.2b, the new method manages to address this when labelling a new operational status. We can also observe that the method also categorize significant more vessels as waiting. There are mainly two reasons for this. The old navigational status does not include a state for waiting without anchorage. As discussed chapter 5.2.2, some vessel tends to drift when waiting to enter a port, something the new method captures. The other possible reason is that the vessel's crew either don't bother or forget to update the navigational status to at anchor when laying at anchor.

### 6.2.3 Leg Numbering

With a more reliable operational status, it is possible to categorise each ship's operations into different voyages, port visits, and time of waiting. Chapter 5.2.3 presents a methodology for utilising the new operational status to number into distinct legs, port visits and time of waitings. The procedure was implemented and can be found in Appendix B.5, under the name *legNumbering()*.

One thing to keep in mind with this function is that this will give even small voyages a distinct number. So, e.g. if a ship has waited to enter a port, the journey from the location where it waited to the port, will be categorised as a distinct voyage. However, this could easily be removed from the operational analyses by filtering out legs with low sailing distance.

## 6.3 Operational Patterns

In this section, the main objective of this thesis is addressed. The goal is to investigate operational patterns for the three different modes of operations. More specifically investigate the operational characteristics for the different modes of operation as presented in Chapter 3.6.

### 6.3.1 Trade Flows

As discussed in chapter 3.6, we expect to find some general trade differences for the three segments. To explore these patterns, and on the same time, carry out a last validation process of the data, as suggested in Chapter 5.1.4, a function for creating a density plot was created. The function plots thin transparent lines throughout each vessels' routes. The function is presented in Appendix B.7 under the name *mapPlotDensity()*. Figure 6.3, 6.4 and 6.5 presents the results for respectively Container, Bulk and LNG segment.

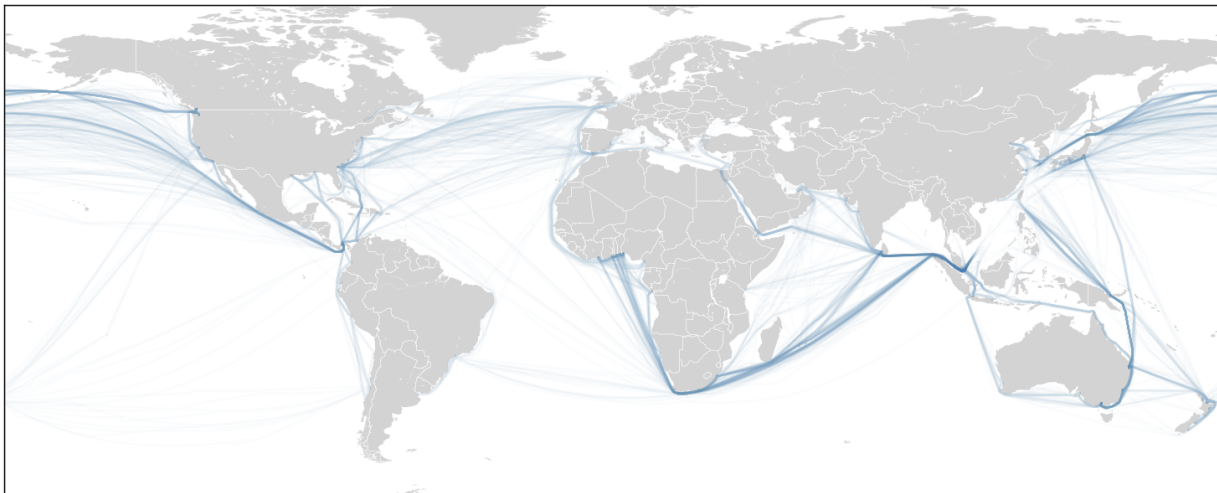


Figure 6.3: Density Plot, Container Vessels

The resulting plots suggest no messages come from unexpected locations, such as messages located on land, and following the last step of the validation presented in chapter 5.1.4 is completed. We also to some extent find support for that the container segment have more established trade routes than bulk and LNG, as each container vessel only visit a small number of ports in a pendulum service as suggested by Fagerholt (2018), while bulk and LNG operate in more a port to port service, that following creates more dynamic trade patterns.

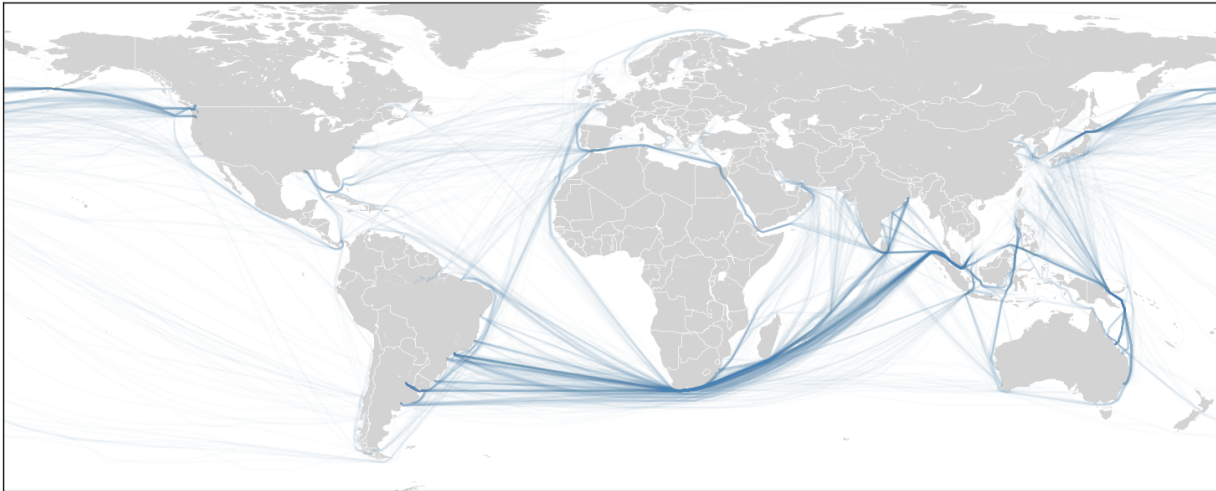


Figure 6.4: Density Plot, Bulk Vessels

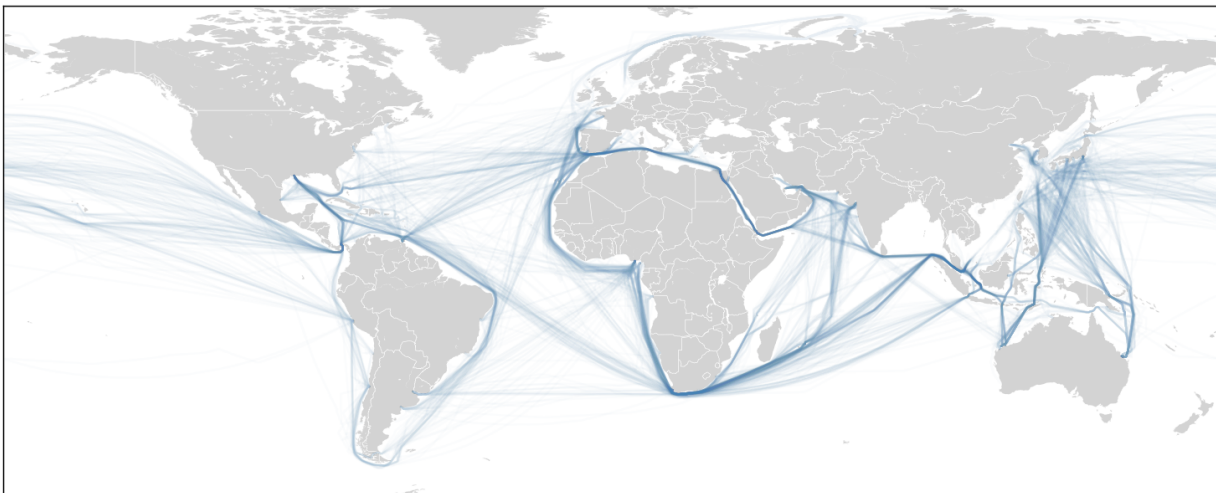


Figure 6.5: Density Plot, LNG Vessels

### 6.3.2 Loading Condition

As discussed in chapter 3.6, it is expected to find significant differences in the operational patterns for LNG and bulk depending on whether the vessels is in ballast or laden condition. First, it must be determined at which draught ratio each vessel could be categorized as laden or in ballast. Chapter 5.3.3 present a method for finding the vessels draught ratio. The design draught of a vessel is included in the ship lists obtained from Sea-web. A function that uses the design draught to label all static message types with the current draught ratio was developed and can be found in Appendix B.5 under the name *lableStatDraughtRatio()*. With inspiration in the methodology as presented by Adland & Jia (2016), the result is

presented as histograms in figure 6.6a and 6.6b for respectively Bulk and LNG vessels.

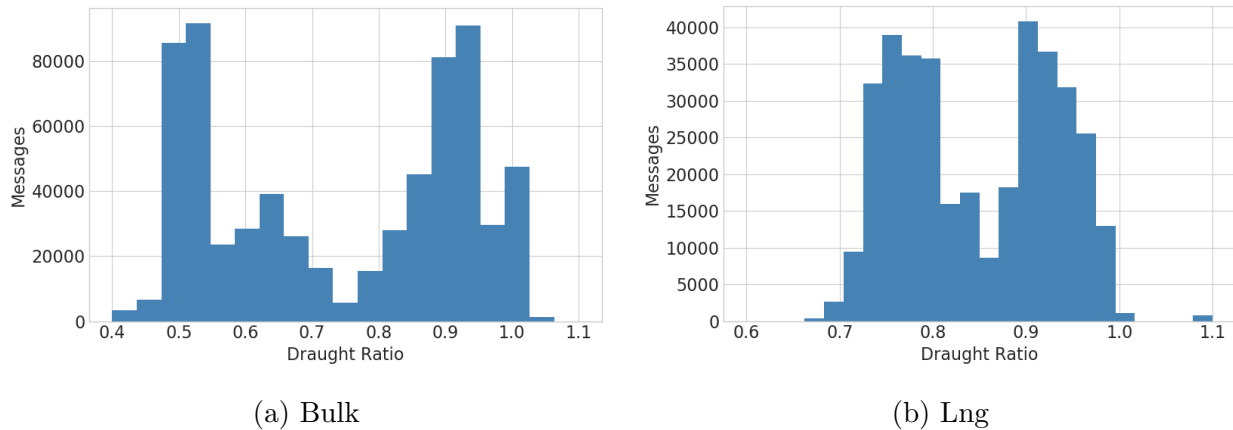


Figure 6.6: Draught Ratio Distribution

As suggested by Adland & Jia (2016) peaks in the resulting distribution of the draught ratio could be used to categorise the ranges for where vessels could be labelled as in ballast or laden condition. In Figure 6.6 we can observe two distinct peaks for both segment. However, the peaks are at different draught ratios. More specifically, the lower peak for the bulk segment seems to happen at lower draught ratio than for the LNG segment.

Levander (2012) suggests that a vessel's design could be categorized into two different types. The first is a capacity carrier where the volume of the cargo determines the size of the vessel, the second is a deadweight carrier where the weight of the cargo determines the size. An LNG vessel could be categorised as a volume critical vessel, as the density of LNG is approximately 40% of the water's, while a bulk could be categorised as a weight critical vessel, as the density of the cargo could be up to three times of the density of water. This explains why we observe a larger difference in draught in the two conditions for bulk vessels compared to LNG.

We can also observe some vessels are assigned a draught ratio over 1. This is most likely because of that the current draught of the ship, as discussed in chapter 4.1, is manually inputted by the vessel's crew. The current draught determines the water depth needed for a ship to operate safely, following it is likely that the crew give a conservative draught, rather an underestimate, and following why some vessels are presented with a draught ratio higher than 1.

From Figure 6.6, we can determine the range of which each segment could be characterized as in ballast or laden. For bulk vessel, a draught ratio below 0.7 should capture most vessels in ballast, while a ratio above 0.8 the laden. The transient state between 0.7 and 0.8 will be removed from analysis to improve the reliability of the method. Similarly, for the LNG segment, vessels with a draught ratio above 0.9 are categorised as laden, while a draught ratio below 0.8 as in ballast.

Now, with the possibility to distinguish between laden and legs in ballast the expected differences for distances and speed, as presented in Chapter 3.6, could be investigated. Assmann

et al. (2015) claim that speed optimisation is more common for trips in ballast than laden. A boxplot was created to investigate the claim, where the average sailing speed for all voyages in the different condition is presented for the two segments. The boxplot is presented in Figure 6.7, where speed is given in knots.

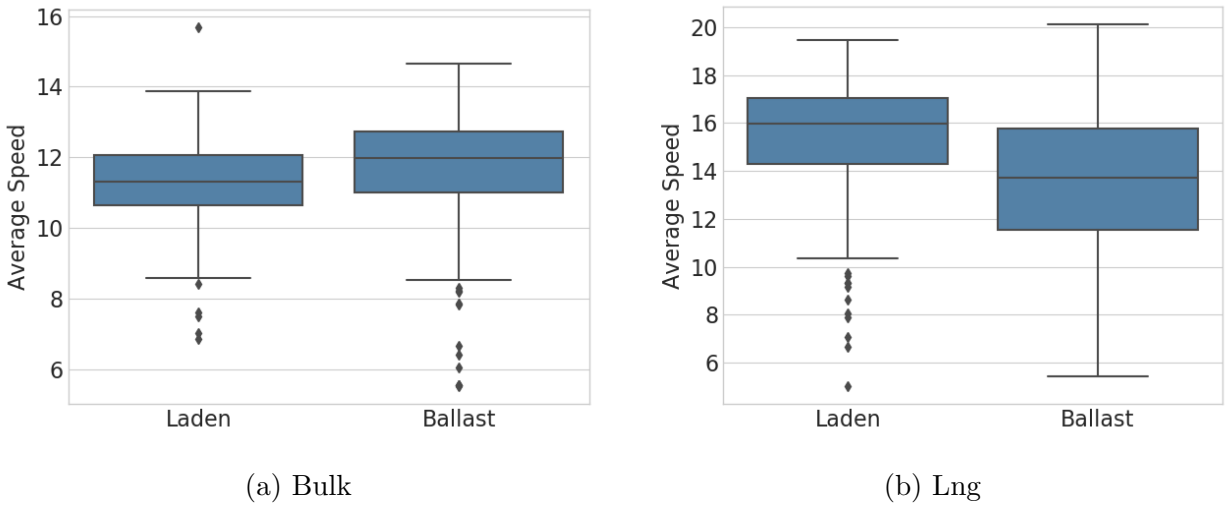


Figure 6.7: Sailing Speed: Ballast vs Laden

However, Assmann et al. (2015) mainly referred to speed optimization as slow steaming. We can observe from Figure 6.7a that even if the variance of the sailing speed is, as expected, higher in ballast, the average speed also tend to be higher. This could be an increase in sailing speed to reach a new spot contract. From Figure 6.7b we also observe a more significant variance in ballast, but the speed tends to be lower than in laden. This is expected as LNG operates in industrial shipping, where the operator also controls more of the operation and could more easily adapt slow steaming to reduce fuel cost on backhaul trips.

It is also expected to find significant differences in voyage distance for the different loading conditions. A function for calculating the sailing distance that uses the haversine formula presented in chapter 5.3.2 was created. This function could be found in Appendix B.5 under the name *legAnalytics()*, and calculates the cumulative sum of the haversine distance between each position related to the same voyage. The result is presented in Table 6.9, where all distances are given in kilometers.

Table 6.9: Sailing Distances: Laden vs Ballast

	Laden Voyage				Ballast Voyage			
	Mean	STD	25%	75%	Mean	STD	25%	75%
Bulk	9 682	10 399	3 641	12 483	5 570	14 050	612	7 811
LNG	8 944	23 235	3 260	8 554	6 842	31 530	1 430	7 003



From the table we can observe, as expected due to the mode of operation, bulk vessels have significant shorter sailing distances in ballast than when laden. We can also see that the standard deviation is higher. This is expected, as even if vessels try to minimize their sailing distance in ballast, as suggested by Christiansen et al. (2004), new spot contracts are not always close, so significant sailing to the next contract is needed. For LNG vessels, as it is harder to minimise the voyages in ballast due to the mode of operation, the result shows some support that the sailing distances in ballast and laden condition are more related than the case for bulk ships. However, there is a somewhat noticeable lower sailing distance in ballast. One reason could be because some LNG vessel operates with short term contract as suggested by IGU (2019), or that some industrial operator also chose to accept some spot contracts.

### 6.3.3 Container Redistribution

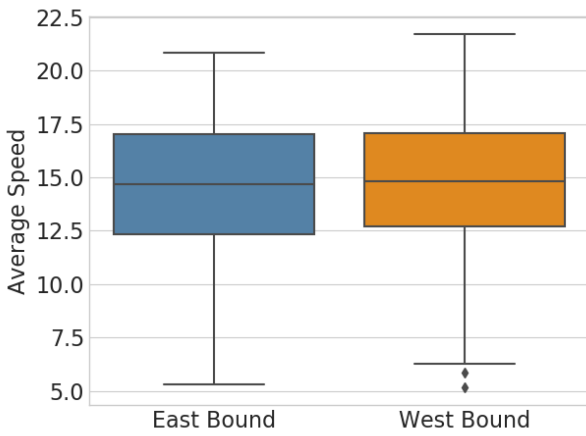
As container vessels are operated as liners, the need for legs without any paying cargo aboard is not a concern in liner shipping (Grammenos 2010). However, due to an imbalance in trade for different regions, a container vessel often has a high load factor for revenue-generating containers in one direction, and a lower load factor in the opposite direction due to the need of container redistribution. According to Grammenos (2010) it is on the trade routes between America and Asia, and Europe and Asia where there is a significant need for container redistribution. As suggested in 5.3.1 geofencing could be used to determine in which region the messages originate from. Figure 6.3 present the trade pattern for the container vessels. By determining in which of the world oceans the voyage is in, as presented in chapter 5.3.1, the voyage could be assigned to different trading routes. To determine if the vessel is heading eastbound or westbound the course of ground variable included in the AIS messages could be used. The result is presented in Figure 6.8 and 6.9 for respectively the Trans-Pacific trade and the Asia-Europe trade.

The result suggests as expected container redistribution do not seem to affect sailing speed, however, we could see that the westbound sailing in Trans-Atlantic trade has somewhat smaller variance. From Figure 6.8b, we can observe a slightly lower draught ratio for the Westbound voyages, and is in line with expectancies. Similarly, from Figure 6.9 we see a noticeably higher draught ratio in the westbound trade. Table 6.10 present the actual containerized trade for these two trade routes.

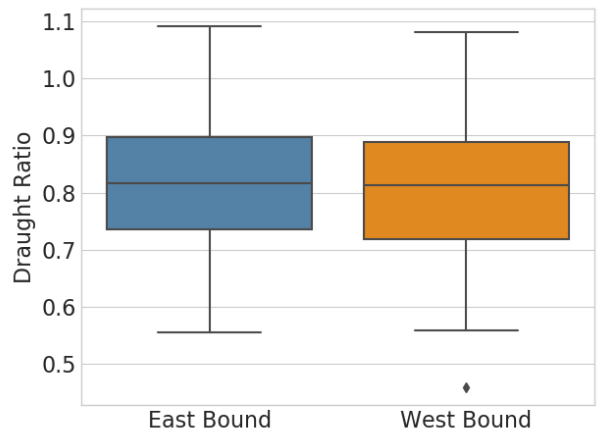
Table 6.10: Containerized trade on major Eas-West trade routes for 2018 (UNCTAD 2018)

	Trans-Pacific		Asia-Europe	
	Eastbound	Westbound	Eastbound	Westbound
2018	19.5	8.1	7.8	16.9

Given in Million 20-foot equivalent

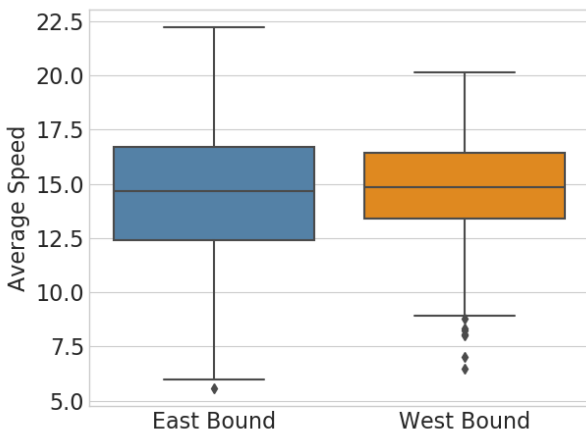


(a) Speed distribution

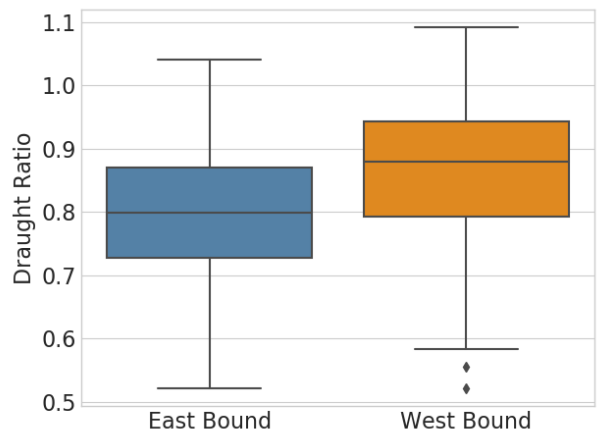


(b) Draught ratio

Figure 6.8: Trans-Atlantic Trade



(a) Speed distribution



(b) Draught ratio

Figure 6.9: Asia - Europe Trade

The draught ratio seems to indicate in which direction the trade is highest. However, from the trade ratio in table 6.10 we should expect the draught ratio differences for the transatlantic trade should be higher than for the Asia-Europe trade routes, not the other way around as Figure 6.8b and 6.9b suggest. From the result, we some support that the draught ratio could help find container redistribution, however as presented by among other Adland et al. (2017) that the draught alone not accurately estimate a vessels payload, but could be used as an indication of the loading condition.

### 6.3.4 Port Times

It is also expected to find significant differences in port times for the different modes of operation. With a more reliable operational status and all port visit categorised into distinct port visits, the time in port can be found efficiently. In appendix B.5 under the name *timeAnalytics* the function for finding the time in port and time of waiting is presented. A statistical presentation of the time spent in port for the different segment is presented in Table 6.11, where the time is given in hours.

Table 6.11: Time in Port

Segment	mean	std	25%	75%
Container	25.8	37.4	12.4	30.5
Bulk	68.9	58.0	30.6	89.7
LNG	35.9	231.9	20.0	30.0

From the table we see, as expected due to the mode of operation, that container vessels use significantly less time in port than especially than the bulk segment. Standardized processes, and increase port efficiency in the liner industry due to the consolidation in the industry, as suggested by UNCTAD (2018), is one of the main contributions for this. We can also observe that the LNG segment have a high standard deviation compared to the port time included in the 75th percentile, this suggests that some vessel spend a significantly longer time in port than others. There could be mainly two reasons for this. First is if the vessel is waiting in a port controlled by the operator and waiting for cargo to transport. Secondly, this could be because of a phenomenon called Boil-off. When an LNG vessel is transporting gas, the gas is in an extremely cooled state. Following the tanks also need to be cooled when the LNG is loaded. As suggested by Dobrota et al. (2013) it is common to keep some of the LNG in the tanks on backhaul trips, often referred to as heel, to keep the tanks cooled. However, due to a problem called boil-off, that is that some of the LNG gas is constantly evaporating during the voyage, some LNG vessels could run out of the heel before entering a port. The tanks are then in need of cooldown, something that is a time demanding porcess.

### 6.3.5 Waiting Times

The different modes of operation should also result in a difference in waiting time outside a port. The waiting times for the different segments were found using the same methodology as for the time in port. As the new operational status also captures vessels waiting outside a port, the waiting time could be found efficiently by categorising each message with status equal to waiting into distinct visits. A statistical presentation of the resulting waiting times is presented in Table 6.12, where all times are given in hours.

The result suggests as expected that container vessels spend less time waiting outside a port. We also find some support for the claim made by Gao et al. (2016), that even if they have

Table 6.12: Waiting Times

<b>Segment</b>	<b>mean</b>	<b>std</b>	<b>25%</b>	<b>75%</b>
Container	15.9	58.1	2.0	12.4
Bulk	77.9	122.2	6.7	96.4
LNG	46.3	89.4	3.0	50.2

the most efficient port operation, they tend to arrive somewhat earlier than their assigned port times to ensure they not being fined if arriving late due to weather. We also expected that LNG vessel could enter a terminal directly; this is however not supported by the data as the mean waiting time is rather high. DHS (2018) suggests it was an overcapacity in the LNG marked early 2018. These periods of wait could following be because the vessels are anchored and waiting for their next cargo, and not because of port restrictions. It was also expected that bulk ships had a higher waiting time than the case for the other two segments; the result supports this. We can also observe the waiting time for the bulk segment has a large standard deviation. This could be because of, as discussed in chapter 3.6, some bulk operators tend to have priority in port. The higher waiting time for the bulk could also be because the vessels are waiting to pick up cargo on a spot contract if they have arrived early at the port.

# Chapter 7

## Discussion

### 7.1 Data Preparation

From chapter 4.1 we know that the Norwegian Coastal Administration possess four AIS receiving satellites. All these have been in operation during the whole timeframe of this thesis. The size of the received database is presented in Table 6.1, and even if this data is of newer date, the message frequency is significantly lower than the transmission intervals as presented in Table 4.1. Message Type 1 made up 87% of the database, but as the average reporting interval in the database was lower than expected, we saw the benefit of combining all dynamic message types into one, and following increasing the data's resolution as much as possible.

Filtering based on ship lists, instead of the heuristics procedure as presented by Smestad (2015), was selected to obtain a representative selection of ships. An important reason why this procedure was chosen was that this lists also include a vessel's design parameters, such as a design draught and speed. The heuristics method had a respectable accuracy of at least 90% for the segments used in this thesis, and could following have been used for analysis without the need for vessels' design parameters.

When investigating the yearly message count from each distinct vessel, we saw considerable variation in the number of messages received. Even if vessels built during the timeframe of this analysis was filtered out, the message frequency was low for some ships. Figure 6.1 present the message count from each vessel, where some had a yearly count of as low as 541. It was observed a slight trend that bulk ships had more received messages, however, the ships in this segment tended to be of newer date than for the other two segments, regardless of the newest ships were selected for all three segments. The impact of filtering out the 200 vessels with the highest message count was significant, resulting in that the final database included at least 33 000 messages from each vessel, or an average of 3.7 messages per hour per ship.

As suggested by Smestad (2019), indexing could have a considerable impact on computational speed when querying a database. The effect was explored and presented in Figure 5.3,

and found to be significant. One drawback with indexing is that the size of the database also increases. Following it was necessary only to index variable frequently used in queries, to keep the database size manageable. The variables selected in this thesis was the ships MMSI number and unixtime; this allowed for fast querying without increasing the size of the database to an unmanageable point.

The final database was created by filtering out vessels found in ship lists. With a background in this methodology, there was as expected no erroneous MMSI number in the database. This methodology is a quick way of obtaining AIS data of interest, and at the same time, skip several needed validation steps. One possible error involving MMSI number is remaining, that if an MMSI number is related to multiple IMO numbers. This could happen e.g. if a vessel changes ownership. Only a few cases of this were found, as the timeframe of the analyses was limited to one year. All messages related to these MMSI numbers were removed from the database. The removal is vital in comparative analysis to ensure no vessels from other segments is included in the study.

Many potential sources of error in the AIS messages were presented in the literature study. A validation process was carried out to address these possible erroneous messages. The result of this process is presented table 6.2, and suggests that with only simple filtration, numerous erroneous messages could be filtered out. It also became clear that the LNG database included significant more erroneous messages than the other two segments. Further investigations did not answer why this was the case, as the same errors were found for many distinct vessels within this database. Regardless of it, these messages were removed, and following have no impact on the operational patterns.

## 7.2 Data Enrichment

From the literature study in Chapter 2, it became clear that many articles suggest that the manually inputted navigational status included in the AIS signal is prone to errors. A substantial part of this thesis has been to develop and carry out a data enrichment process that gives each AIS message a new and more reliable operational status. The goal of the new operational status is to distinguish between a vessel's different operational states more accurately and use it in operational studies.

### 7.2.1 Port Locations

A vital part of the methodology for establishing a new operational status is to have accurate port locations. With limited accuracy of the port locations obtained from Sea-web, in combination with that a port's operational area tend to evolve, as suggested by Millefiori et al. (2016), was the background for investigating if port locations could be determined using AIS data alone. Chapter 5.2.1 presents a clustering method used to locate port locations without

the need of external data. From table 6.3, we saw this methodology also had another potential benefit. The port list established with clustering contains only the port visited by the vessel subject to analyses. Even if Sea-web Ports include the possibility of filtering on port facilities, the port list obtained from clustering was significantly shorter, and would following have a lower computational cost when used to generate a new operational status.

The accuracy of the ports found using the clustering method was evaluated and compared to the ones found in the database, where the clustering method had promising results. With a smaller distance for up to 98% of all messages for the different segments, confirmed that the port location derived from AIS data alone, was more accurately positioned than the ones in the database. However, the method failed to locate some ports, and the reason is most likely as suggested by Seif (2018), that density-based clustering could struggle to find clusters with different densities. This could especially be the case for messages that have visited a terminal close to a more frequently visited terminal, and is following labelled as noise points.

Sea-web Ports was selected as the source for port location even if the clustering method had promising results. The main reason for this was that the clustering method, as discussed earlier, failed to locate some port. The process for determination of operational status relies on having all port location to give accurate results. However, the result from the clustering was so promising that with more advanced machine learning methodologies, all port location could possibly be located with a similar data foundation. With more frequent AIS messages or even land-based AIS data, the presented clustering methodology should locate all ports.

## 7.2.2 Operational Status

When investigating the magnitude of messages with obviously wrong statuses it was found, as suggested by the literature study in chapter 2, that a significant number of messages contains an erroneous navigational status. This proportion was even more significant for messages with speed below 5 knots. For operational patterns, it is especially important that the operational status is correct at low speeds to distinguish whether the vessel is in port, waiting outside a port or navigating.

Messages with other navigational statuses than expected were also found and further investigated. Common in all three segments was that mainly two statuses were dominating. That was the statuses: *undefined* and *under way sailing*. The reason for many vessels had status equal to underway sailing could be two-fold. First is that some crew could utilise this status if the vessel is drifting while waiting to enter a port. Secondly, it could be a simple misunderstanding by the vessel's crew, as a ship is often referred to as sailing even if it uses engines as propulsion, while the included navigational status distinguishes between if a vessel is using sail or engine as propulsion. It is the latter that is most likely, as most vessels with this status had a speed above 10 knots.

The method for establishing new operational statuses rely on port locations. As discussed earlier, the Swa-web port database was selected for port locations. The port locations'

accuracy was somewhat limited in this database and following the range from where a ship could be categorised as in port was set to 1.5 kilometres. With this range, some vessels that were waiting to enter a port could be categorised as in port. The waiting distance was investigated, as is presented in Table 6.7. The results suggest that few vessels tend to wait outside a port closer than the range of 1.5 kilometres. This result, in combination with that the new operational status also take the old navigational status into account, the erroneous labelling will only happen for vessels with wrong navigational status and is waiting within the suggested range. Following this will be a neglectable, as it will occur only effect a few messages and only result in an extended port visit.

The range of the distance of where a vessel could be categorised as waiting was also determined. This range was investigated by exploring the distances a vessel tends to lay at anchor outside a port, resulting in a distance of 50 kilometres should capture most waiting ships. This distance could be affected by among other local geographics. One potential problem is that the new operational statuses also try to include vessels waiting by drifting and that this distance could be different than the anchorage distance. The upper limit was selected to capture as many vessels waiting as possible.

The LNG segment had a number of vessels waiting with a distance that was significantly higher than the normal for the other segments. As mentioned in chapter 5.2.2, this could possibly be because of the low utilisation rate in the first quarter of 2018 as suggested by IGU (2019), and that ship was at anchor waiting for cargo. Another reason could be that LNG vessels sometimes use ship-to-ship transfers, and that the navigational statuses used for this were at anchor instead or moored.

The final algorithm was run on all three segments, resulting in that a significant number of messages got a different operational state than the one included in the AIS messages. By visually inspecting the statuses around port areas, the methodology gave more accurate results than the included navigational status, especially as the new operational status was able to capture many more vessels waiting outside a port.

When using the operational status to distinguish between different legs, port visits, and time of waiting, even the shorter legs between an anchorage and a port visit would be categorised as a distinct leg. However, this could be addressed in the operational analyses by removing legs with a short sailing distance.

### 7.3 Operational Patterns

The extensive data preparation and enrichment process presented this thesis have simplified the process of analysing operational patterns using AIS data. With access to a more reliable operational status, different parts of a vessel's voyage could be distinguished by simple procedures and further analysed.

The draught ratio gave accurate results for distinguishing between the laden voyages and the



ones in ballast. However, for vessels that have a weight critical design, such as bulk vessels, we observe a higher difference in draught ratio for the two different loading conditions than for vessels with a volume critical design, such as LNG vessels. Hence, the draught ratio could be used when distinguishing between voyages in ballast and laden, but it is essential to investigate the distribution for the specific segment analysed to ensure correct loading condition.

With the legs distinguished for each loading condition, it was found support for the theory made by Assmann et al. (2015) that speed optimisation is more used on trips in ballast. We observed a higher variance in sailing speed for both bulk and LNG vessels. However, while the speed for LNG vessels was lower, the speed for bulk vessel tended to be higher for voyages in ballast. A reason could be that some bulk vessels could potentially increase the speed if they need to reach a new spot cargo. As the operator in the LNG segments also controls the shipment, the operator has more control of the operation and could easier adapt slow seaming on backhaul trips, and why we observe a lower speed for voyages in ballast for this segment.

By also analysing sailing distances for the two different loading conditions, it was found that bulk ships have a significant lower sailing distance in ballast condition compared to when laden. This was expected due to bulk ships' mode of operation, where in tramp trade the vessels are transporting cargo on spot contracts, and following want to find new spot contracts close to the destination of the last. Interestingly, we also found a trend that LNG vessels also had shorter legs in ballast. This could be because some industrial operators also accept some spot cargoes for improving utilisation of their fleet.

The need for voyages without any paying cargo aboard is not a concern in liner shipping. However, due to an imbalance in trade for different regions, a container vessel often has a high load factor for revenue-generating containers in one direction, and a lower in the opposite direction due to the need of container redistribution. It was investigated if this could be seen in operational patterns. The finding was that it was no difference in sailing speed as expected. However, there was found a trend for lower draught ratio in the direction of container redistribution. The draught ratio did not fall in line with the actual trading differences when comparing two trade routes. This fall in line with the findings of Adland et al. (2017) that a vessel's draught ratio alone could only estimate a vessels payload.

The port operation was also compared for the different segments. The result suggested that the container vessel had the most efficient port visits, and that the bulk segment spends significantly more time in port than both LNG and container. It was also found a significant standard deviation in the time spent in port for LNG vessels. This could be because of the low utilisation rate the fleet had the first quarter of 2018, and that the vessels were lying in a port waiting for cargo. Another reason could be, as earlier discussed, the potential need of a cool down of the LNG tanks if the heel has evaporated. In this case, the ship will spend significantly more time in port, as the vessel's tanks need to be cooled down before it can be loaded again.

The new operational status also captures vessels waiting to enter a port, and following waiting

times for the different segments could be compared. The result is as expected the container vessel has the most efficient port operations. However, many container vessels seem to have a short wait outside a port due to many operators tend to arrive before their scheduled port time to avoid potential fines. As expected, bulk vessels spend significantly more time waiting than the case for the other segment. The standard deviation was also rather high for this segment, something that is strengthening the assumption that some bulk operators have priority deals with some ports, and following spends less time waiting in some cases. For the LNG segment, it was expected that the vessels could enter a terminal directly without waiting as the operator also control the terminal. However, it was found that LNG vessels spend more time waiting than Container vessels.

# Chapter 8

## Conclusion

### 8.1 Concluding Remarks

The objective of this thesis has been twofold. First, a data enrichment process has been established that finds a vessel's actual operational status, as the included navigational status is of limited reliability. The presented method utilises the port locations, in combination with the vessels' speed, to determine a new operational status. Secondly, operational patterns from three different modes of operation are investigated using AIS data. The improved operational status was essential when analysing operational patterns.

When investigating the reliability of the included navigational status in the AIS message, it was found a significant number of obvious erroneous statuses, and following that operational analyses could clearly benefit for a more reliable operational status. It was investigated whether the port location could be found directly from the AIS messages. The clustering method used had promising results, but the data foundation was not sufficient, and following the method failed to locate a few ports. With more frequent data this method should give accurate results.

The presented method for establishing a new operation status gave excellent results, and with visual inspections of numerous port locations, the method was validated. The new operational status manages both to capture many more vessels waiting to enter a port, as well as more vessels in port.

When investigating the operational patterns for the three different modes of operation, the new operational status plays a vital role. The result supports most of the operational characteristics as expected in advance. Where more specifically, it was found support for container redistribution in liner operation, and that the voyages in ballast are affected by the mode of operation. For port operations, the result suggests as expected that the mode of operation affect both waiting times and time spent in port.

In conclusion, the extensive data preparation and enrichment process presented in this thesis

have simplified the process for analysing operational patterns using AIS data. Furthermore, it was found support for that the mode operation affects all aspect of a vessel's operation.

## 8.2 Recommendations for Further Work

Further research within the use of AIS data in operational analytics is recommended. The thesis has presented the need for a new navigational status quantitatively and how operational analyses could benefit from a more reliable operational status. The presented methodology produced promising results. However, some further developments and validations are recommended.

The thesis has investigated if port locations could be determined from the AIS data alone, and with access to more frequent data, this method could be used to locate all port of interest. This would especially be beneficial for segments that utilize ship-to-ship transfers, as these port locations not are found in port databases.

The new operational status includes vessels that are waiting to enter a port. Even if the new status also captures vessels drifting while waiting, and not only waiting at anchor, it is recommended further investigation to ensure all waiting vessels are captured.

Other investigations concerning the operational patterns should be developed. When investigating a vessel's waiting time outside a port, the sailing speed prior to the waiting period should be included. Following the correlation between the waiting time and speed could be investigated. This could determine if it is any potential of implementing slow steaming.

It is also recommended to conduct studies with a longer time frame. Following, it could be investigated if market-related factors could have an impact on the different operations of a vessel so that future operations could be adapted as market changes.

From a ship operator's perspective, more specific operational patterns in the particular segment could be investigated. With these operational patterns, both for the segment in general, and for their own fleet, the performance of their vessels could be evaluated against the competitors, and following find possible ways of optimizing performance.

# Bibliography

- Adland, R. & Jia, H. (2018), ‘Dynamic speed choice in bulk shipping’, *Maritime Economics & Logistics* **20**(2), 253–266.
- Adland, R., Jia, H. & Strandenes, S. P. (2017), ‘Are ais-based trade volume estimates reliable? the case of crude oil exports’, *Maritime Policy & Management* **44**(5), 657–665.
- Adland, R. O. & Jia, H. (2016), Vessel speed analytics using satellite-based ship position data, in ‘2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)’, pp. 1299–1303.
- ArcGIS (2018), ‘Density-based clustering’, <http://pro.arcgis.com/en/pro-app/tool-reference/spatial-statistics/densitybasedclustering.htm> (Accessed: 04.02.19).
- Assmann, L. M., Andersson, J. & Eskeland, G. S. (2015), ‘Missing in action? speed optimization and slow steaming in maritime shipping’, *Speed Optimization and Slow Steaming in Maritime Shipping (March 12, 2015)*. NHH Dept. of Business and Management Science Discussion Paper (2015/13).
- Ball, H. (2012), *Custom satellite ais for dummies*, John Wiley & Sons Inc.
- Branch, A. E. (2014), *Branch’s elements of shipping*, 9th edn, Routledge, London.
- Chi, H., Pedrielli, G., Kister, T., Ng, S. & Bressan, S. (2016), ‘An ais-based framework for real time monitoring of vessels efficiency’, *IEEM 2015 - 2015 IEEE International Conference on Industrial Engineering and Engineering Management 2016-January*, 1218–1222.
- Christiansen, M., Fagerholt, K. & Ronen, D. (2004), ‘Ship routing and scheduling: Status and perspectives’, *Transportation science* **38**(1), 1–18.
- DHS (2018), ‘Shipping maket review’, *Danish Ship Fincance* .
- Dobrota, D., Lalic, B. & Komar, I. (2013), ‘Problem of boil - off in lng supply chain’, *University of Split, Faculty of Maritime Studies* .
- Eriksen, T., Skauen, A. N., Narheim, B., Hellenen, Ø., Olsen, Ø. & Olsen, R. B. (2010), Tracking ship traffic with space-based ais: Experience gained in first months of operations, in ‘Waterside Security Conference (WSS), 2010 International’, IEEE, pp. 1–8.
- exactEarth (2015), ‘Saatellite ais’, *exactEarth Technical White Paper* .

- Fagerholt, K. (2018), 'Introduction to ship routing and scheduling', *Lecture Note: Fleet Scheduling and supply chains* .
- Fiorini, M., Capata, A. & Bloisi, D. D. (2016), 'Ais data visualization for maritime spatial planning (msp)', *International Journal of e-Navigation and Maritime Economy* **5**, 45 – 60.
- Gao, X., Makino, H. & Furusho, M. (2016), 'Ship behavior analysis for real operating of container ships using ais data', *TransNav: International Journal on Marine Navigation and Safety of Sea Transportation* **10**(2), 213–220.
- Goldsworthy, L. & Goldsworthy, B. (2015), 'Modelling of ship engine exhaust emissions in ports and extensive coastal waters based on terrestrial ais data – an australian case study', *Environmental Modelling & Software* **63**, 45 – 60.
- Grammenos, C. T. (2010), *The Handbook of maritime economics and business*, The Grammenos library, 2nd ed. edn, Lloyd's List, London.
- Høyve, G. K., Eriksen, T., Meland, B. J. & Narheim, B. T. (2008), 'Space-based ais for global maritime traffic monitoring', *Acta Astronautica* **62**(2), 240 – 245.
- IGU (2019), '2019 world lng report', *Internatioanl Gass Union* .
- IHS (2019a), 'Sea-web ships', <https://maritime.ihs.com/Areas/Seaweb> (Accessed:20.05.19).
- IHS (2019b), 'World fleet analysis', [https://maritime.ihs.com/Seaweb/Analysis/Ships? control=shipWorldFleetAnalysis](https://maritime.ihs.com/Seaweb/Analysis/Ships?control=shipWorldFleetAnalysis) (Accessed: 04.03.19).
- IMO (2014), 'Safety of navigation', *International Convention for the Safety of Life at Sea (SOLAS)* (Chapter V, Regulation 19).
- IMO (2015), 'Revised guidelines for the onboard operational use of shipborne automatic identification systems (ais)', (Resolution A.1106(29)).
- IMO (2016), 'Module 3 - from management to operations', *IMO TTT Coure on Energy Efficient Ship Operations* .
- IMO (2018), 'Regulations on energy efficiency for ships', *MARPOOL Annex IV, Regulations for the Prevention of Air Pollution from Ships (2018 amendment)* .
- IMO (2019), 'Imo identification number schemes', <http://www.imo.org/en/OurWork/MSAS/Pages/IMO-identification-number-scheme.aspx> (Accessed 25.04.19).
- ITF (2017), 'Itf transport outlook 2017 (international transport forum)', <https://www.oecd-ilibrary.org/content/publication/9789282108000-en> (Accessed: 10.02.19), 224.
- ITU (2014), 'Technical characteristics for an automatic identification system using time-division multiple access in the vhf maritime mobile band', *ITU-R M. 1371.5* .
- ITU (2015), 'Assignment and use of identities in the maritime mobile service', *ITU-R M.585-7* .

- Jafarzadeh, S. & Schjøberg, I. (2018), 'Operational profiles of ships in norwegian waters: An activity-based approach to assess the benefits of hybrid and electric propulsion', *Transportation Research Part D: Transport and Environment* **65**, 500 – 523.
- Jia, H., Lampe, O. D., Solteszova, V. & Strandenes, S. P. (2017), 'An automatic algorithm for generating seaborne transport pattern maps based on ais', *Maritime Economics & Logistics* **19**(4), 619–630.
- Jia, H., Prakash, V. & Smith, T. (2019), 'Estimating vessel payloads in bulk shipping using ais data', *International Journal of Shipping and Transport Logistics* **11**, 25.
- Lane, B. C. (2006), 'Ais parser sdk v1.10', Available at: <https://github.com/bcl/aisparser/>.
- Leonhardsen, J. H. (2017), Estimation of fuel savings from rapidly reconfigurable bulbous bows, Master's thesis.
- Levander, K. (2012), 'System based ship design', *SeaKey Naval Architecture*.
- MarineTraffic (2018), 'What is the significance of the ais shiptype number?', <https://help.marinetraffic.com/hc/en-us/articles/205579997-What-is-the-significance-of-the-AIS-Shiptype-number-> (Accessed: 03.11.18).
- Metcalf, K., Bréheret, N., Chauvet, E., Collins, T., Curran, B. K., Parnell, R. J., Turner, R. A., Witt, M. J. & Godley, B. J. (2018), 'Using satellite ais to improve our understanding of shipping and fill gaps in ocean observation data to support marine spatial planning', *Journal of Applied Ecology* **55**(4), 1834–1845.
- Millefiori, L. M., Zisis, D., Cazzanti, L. & Arcieri, G. (2016), 'A distributed approach to estimating sea port operational regions from lots of ais data', *2016 IEEE International Conference on Big Data (Big Data)* pp. 1627–1632.
- Norwegian Coastal Administration, N. (2014), 'Forskrift om navigasjon og navigasjonshjelpemidler for skip og flyttbare innretninger', *Forskrift om navigasjonshjelpemidler for skip mv.*
- Norwegian Coastal Administration, N. (2017), 'Nye ais-satellitter skutt opp', <https://www.kystverket.no/Nyheter/2017/juli/nye-ais-satellitter-skutt-opp-i-dag/> (Accessed : 12.01.18).
- Næss, P. A. (2018), Investigation of multivariate freight rate prediction using machine learning and ais data, Master's thesis.
- Rhodes, B. J., Bomberger, N. A., Seibert, M. & Waxman, A. M. (2005), Maritime situation monitoring and awareness using learning mechanisms, in 'MILCOM 2005 - 2005 IEEE Military Communications Conference', pp. 646–652 Vol. 1.
- Schill, N. & Browning, M. (2015), The impact of satellite ais to the environmental challenges of modern shipping, in 'Clean Mobility and Intelligent Transport Systems', Institution of Engineering and Technology, pp. 295–309.

- Seif, G. (2018), ‘The 5 clustering algorithms data scientists need to know’, <https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68>, (Accessed : 02.12.18).
- Skauen, A., Hellenen, Ø., Olsen, Ø. & Olsen, R. (2013), ‘Operator and user perspective of fractionated ais satellite systems’, *Norwegian Defence Research Establishment (FFI)* .
- Smestad, B. B. (2015), A study of satellite ais data and the global ship traffic through the singapore strait, Master’s thesis.
- Smestad, B. B. (2019), ‘Preparing a database for analysis’, <https://www.researchgate.net/project/AIS-data-for-maritime-research> (Accessed: 28.01.19).
- Smestad, B. B., Asbjørnslett, B. E. & Rødseth, Ø. J. (2017), ‘Expanding the possibilities of ais data with heuristics’, *TransNav: International Journal on Marine Navigation and Safety of Sea Transportation* .
- Smith, T. W. P., Jalkanen, J. P., Anderson, B. A., Corbett, J. J., Faber, J., Hanayama, S., O’keeffe, E., Parker, S., Johanasson, L., Aldous, L. et al. (2015), ‘Third imo grennhouse gas study’, *IMO* .
- Spiliopoulos, G., Zissis, D. & Chatzikokolakis, K. (2018), A big data driven approach to extracting global trade patterns, in C. Doukeridis, G. A. Vouros, Q. Qu & S. Wang, eds, ‘Mobility Analytics for Spatio-Temporal and Social Data’, Springer International Publishing, Cham, pp. 109–121.
- Stopford, M. (2009), *Maritime economics*, 3rd edn, Routledge, London.
- Tixerant, M. L., Guyader, D. L., Gourmelon, F. & Queffelec, B. (2018), ‘How can automatic identification system (ais) data be used for maritime spatial planning?’, *Ocean & Coastal Management* **166**, 18 – 30. Maritime Spatial Planning, Ecosystem Approach and Supporting Information Systems (MapSIS 2017).
- Tu, E., Zhang, G., Rachmawati, L., Rajabally, E. & Huang, G.-B. (2018), ‘Exploiting ais data for intelligent maritime navigation: A comprehensive survey from data to methodology’, *Intelligent Transportation Systems, IEEE Transactions on* **19**(5), 1559–1582.
- UNCTAD (2018), ‘Review of maritime transport 2018’, *United Nations Conference on Trade and Development* .
- Wu, L., Xu, Y., Wang, Q., Wang, F. & Xu, Z. (2017), ‘Mapping global shipping density from ais data’, *Journal of Navigation* **70**(1), 67–81.



# Appendices

# Appendix A

## AIS Message Contents

All AIS information sent by ships, as presented in IMO (2015)

Table A.1: Information included in static AIS messages

Information item	Information generation, type and quality of information
<b>Static</b>	
MMSI	Set on installation Note that this might need amending if the ship changes ownership
Call sign and name	Set on installation Note that this might need amending if the ship changes ownership
IMO Number	Set on installation
Length and beam	Set on installation or if changed
Type of ship	Select from pre-installed list
Location of electronic position fixing system (EPFS) antenna	Set on installation or may be changed for bi-directional vessels or those fitted with multiple antennas

Table A.2: Information included in dynamic AIS messages

Information item	Information generation, type and quality of information
<b>Dynamic</b>	
Ship's position with accuracy indication and integrity status	Automatically updated from the position sensor connected to AIS. The accuracy indication is approximately 10 m.
Position Time stamp in UTC	Automatically updated from ship's main position sensor connected to AIS
Course over ground (COG)	Automatically updated from ship's main position sensor connected to AIS, if that sensor calculates COG. This information might not be available
Speed over ground (SOG)	Automatically updated from the position sensor connected to AIS. This information might not be available
Heading	Automatically updated from the ship's heading sensor connected to AIS
Navigational status	<p>Navigational status information has to be manually entered by the OOW and changed as necessary, for example:</p> <ul style="list-style-type: none"> <li>- underway by engines</li> <li>- at anchor</li> <li>- not under command (NUC)</li> <li>- restricted in ability to manoeuvre (RIATM) - moored</li> <li>- constrained by draught</li> <li>- aground</li> <li>- engaged in fishing</li> <li>- underway by sail</li> </ul> <p>In practice, since all these relate to the COLREGs, any change that is needed could be undertaken at the same time that the lights or shapes were changed</p>
Rate of turn (ROT)	Automatically updated from the ship's ROT sensor or derived from the gyro. This information might not be available

Table A.3: Information included in voyage related AIS messages

<b>Information item</b>	<b>Information generation, type and quality of information</b>
<b>Voyage-related</b>	
Ship's draught	To be manually entered at the start of the voyage using the maximum draft for the voyage and amended as required (e.g. – result of de-ballasting prior to port entry)
Hazardous cargo (type)	To be manually entered at the start of the voyage confirming whether or not hazardous cargo is being carried, namely: <ul style="list-style-type: none"> <li>- DG (Dangerous goods)</li> <li>- HS (Harmful substances)</li> <li>- MP (Marine pollutants)</li> </ul> Indications of quantities are not required
Destination and ETA	To be manually entered at the start of the voyage and kept up to date as necessary
Route plan (waypoints)	To be manually entered at the start of the voyage, at the discretion of the master, and updated when required

Table A.4: Information included in safety related AIS messages

<b>Information item</b>	<b>Information generation, type and quality of information</b>
<b>Safety-related</b>	
Short safety-related messages	Free format short text messages would be manually entered, addressed either a specific addressee or broadcast to all ships and shore stations

# Appendix B

## Code

### B.1 database.py

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Thu Mar 28 12:30:22 2019
5
6 @author: havard
7 """
8
9 #Import Packages
10 import sqlite3 as sql
11 from sqlite3 import Error
12 import time
13 import datetime
14 import calendar
15 import numpy as np
16 import matplotlib.pyplot as plt
17 import pandas as pd
18 import readCSV as readCSV
19 import db as db
20
21 #Taking time
22 tic = time.time()
23
24 #Specifying path from where to read
25 #PATH OF THE ORIGINAL DATABASE
26
27 #path = '/Volumes/Transcend/PrelimContainer.db'
28
29 path = '/Users/havard/Desktop/Master/origianlDB/haavardDBnew.db'
30
31
32 #TABLE THAT SHOULD BE EXTRACTED FROM THE ORIGINAL DATABASE
```

```

33 tables = [ 'messagetype1', 'messagetype2', 'messagetype3', 'messagetype5' ]
34
35
36 #PATH AND NAME FOR THE NEW DATABASE
37 #newDbPathAndName = '/Users/havard/Desktop/Master/finalDB/container.db'
38 #newDbPathAndName = '/Users/havard/Desktop/Master/finalDB/bulk.db'
39 newDbPathAndName = '/Users/havard/Desktop/Master/finalDB/lng.db'
40 table = 'messagetype5'
41
42 #Import Ship List from CSV file and create list with MMSI numbers
43 shipList = readCSV.importShips(3,2018)
44 shipList = shipList.MMSI.values
45
46 #####
47 #           Exploration of the db           #
48 #####
49 #ORIGINAL DATABASE
50
51 #Find first and last date, and count of messages for each msgType
52 exploreDatabase = 0
53 if exploreDatabase == 1:
54     for tableNr in tables:
55         print(tableNr)
56         con = sql.connect(path)
57         unixtime = con.execute("SELECT COUNT(unixtime), min(unixtime),\
58                               max(unixtime) FROM %s" % (tableNr)).fetchall()
59         print('\nNumber of messages: ', unixtime[0][0])
60         print('First date: ', datetime.datetime.utcfromtimestamp(unixtime
61                               [0][1]))
62         print('Last date: ', datetime.datetime.utcfromtimestamp(unixtime
63                               [0][2]))
64         con.close
65
66 #Count Number in shiptype
67 shipTypeCount = 0
68 if shipTypeCount ==1:
69     query = "SELECT count(distinct userid) as NumberOfShips, \
70            cast(ship_type/10 as int)*10 AS shiptype FROM %s \
71            group by shiptype order by count(*)" %(table)
72     con = sql.connect(path)
73     with con:
74         ais_dataframe = pd.read_sql_query(query,con)
75     con.close()
76     print(ais_dataframe.tail(20))
77
78 #Count number of different ships
79 shipTypeCount1 = 0
80 if shipTypeCount1 ==1:
81     query = "SELECT DISTINCT userid as UserID FROM %s" %(table)
82     con = sql.connect(path)
83     with con:
84         ais_dataframe = pd.read_sql_query(query,con)

```

```

83     con.close()
84     print(len(ais_dataframe))
85
86
87 #FINAL DATABASES
88 #Count Messges
89 countMessages = 0
90 if countMessages == 1:
91     con = sql.connect(newDbPathAndName)
92     countdyn = con.execute("SELECT COUNT(*) FROM dyn").fetchone()
93     countstat = con.execute("SELECT COUNT(*) FROM stat").fetchone()
94     print('Number of dynamic: ', countdyn[0])
95     print('Number of static: ', countstat[0])
96     con.close()
97
98 #Number of unique vessels
99 countUniqueShips = 0
100 if countUniqueShips ==1:
101     query = "SELECT DISTINCT userid as UserID FROM %s" %('dyn')
102     con = sql.connect(newDbPathAndName)
103     with con:
104         ais_dataframe = pd.read_sql_query(query, con)
105     con.close()
106     print(len(ais_dataframe))
107
108
109 #Message Frequency Per ship
110 messageCountShip = 0
111 if messageCountShip ==1:
112     query = "SELECT userid , count(*) FROM %s GROUP BY userid ORDER BY count(*)"
113     " %('dyn')
114     con = sql.connect(newDbPathAndName)
115     with con:
116         ais_dataframe = pd.read_sql_query(query, con)
117     con.close()
118     print(len(ais_dataframe))
119     containerShipCount = ais_dataframe
120     #bulkShipCount = ais_dataframe
121     #lngShipCount = ais_dataframe
122
123 #####
124 #           Creating New Database           #
125 #####
126
127 #Create New Database with Ships from ShipList
128 createNewDatabase = 0
129 if createNewDatabase == 1:
130     #Initiate Database
131     con = sql.connect(newDbPathAndName)
132     con.execute("CREATE TABLE IF NOT EXISTS ShipList(userid INT)")
133     con.execute("CREATE TABLE IF NOT EXISTS dyn(unixtime INT,\
134     cog INT,latitude INT, longitude INT, nav_status INT, sog INT, userid INT,\

```

```

134     msgType INT)”)
135     con.execute(“CREATE TABLE IF NOT EXISTS stat(unixtime INT,\
136     dest string, draught INT, eta INT, name text, ship_type INT, userid INT,\
137     imo INT,msgType INT)”)
138
139     #Insert Ship List into own table only if list is empty
140     count = con.execute(“SELECT COUNT(*) FROM ShipList”).fetchone()[0]
141     if count == 0:
142         for ship in shipList:
143             con.execute(“INSERT INTO ShipList(userid) VALUES(?)”, (ship,))
144     con.commit()
145     con.close()
146
147     #Connect the new database with the total
148     con = sql.connect(path)
149     query = “ATTACH ‘%s’ AS new;” % newDbPathAndName
150     con.execute(query)
151
152     # Insert the ships from ship list into new databse
153     if int(table[-1]) == 5:
154         #Static message
155         query = “INSERT INTO new.stat SELECT unixtime,destination, draught,
156         eta_day,name,\
157         shiptype,mmsi,imo_num,%s FROM %s WHERE (mmsi in (SELECT userid FROM\
158         new.ShipList)) ORDER BY UNIXTIME ASC” %(int(table[-1]),table)
159     else:
160         #Dynamic message
161         query = “INSERT INTO new.dyn SELECT unixtime,cog,latitude,\
162         longitude,nav_status,sog,mmsi,%s FROM %s WHERE (mmsi in \
163         (SELECT userid FROM new.ShipList)) ORDER BY UNIXTIME ASC” %(int(table
164         [-1]),table)
165     con.execute(query)
166     con.commit()
167     con.close()
168
169     #Create Index on userid and unixtime
170     createIndex = 0
171     if createIndex ==1:
172         con = sql.connect(newDbPathAndName)
173         query = “CREATE INDEX userid ON dyn(userid)”
174         con.execute(query)
175         query = “CREATE INDEX unix ON dyn(unixtime)”
176         con.execute(query)
177         query = “CREATE INDEX useridS ON stat(userid)”
178         con.execute(query)
179         query = “CREATE INDEX unixS ON stat(unixtime)”
180         con.execute(query)
181         con.commit()
182         con.close()
183
184     #####
185     # Only For Generation of Preliminary Database #

```



```

184 #####
185 createPrelimDatabase = 0
186
187 if createPrelimDatabase == 1:
188     path = '/Volumes/Transcend/AIS/S-AISGlobalOriginal.db'
189     table = 'MessageType1'
190     newDbPathAndName = "/Volumes/Transcend/PrelimContainer.db"
191     shipList = readCSV.importShips(1,2015).MMSI.values
192
193     sTime = calendar.timegm(time.strptime('01/1/2015', '%d/%m/%Y'))
194     eTime = calendar.timegm(time.strptime('01/01/2016', '%d/%m/%Y'))
195
196
197 #Create NewDatabase
198 con = sql.connect(newDbPathAndName)
199 con.execute("CREATE TABLE IF NOT EXISTS ShipList(userid INT)")
200 con.execute("CREATE TABLE IF NOT EXISTS dyn(unixtime INT,\
201 cog INT,latitude INT, longitude INT, nav_status INT, sog INT, userid INT,\
202 msgType INT)")
203 con.execute("CREATE TABLE IF NOT EXISTS stat(unixtime INT,\
204 dest string, draught INT, eta INT, name text, ship_type INT, userid INT,\
205 imo INT,msgType INT)")
206
207 #Insert Ship List into own table only if list is empty
208 count = con.execute("SELECT COUNT(*) FROM ShipList").fetchone()[0]
209 if count == 0:
210     for ship in shipList:
211         con.execute("INSERT INTO ShipList(userid) VALUES(?)", (ship,))
212 con.commit()
213 con.close()
214
215 #Connect the new database with the total
216 con = sql.connect(path)
217 query = "ATTACH '%s' AS new;" % newDbPathAndName
218 con.execute(query)
219
220 # Insert the ships from ship list into new database
221 if int(table[-1]) == 5:
222     #Static message
223     query = "INSERT INTO new.stat SELECT unixtime,dest, draught, eta, name,\
224 ship_type,userid,imo,%s FROM %s WHERE (userid in (SELECT userid FROM \
225 new.ShipList)) and unixtime >= %s and unixtime <= %s ORDER BY \
226 UNIXTIME ASC" %(int(table[-1]),table,sTime,eTime)
227 else:
228     #Dynamic message
229     query = "INSERT INTO new.dyn SELECT unixtime,cog,latitude,\
230 longitude,nav_status,sog,userid,%s FROM %s WHERE \
231 (userid in (SELECT userid FROM new.ShipList)) and \
232 unixtime >= %s and unixtime <= %s ORDER BY \
233 UNIXTIME ASC" %(int(table[-1]),table,sTime,eTime)
234 con.execute(query)
235 con.commit()

```

```

236     con.close()
237
238
239 #FOR MEASSURING INDEX EFFICIENCY
240
241 #shipList = ais_dataframe[ais_dataframe.UserID > 23000000]
242 #shipList = shipList[shipList.UserID < 799999999]
243 #shipList = shipList.head(3000).UserID.values
244 #shipList = shipList.astype(np.float64)
245
246 #takeTime = 0
247 #if takeTime == 1:
248 #    tic1 = time.time()
249 #    query = "SELECT distinct userid as UserID FROM %s" %(table)
250 #    con = sql.connect(newDbPathAndName)
251 #    with con:
252 #        ais_dataframe1 = pd.read_sql_query(query,con)
253 #    con.close()
254 #    print(ais_dataframe.size)
255 #    print("Count" , time.time()-tic1)
256
257
258 print('\nTotal time: ',time.time() - tic)

```

## B.2 main.py

```

1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Mon Feb  4 11:29:23 2019
5
6 @author: havard
7 """
8
9 #IMPORTING PACKAGES
10 import time
11 import datetime
12 import calendar
13 import numpy as np
14 import pandas as pd
15 import matplotlib.pyplot as plt
16 plt.rcParams.update({'font.size': 16})
17 import seaborn as sns
18
19 import ais as ais
20 import ais_Plotting as aisP
21 import db as db
22 import locationCheck as loc
23 import readCSV as csv
24 import machineLearning as ml
25

```

```

26 tic = time.time()
27
28 #####
29 #           Data Management           #
30 #####
31 #DEFINING DATABASE PATH
32
33 #dbPath = '/Volumes/Transcend/AIS-Master/AISdatabase.db'
34 dbPathContainer = '/Users/havard/Desktop/Master/finalDB/container.db'
35 dbPathBulk = '/Users/havard/Desktop/Master/finalDB/bulk.db'
36 dbPathLng = '/Users/havard/Desktop/Master/finalDB/lng.db'
37
38 tableDyn = 'dyn'
39 tableStat = 'stat'
40
41
42 #SPESIFING RELEVANT DATA
43 #Relevant Time Window [year ,month ,day]
44 fromTime = calendar.timegm(time.strptime('01/1/2018', '%d/%m/%Y'))
45 toTime = calendar.timegm(time.strptime('01/01/2019', '%d/%m/%Y'))
46
47
48
49
50 #####
51 #           DASHBOARD           #
52 #####
53 DisplayDataBaseStatistics = 0
54
55 #Data Preparation
56 ExtractData = 0
57 ValidateData = 0
58
59 #Data Enrichment
60 FindPorts = 0
61 LableStatus = 0
62 LegNumbering = 0
63
64 #Operational Analytics
65 LableDraughtRatio = 0
66 LableArea = 0
67 OperationalAnalytics = 0
68 PresentResults = 1
69
70
71
72 SpeedAnalysis = 0
73 OperationalAnalysis = 0
74 LegAnalysis = 0
75 PlotData = 0
76 eastWestTrade = 0
77

```

```

78 lableStatus = 0
79 lableArea = 0
80 lableDraughtRatio = 0
81
82 Testing = 0
83
84 legLableHeadingAndArea =0
85
86
87 #####
88 #                Function calls                #
89 #####
90
91 if DisplayDataBaseStatistics == 1:
92     db.DatabaseInfo(dbPathContainer , table)
93     db.DatabaseInfo(dbPathBulk , table)
94
95
96 # DATA PREPARATION
97 #####
98 #Extract infomation form database into dataframes
99 if ExtractData ==1:
100     #Container Vessels
101     vesselCount = db.msgCountPrVessel(dbPathContainer , tableDyn)
102     vessellist = vesselCount.tail(200).userid.values
103     dfContainer1 = db.Extract1(dbPathContainer , tableDyn , fromTime , toTime)
104     dfContainer1 = dfContainer1[dfContainer1.userid.isin(vessellist)]
105     dfContainer5 = db.Extract5(dbPathContainer , tableStat , fromTime , toTime)
106     dfContainer5 = dfContainer5[dfContainer5.userid.isin(vessellist)]
107
108     #Bulk Vessels
109     vesselCount = db.msgCountPrVessel(dbPathBulk , tableDyn)
110     vessellist = vesselCount.tail(200).userid.values
111     dfBulk1 = db.Extract1(dbPathBulk , tableDyn , fromTime , toTime)
112     dfBulk1 = dfBulk1[dfBulk1.userid.isin(vessellist)]
113     dfBulk5 = db.Extract5(dbPathBulk , tableStat , fromTime , toTime)
114     dfBulk5 = dfBulk5[dfBulk5.userid.isin(vessellist)]
115
116     #Container Vessels
117     vesselCount = db.msgCountPrVessel(dbPathLng , tableDyn)
118     vessellist = vesselCount.tail(200).userid.values
119     dfLng1 = db.Extract1(dbPathLng , tableDyn , fromTime , toTime)
120     dfLng1 = dfLng1[dfLng1.userid.isin(vessellist)]
121     dfLng5 = db.Extract5(dbPathLng , tableStat , fromTime , toTime)
122     dfLng5 = dfLng5[dfLng5.userid.isin(vessellist)]
123
124
125 #Validate The Dataframes
126 if ValidateData ==1:
127     #Test that IMO number not change
128     imoTest = 1
129     if imoTest ==1:

```

```

130     #Remove messagees with changing imo number
131     #Container Vessels
132     changesContainer = ais.validateIMO(dfContainer5)
133     dfContainer1 = dfContainer1[~dfContainer1.userid.isin(changesContainer
)]
134     #Bulk vessels
135     changesBulk = ais.validateIMO(dfBulk5)
136     dfBulk1 = dfBulk1[~dfBulk1.userid.isin(changesBulk)]
137     #Lng vessels
138     changesLng = ais.validateIMO(dfLng5)
139     dfLng1 = dfLng1[~dfLng1.userid.isin(changesLng)]
140
141     speedValidate = 1
142     if speedValidate == 1:
143         #Container Vessels
144         dfContainer1 = dfContainer1[(dfContainer1.sog>=0)&(dfContainer1.sog
<35)]
145         #Bulk vessels
146         dfBulk1 = dfBulk1[(dfBulk1.sog>=0)&(dfBulk1.sog<35)]
147         #Lng vessels
148         dfLng1 = dfLng1[(dfLng1.sog>=0)&(dfLng1.sog<35)]
149
150     locValidation = 1
151     if locValidation == 1:
152         #Container Vessels
153         dfContainer1 = dfContainer1[(dfContainer1.longitude>=-180)&\
(dfContainer1.longitude<=180)]
154         dfContainer1 = dfContainer1[(dfContainer1.latitude>=-90)&\
(dfContainer1.latitude<=90)]
155
156         #Bulk vessels
157         dfBulk1 = dfBulk1[(dfBulk1.longitude>=-180)&(dfBulk1.longitude<=180)]
158         dfBulk1 = dfBulk1[(dfBulk1.latitude>=-90)&(dfBulk1.latitude<=90)]
159         #LNG vessels
160         dfLng1 = dfLng1[(dfLng1.longitude>=-180)&(dfLng1.longitude<=180)]
161         dfLng1 = dfLng1[(dfLng1.latitude>=-90)&(dfLng1.latitude<=90)]
162
163
164     #Find Ports
165     if FindPorts == 1:
166
167         #Import Ports from Databse
168         containerPorts = csv.importPorts(1)
169         containerPorts.rename(columns = {'Longitude': 'longitude', \
'Latitude': 'latitude'}, inplace=True)
170
171         bulkPorts = csv.importPorts(2)
172         bulkPorts.rename(columns = {'Longitude': 'longitude', \
'Latitude': 'latitude'}, inplace=True)
173
174         lngPorts = csv.importPorts(3)
175         lngPorts.rename(columns = {'Longitude': 'longitude', \
'Latitude': 'latitude'}, inplace=True)
176
177     #Visualice Ports
178     #aisP.mapScatterLocal(containerPorts['Longitude'], containerPorts['Latitude
'])

```

```

179
180 #Find Ports using Clustering
181 clusterPorts = 0
182 if clusterPorts == 1:
183     #Container Ports
184     df1 = dfContainer1
185     containerPortsCluster = ml.clusterDBSCAN(df1[(df1['sog'] < 3) & \
186         (df1['nav_status'] == 5)]['longitude'], df1[(df1['sog']
] < 3) \
187         & (df1['nav_status'] == 5)]['latitude'], 1)
188     print('here')
189     #Bulk Ports
190     df1 = dfBulk1
191     bulkPortsCluster = ml.clusterDBSCAN(df1[(df1['sog'] < 3) & \
192         (df1['nav_status'] == 5)]['longitude'], df1[(df1['sog']
] < 3) \
193         & (df1['nav_status'] == 5)]['latitude'], 1)
194     #LNG Ports
195     df1 = dfLng1
196     lngPortsCluster = ml.clusterDBSCAN(df1[(df1['sog'] < 3) & \
197         (df1['nav_status'] == 5)]['longitude'], df1[(df1['sog']
] < 3) \
198         & (df1['nav_status'] == 5)]['latitude'], 1)
199
200 #Validate clustered ports against databasePorts
201 validateClusteredPorts = 0
202 if validateClusteredPorts == 1:
203     #Calculate distance from point to closest dbPort
204     def closestPort(row, df):
205         distances = []
206         for port in df.itertuples():
207             distances.append(ais.haversine(row.longitude, row.latitude, \
208                 port.longitude, port.latitude))
209         dist = min(distances)
210         return dist
211
212 #Distance from clustered port to closes port in portDatabase
213 #For finding ports not identified in clustering
214 containerPortsCluster['minDistToPort'] = \
215     containerPortsCluster.apply(closestPort, args=(containerPorts, ), axis=1)
216
217 bulkPortsCluster['minDistToPort'] = \
218     bulkPortsCluster.apply(closestPort, args=(bulkPorts, ), axis=1)
219
220 lngPortsCluster['minDistToPort'] = \
221     lngPortsCluster.apply(closestPort, args=(lngPorts, ), axis=1)
222
223
224 #Accuracy of Port Locations, clustered vs port list
225 #Check points with sog = 0 and status == Moored
226 #Container
227 dfTContainer = dfContainer1[(dfContainer1.sog == 0) & \

```

```

228         (dfContainer1.nav_status==5)].copy()
229     dfTContainer['closestPortCluster'] = \
230     dfTContainer.apply(closestPort, args=(containerPortsCluster, ), axis=1)
231     dfTContainer['closestPort'] = \
232     dfTContainer.apply(closestPort, args=(containerPorts, ), axis=1)
233
234     #Bulk
235     dfTBulk = dfBulk1[(dfBulk1.sog==0)&(dfBulk1.nav_status==5)].copy()
236     dfTBulk['closestPortCluster'] = \
237     dfTBulk.apply(closestPort, args=(bulkPortsCluster, ), axis=1)
238     dfTBulk['closestPort'] = dfTBulk.apply(closestPort, args=(bulkPorts, ),
axis=1)
239
240     #Lng
241     dfTLng = dfLng1[(dfLng1.sog==0)&(dfLng1.nav_status== 5)].copy()
242     dfTLng['closestPortCluster'] = \
243     dfTLng.apply(closestPort, args=(lngPortsCluster, ), axis=1)
244     dfTLng['closestPort'] = dfTLng.apply(closestPort, args=(lngPorts, ), axis
=1)
245
246 # DATA ENRICHMENT
247 #####
248 #Fucntioncall for labelling of new operational statuses
249 if LableStatus ==1:
250     lableStatuses = 0
251     if lableStatuses == 1:
252         #Container Vessels
253         dfContainer1 = ais.lableStatus(dfContainer1, containerPorts)
254         aisP.mapStatusScatterLocal(dfContainer1)
255         #Bulk Vessels
256         dfBulk1 = ais.lableStatus(dfBulk1, bulkPorts)
257         aisP.mapStatusScatterLocal(dfBulk1)
258         #LNG vessels
259         dfLng1 = ais.lableStatus(dfLng1, lngPorts)
260         aisP.mapStatusScatterLocal(dfLng1)
261
262 #Visualice erroneous messages
263 showErroneous=0
264 if showErroneous == 1:
265     #aisP.mapErroneousStatus(dfBulk1)
266     aisP.mapErroneousStatus(dfContainer1)
267
268 #Explore parameters
269 exploreThreshold = 0
270 if exploreThreshold == 1:
271     def closestPort(row, df):
272         distances = []
273         for port in df.itertuples():
274             distances.append(ais.haversine(row.longitude, row.latitude, \
275                                             port.longitude, port.latitude))
276         return min(distances)
277     #Validation of at anchor/waiting

```

```

278     #Contaienr Vessels
279     dfAContainer = dfContainer1 [(dfContainer1.sog < 3) & \
280                               (dfContainer1.nav_status == 1)].copy()
281     dfAContainer['distToPortAnch'] = dfAContainer.apply(closestPort, \
282               args=(containerPorts, ), axis = 1)
283     #Bulk vessels
284     dfABulk = dfBulk1 [(dfBulk1.sog < 3) & (dfBulk1.nav_status == 1)].copy()
285     dfABulk['distToPortAnch'] = dfABulk.apply(closestPort, args=(bulkPorts
286     ), axis = 1)
287     #Lng vessels
288     dfALng = dfLng1 [(dfLng1.sog < 3) & (dfLng1.nav_status == 1)].copy()
289     dfALng['distToPortAnch'] = dfALng.apply(closestPort, args=(lngPorts, ),
290     axis = 1)
291     exploreChanges = 1
292     if exploreChanges == 1:
293         changesContainer = ais.statusChange(dfContainer1)
294         changesBuk = ais.statusChange(dfBulk1)
295         changesLng = ais.statusChange(dfLng1)
296
297     #Chategorise into distinct legs ,port visits and time of waiting
298     if LegNumbering == 1:
299         dfContainer1 = ais.legNumbering(dfContainer1)
300         dfBulk1 = ais.legNumbering(dfBulk1)
301         dfLng1 = ais.legNumbering(dfLng1)
302
303
304
305     # Operational Analytics
306     #####
307
308     #Lable Draught Ratio in static message
309     if LableDraughtRatio == 1:
310         #Container
311         shipListContainer = csv.importShips(1,2018)
312         shipListContainer = shipListContainer[shipListContainer.MMSI.isin(
313         dfContainer5.userid.unique())]
314         dfContainer5 = ais.lableStatDraughtRatio(dfContainer5, shipListContainer)
315         #Bulk
316         shipListBulk = csv.importShips(2,2018)
317         shipListBulk = shipListBulk[shipListBulk.MMSI.isin(dfBulk5.userid.unique(
318         ))]
319         dfBulk5 = ais.lableStatDraughtRatio(dfBulk5, shipListBulk)
320         #LNG
321         shipListLng = csv.importShips(3,2018)
322         shipListLng = shipListLng[shipListLng.MMSI.isin(dfLng5.userid.unique())]
323         dfLng5 = ais.lableStatDraughtRatio(dfLng5, shipListLng)
324
325     #Lable Area in Dynamic Messages
326     if LableArea == 1:
327         dfContainer1 = loc.lableArea(dfContainer1)

```



```

326     dfBulk1 = loc.lableArea(dfBulk1)
327     dfLng1 = loc.lableArea(dfLng1)
328
329 if OperationalAnalytics == 1:
330     #Define Dataframe for Leg,wait, and port infomation
331     sailContainer = pd.DataFrame()
332     portContainer = pd.DataFrame()
333     waitContainer = pd.DataFrame()
334
335     sailBulk = pd.DataFrame()
336     portBulk = pd.DataFrame()
337     waitBulk = pd.DataFrame()
338
339     sailLng = pd.DataFrame()
340     portLng = pd.DataFrame()
341     waitLng = pd.DataFrame()
342
343     #Calculate sailing ,port and wait time
344     sailContainer , portContainer ,waitContainer = ais.timeAnalytics(
345         dfContainer1\
346         ,sailContainer ,portContainer ,waitContainer
347     )
348     sailBulk , portBulk , waitBulk = ais.timeAnalytics(dfBulk1 ,sailBulk ,\
349         portBulk ,waitBulk)
350     sailLng , portLng , waitLng = ais.timeAnalytics(dfLng1 ,sailLng ,\
351         portLng ,waitLng)
352
353     #Leg Analytics
354     sailContainer = ais.legAnalytics(dfContainer1 ,dfContainer5 ,sailContainer)
355     sailBulk = ais.legAnalytics(dfBulk1 ,dfBulk5 ,sailBulk)
356     sailLng = ais.legAnalytics(dfLng1 ,dfLng5 ,sailLng)
357
358 if PresentResults == 1:
359     c1 = 'steelblue'
360     c2 = 'darkorange'
361     c3 = 'lightseagreen'
362     c11 = 'lightblue'
363     c21 = 'bisque'
364     c31 = 'paleturquoise'
365
366     plt.tick_params(labelsize=20)
367
368     tradePattern = 0
369     if tradePattern == 1:
370         aisP.mapPlotDensity(dfContainer1)
371         aisP.mapPlotDensity(dfBulk1)
372         aisP.mapPlotDensity(dfLng1)
373         aisP.mapPlotDensityType(dfContainer1 ,dfBulk1 ,dfLng1)
374
375     #Explore Draught Ratio Diist , lng and bulk for determanating ballas vs
376     laden
377     draughtRatio = 0

```

```

375     if draughtRatio ==1:
376         #df1 = sailContainer [(sailContainer.maxMsgInt<12)&(sailContainer.area
=='Pac ')]
377         #df1 = sailBulk [(sailBulk.maxMsgInt<12)]
378         df1 = dfLng5
379         plt.figure()
380         bins = np.linspace(0.6, 1.1, 25)
381         plt.hist(df1.dRatio, bins, alpha=1,color=c1,density=False)
382         #plt.legend(loc='upper right')
383         plt.xlabel('Draught Ratio')
384         plt.ylabel('Messages')
385         plt.show()
386
387         df1 = dfBulk5
388         plt.figure()
389         bins = np.linspace(0.4, 1.1, 20)
390         plt.hist(df1.dRatio, bins, alpha=1,color=c1,density=False)
391         # plt.legend(loc='upper right')
392         plt.xlabel('Draught Ratio')
393         plt.ylabel('Messages')
394         plt.show()
395
396     #Bulk Laden vs Ballst Voyages
397     bulkLadenVsBallast = 0
398     if bulkLadenVsBallast ==1:
399         d = [sailBulk [(sailBulk.maxMsgInt<6)&(sailBulk.voyageDist>200)&(
sailBulk.avgSpeed>5)&(sailBulk.dRatio>0.8)].avgSpeed.values,\
400             sailBulk [(sailBulk.maxMsgInt<6)&(sailBulk.voyageDist>200)&(
sailBulk.avgSpeed>5)&(sailBulk.dRatio<0.7)].avgSpeed.values]
401         plt.figure()
402         sns.boxplot(data = d, palette = [c1,c1], boxprops=dict(alpha=1))
403         plt.xticks(plt.xticks()[0],['Laden', 'Ballast'])
404         plt.ylabel('Average Speed')
405         plt.show()
406
407         print('Voyage Length: Bulk in Laden')
408         print(sailBulk [(sailBulk.maxMsgInt<6)&(sailBulk.voyageDist>200)&(
sailBulk.avgSpeed>5)&(sailBulk.dRatio>0.8)].voyageDist.describe())
409         print('Voyage Length: Bulk in Ballast')
410         print(sailBulk [(sailBulk.maxMsgInt<6)&(sailBulk.voyageDist>200)&(
sailBulk.avgSpeed>5)&(sailBulk.dRatio<0.7)].voyageDist.describe())
411
412     #LNG Laden vs Ballas Voyages
413     lngLadenVsBallast = 0
414     if lngLadenVsBallast ==1:
415         d = [sailLng [(sailLng.maxMsgInt<6)&(sailLng.voyageDist>200)&(sailLng.
avgSpeed>5)&(sailLng.dRatio>0.9)].avgSpeed.values,\
416             sailLng [(sailLng.maxMsgInt<6)&(sailLng.voyageDist>200)&(sailLng.
avgSpeed>5)&(sailLng.dRatio<0.8)].avgSpeed.values]
417         plt.figure()
418         sns.boxplot(data = d, palette = [c1,c1], boxprops=dict(alpha=1))
419         plt.xticks(plt.xticks()[0],['Laden', 'Ballast'])

```

```

420     plt.ylabel('Average Speed')
421     plt.show()
422
423     print('Voyage Length: LNG in Laden')
424     print(sailLng[(sailLng.maxMsgInt<6)&(sailLng.voyageDist>200)&(sailLng.
avgSpeed>5)&(sailLng.dRatio>0.9)].voyageDist.describe())
425     print('Voyage Length: LNG in Ballast')
426     print(sailLng[(sailLng.maxMsgInt<6)&(sailLng.voyageDist>200)&(sailLng.
avgSpeed>5)&(sailLng.dRatio<0.8)].voyageDist.describe())
427
428
429
430     #Container Redistribution EastWest Atlntatic
431     containerRedist = 1
432     if containerRedist == 1:
433         #ASIA-AMERICA
434         #Speed
435         d = [sailContainer[(sailContainer.maxMsgInt<6)&(sailContainer.
voyageDist>200)&(sailContainer.avgSpeed>5)&(sailContainer.dir<180)&(
sailContainer.area=='Atl')].avgSpeed.values, \
436             sailContainer[(sailContainer.maxMsgInt<6)&(sailContainer.voyageDist
>200)&(sailContainer.avgSpeed>5)&(sailContainer.dir>=180)&(sailContainer.
area=='Atl')].avgSpeed.values]
437         plt.figure()
438         sns.boxplot(data = d, palette = [c1, c2], boxprops=dict(alpha=1))
439         plt.xticks(plt.xticks()[0], ['East Bound', 'West Bound'])
440         plt.ylabel('Average Speed')
441         plt.show()
442         #Draught Raio
443         d = [sailContainer[(sailContainer.maxMsgInt<6)&(sailContainer.
voyageDist>200)&(sailContainer.avgSpeed>5)&(sailContainer.dir<180)&(
sailContainer.area=='Atl')].dRatio.values, \
444             sailContainer[(sailContainer.maxMsgInt<6)&(sailContainer.voyageDist
>200)&(sailContainer.avgSpeed>5)&(sailContainer.dir>=180)&(sailContainer.
area=='Atl')].dRatio.values]
445         plt.figure()
446         sns.boxplot(data = d, palette = [c1, c2], boxprops=dict(alpha=1))
447         plt.xticks(plt.xticks()[0], ['East Bound', 'West Bound'])
448         plt.ylabel('Draught Ratio')
449         plt.show()
450
451         #ASIA-Europ
452         #Speed
453         d = [sailContainer[(sailContainer.maxMsgInt<6)&(sailContainer.
voyageDist>200)&(sailContainer.avgSpeed>5)&(sailContainer.dir<180)&(
sailContainer.area=='Ind')].avgSpeed.values, \
454             sailContainer[(sailContainer.maxMsgInt<6)&(sailContainer.voyageDist
>200)&(sailContainer.avgSpeed>5)&(sailContainer.dir>=180)&(sailContainer.
area=='Ind')].avgSpeed.values]
455         plt.figure()
456         sns.boxplot(data = d, palette = [c1, c2], boxprops=dict(alpha=1))
457         plt.xticks(plt.xticks()[0], ['East Bound', 'West Bound'])

```

```

458     plt.ylabel('Average Speed')
459     plt.show()
460     #Draught Raio
461     d = [sailContainer[(sailContainer.maxMsgInt<6)&(sailContainer.voyageDist>200)&(sailContainer.avgSpeed>5)&(sailContainer.dir<180)&(sailContainer.area=='Ind')].dRatio.values,\
462         sailContainer[(sailContainer.maxMsgInt<6)&(sailContainer.voyageDist>200)&(sailContainer.avgSpeed>5)&(sailContainer.dir>=180)&(sailContainer.area=='Ind')].dRatio.values]
463     plt.figure()
464     sns.boxplot(data = d, palette = [c1,c2], boxprops=dict(alpha=1))
465     plt.xticks(plt.xticks()[0], ['East Bound', 'West Bound'])
466     plt.ylabel('Draught Ratio')
467     plt.show()
468
469
470 toc = time.time()
471 print("Elapsed time total= ",toc-tic)

```

### B.3 db.py

```

1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Tue Feb  5 14:00:09 2019
5
6 @author: havard
7 """
8
9 #Import Packages
10 import sqlite3 as sql
11 import time
12 import datetime
13 import calendar
14 import numpy as np
15 import matplotlib.pyplot as plt
16 import pandas as pd
17
18 plt.style.use('seaborn-whitegrid')
19 c1 = 'steelblue'
20 c2 = 'darkorange'
21 c3 = 'lightseagreen'
22 c11 = 'lightblue'
23 c21 = 'bisque'
24 c31 = 'paleturquoise'
25
26
27
28 #Return df with messagecount pr vessel
29 #Message Frequency Per ship
30

```

```

31 def msgCountPrVessel(dbPath, table):
32     query = "SELECT userid, count(*) FROM %s GROUP BY userid ORDER BY count(*)"
33           "%(table)"
34     con = sql.connect(dbPath)
35     with con:
36         ais_dataframe = pd.read_sql_query(query, con)
37         con.close()
38
39     ais_dataframe.columns = ['userid', 'count']
40
41     return ais_dataframe
42
43 #EXTRACT RELEVANT DATA FROM DATABASE (Message Type 1)
44 def Extract1(dbPath, table, fromTime, toTime):
45     '''
46     Extracts info given f(dbpath, table, fromTime, toTime, fromSpe, toSpeed)
47     '''
48     tic = time.time()
49     conn = sql.connect(dbPath)
50     SQLstring = "SELECT unixtime, sog, latitude, longitude, userid, nav_status
51               ,\
52               cog, msgType FROM %s WHERE unixtime >= %s and unixtime <= %s ORDER BY \
53               UNIXTIME ASC" % (table, str(fromTime), str(toTime))
54     with conn:
55         df = pd.read_sql_query(SQLstring, conn)
56         conn.close()
57         print("Message Type 1 extraction time= ", time.time()-tic)
58         return df
59
60 #EXTRACT RELEVANT DATA FROM DATABASE (Message Type 5)
61 def Extract5(dbPath, table, fromTime, toTime):
62     tic = time.time()
63     conn = sql.connect(dbPath)
64     SQLstring = "SELECT unixtime, dest, imo, ship_type, \
65               userid, draught, msgType FROM %s WHERE unixtime >= %s and unixtime <= %s
66               ORDER BY \
67               UNIXTIME ASC" % (table, str(fromTime), str(toTime))
68     with conn:
69         df = pd.read_sql_query(SQLstring, conn)
70         conn.close()
71         print("Message Type 5 extraction time= ", time.time()-tic)
72         return df
73
74 def DatabaseInfo(dbPath, table):
75     '''
76     Function for displaying database statisites, f(dbpath, table)
77     '''
78     st, et, count, unique = Statistics(dbPath, table)
79     sT = datetime.datetime.utcnow().timestamp(st)
80     eT = datetime.datetime.utcnow().timestamp(et)
81     print(sT)

```

```

80     print(eT)
81     sY = datetime.datetime.utcnow().timestamp(st).year
82     eY = datetime.datetime.utcnow().timestamp(et).year
83     countYearly, uniqueYearly = StatisticsPrYear(dbPath, table, sY, eY)
84
85     years = list(range(sY, eY+1))
86     df = pd.DataFrame()
87
88     # df[0] = years
89     # df[1] = countYearly
90     # df[2] = uniqueYearly
91
92     print(years)
93     print(countYearly)
94     print(uniqueYearly)
95
96     fig, ax1 = plt.subplots()
97     ax1.set_xlabel('Year')
98     ax1.set_ylabel('Number of Messages', color = c1)
99     ax1.bar(np.arange(len(years)), countYearly, width=0.8, color=c1)
100    ax1.set_xticks(range(len(years)))
101    ax1.set_xticklabels(years)
102    ax2 = ax1.twinx()
103    ax2.set_ylabel('Number of Unique Vessels', color=c2)
104    ax2.scatter(np.arange(len(years)), uniqueYearly, c=c2)
105    # for i,txt in enumerate(uniqueYearly):
106    #     ax2.annotate(txt,(i, countYearly[i]))
107    for i in np.arange(len(years)):
108        ax2.annotate(uniqueYearly[i],(i+1, uniqueYearly[i]))
109    ax2.grid(False)
110    fig.tight_layout()
111    plt.show()
112    '''
113    fig = plt.figure() # Create matplotlib figure
114    ax = fig.add_subplot(111) # Create matplotlib axes
115    ax2 = ax.twinx() # Create another axes that shares the same x-axis as ax.
116    #width = 0.4
117
118    #df[1].plot(kind='bar', color= c1, ax=ax, width=width, position=1, alpha =
119    #df[2].plot(kind='bar', color= c2, ax=ax2, width=width, position=0, alpha
120    # = 0.7)
121
122    ax.set_ylabel('Number of Messages',color = c1)
123    ax.set_xlabel('Year')
124    ax.set_xticklabels(df[0])
125    ax2.set_ylabel('Number of Unique Vessels',color = c2)
126    ax2.grid(False)
127    fig.tight_layout()
128    plt.show()
129    '''
130 #Statistics for whole table

```

```

130 def Statistics(dbPath, table):
131     test = sql.connect(dbPath).cursor()
132     SQLstring = "SELECT COUNT(*) from %s" % (table)
133     count = test.execute(SQLstring).fetchall()
134     SQLstring = "SELECT MIN(unixtime) from %s" % (table)
135     st = test.execute(SQLstring).fetchall()
136     SQLstring = "SELECT MAX(unixtime) from %s" % (table)
137     et = test.execute(SQLstring).fetchall()
138     SQLstring = "SELECT COUNT(DISTINCT userid) from %s" % (table)
139     unique = test.execute(SQLstring).fetchall()
140     test.close()
141     return st[0][0], et[0][0], count[0][0], unique[0][0]
142
143 #Statistics for table pr. year
144 def StatisticsPrYear(dbPath, table, sY, eY):
145     countYearly = []
146     uniqueYearly = []
147     test = sql.connect(dbPath).cursor()
148     for i in range(sY, eY+1):
149         a = i
150         b = i+1
151         # convert the dates to unixtime
152         fromTime = calendar.timegm(datetime.datetime(a, 1, 1, 0, 0).timetuple())
153         toTime = calendar.timegm(datetime.datetime(b, 1, 1, 0, 0).timetuple())
154
155         SQLstring = "SELECT COUNT(*) from %s WHERE\
156         unixtime >= %s and unixtime <= %s" % (table, fromTime, toTime)
157         count = test.execute(SQLstring).fetchall()
158         SQLstring = "SELECT COUNT(DISTINCT userid) from %s WHERE\
159         unixtime >= %s and unixtime <= %s" % (table, fromTime, toTime)
160         countU = test.execute(SQLstring).fetchall()
161
162         countYearly.append(count[0][0])
163         uniqueYearly.append(countU[0][0])
164     test.close()
165     return countYearly, uniqueYearly

```

## B.4 readCSV.py

```

1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Wed Mar 13 09:33:17 2019
5
6 @author: havard
7 """
8 import pandas as pd
9
10
11 def decimalDegrees(str):
12     if str[-1]== 'N' or str[-1]== 'E':

```

```

13     a = int(str.split()[0].split('°')[0])
14     b= float(str.split()[0].split('°')[1])
15     return a + b/60
16     else:
17         a = int(str.split()[0].split('°')[0])
18         b= float(str.split()[0].split('°')[1])
19         return -a - b/60
20
21 def importPorts(type):
22     if type == 1:
23         str = '/Users/havard/Desktop/Master/Support_Files/container_ports.csv'
24     elif type == 2:
25         str = '/Users/havard/Desktop/Master/Support_Files/bulk_ports.csv'
26     elif type == 3:
27         str = '/Users/havard/Desktop/Master/Support_Files/lng_ports.csv'
28     else:
29         print('Wrong type for ports')
30     df = pd.read_csv(str, encoding = "ISO-8859-1")
31     df.dropna(subset=['Latitude', 'Longitude'], inplace=True)
32     df.drop_duplicates(subset=['Latitude', 'Longitude'], inplace=True)
33     df['Longitude'] = df['Longitude'].apply(decimalDegrees)
34     df['Latitude'] = df['Latitude'].apply(decimalDegrees)
35     return df
36
37 def importShips(type, maxYear):
38     if type == 1:
39         #str = '/Users/havard/Desktop/Master/Support_Files/
panamax_container_list.csv'
40         str = '/Users/havard/Desktop/Master/Support_Files/container.csv'
41     elif type == 2:
42         #str = '/Users/havard/Desktop/Master/Support_Files/panamax_bulk_list.
csv'
43         str = '/Users/havard/Desktop/Master/Support_Files/bulk.csv'
44     elif type == 3:
45         #str = '/Users/havard/Desktop/Master/Support_Files/lng_list.csv'
46         str = '/Users/havard/Desktop/Master/Support_Files/lng.csv'
47     else:
48         print('Wrong type for ports')
49     dfShips = pd.read_csv(str)
50     dfShips.dropna(subset=['MMSI'], inplace=True)
51     #Remove ships created after year of study
52     def convDateToNum(str):
53         temp = str.split('-')
54         return int(temp[0])+int(temp[1])/100
55     dfShips['Age'] = dfShips['Built'].apply(convDateToNum)
56     dfShips = dfShips[dfShips['Age']<maxYear]
57
58     dfShips = dfShips.drop(columns=['Age'])
59     #Sort by date so the x newest ships can esily be found
60     dfShips.sort_values(by=['Built'], axis=0, ascending=False, inplace=True)
61     return dfShips

```



## B.5 ais.py

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Thu Feb  7 14:27:01 2019
5
6 @author: havard
7 """
8
9 import numpy as np
10 import pandas as pd
11 import math
12 import matplotlib.pyplot as plt
13 import sqlite3 as sql
14 import time
15 import datetime
16 import calendar
17 import seaborn as sns
18 import matplotlib.pyplot as plt
19 import sklearn.cluster as cluster
20 from mpl_toolkits.basemap import Basemap
21
22
23
24 c1 = 'steelblue'
25 c2 = 'darkorange'
26 c3 = 'lightseagreen'
27 c11 = 'lightblue'
28 c21 = 'bisque'
29 c31 = 'paleturquoise'
30
31
32 #Validate that MMSI number not is changing
33 def validateIMO(df5):
34     changes = []
35     vessels = df5.userid.unique()
36     for vessel in vessels:
37         imoA = df5[df5.userid == vessel].imo.iloc[0]
38         imoB = df5[df5.userid == vessel].imo.iloc[-1]
39         if imoA != imoB:
40             changes.append(vessel)
41
42     return changes
43
44 #Hversinse Distanse: input = point
45 def haversine(lon1, lat1, lon2, lat2): # FINN KILDE!!!
46     """
47     Calculate the great circle distance between two points
48     on the earth (specified in decimal degrees)
49     """
50     # convert decimal degrees to radians
```

```

51 lon1 , lat1 , lon2 , lat2 = map(math.radians , [lon1 , lat1 , lon2 , lat2])
52 # haversine formula
53 dlon = lon2 - lon1
54 dlat = lat2 - lat1
55 a = math.sin(dlat/2)**2 + math.cos(lat1) * math.cos(lat2) * math.sin(dlon
/2)**2
56 c = 2 * math.asin(math.sqrt(a))
57 # Radius of earth in kilometers is 6371
58 km = 6371* c
59 return km
60
61 #Haversine Distances: input = list
62 def haversineDF(lon1 , lat1 , lon2 , lat2):
63     """
64     Calculate the great circle distance between two points
65     on the earth (specified in decimal degrees)
66     """
67
68     # convert decimal degrees to radians
69     lon1 , lat1 , lon2 , lat2 = map(np.radians , [lon1 , lat1 , lon2 , lat2])
70     # haversine formula
71     dlon = lon2 - lon1
72     dlat = lat2 - lat1
73     a = np.sin(dlat/2)**2 + np.cos(lat1) * np.cos(lat2) * np.sin(dlon/2)**2
74     c = 2 * np.arcsin(np.sqrt(a))
75     # Radius of earth in kilometers is 6371
76     km = 6371* c
77     return km
78
79 #Give new operational status
80 def lableStatus(df , ports):
81     tic = time.time()
82
83     #Function for finding the distance to the closes port
84     def distToPorts(lon , lat , dfPorts):
85         dist=list()
86         for row in dfPorts.itertuples():
87             dist.append(haversine(lon , lat , row[4] , row[3]))
88         return dist
89
90     #Funciton for assigning operatioanl status
91     def statusLable(row , dfPorts):
92         if row['sog']>3:
93             return 1
94         else:
95             if row['sog']<1:
96                 if row['nav_status'] == 5:
97                     return 2
98                 if row['nav_status'] == 1:
99                     return 3
100
101     dist = distToPorts(row['longitude'] , row['latitude'] , dfPorts)

```

```

102         d = min(dist)
103         if d<1.4:
104             return 2
105         elif d<45 and row['sog']<3:
106             return 3
107         else:
108             return 1
109
110     #Assign status as new columns
111     df['status'] = df.apply(statusLable, args=(ports, ), axis=1)
112
113     print('Lableing Time: ',time.time()-tic)
114     return df
115
116 def statusChange(df):
117     changes = 0
118
119     for row in df.iteruples():
120         if row.status ==1 and row.nav_status != 0:
121             changes = changes + 1
122         elif row.status == 2 and row.nav_status != 5:
123             changes = changes +1
124         elif row.status == 3 and row.nav_status != 1:
125             changes = changes + 1
126
127     return changes
128
129
130
131 #Use new operational Status to lable each leg ,port and wait
132 def legNumbering(df):
133     vessels = df.userid.unique()
134     df['Leg'] = 0
135     df['Port'] = 0
136     df['Wait'] = 0
137     #Help function for lable Leg nr
138     def legList(df):
139         leg = []
140         legNr = 1
141         temp = 2
142         for row in df.iteruples():
143             if row.status == 1:
144                 leg.append(legNr)
145                 temp = legNr
146             else:
147                 leg.append(0)
148                 if temp == legNr:
149                     legNr +=1
150         return leg
151     #Help function for lable port nr
152     def portList(df):
153         port = []

```

```

154     portNr = 1
155     temp = 2
156     for row in df.itertuples():
157         if row.status == 2:
158             port.append(portNr)
159             temp = portNr
160         else:
161             port.append(0)
162             if temp == portNr:
163                 portNr +=1
164     return port
165 #Help function for lable port nr
166 def waitList(df):
167     wait = []
168     waitNr = 1
169     temp = 2
170     for row in df.itertuples():
171         if row.status == 3:
172             wait.append(waitNr)
173             temp = waitNr
174         else:
175             wait.append(0)
176             if temp == waitNr:
177                 waitNr +=1
178     return wait
179
180 #Loop through all vessel and lable legs , port and wait number
181 for vessel in vessels:
182     legL = legList(df[df.userid == vessel])
183     df.loc[df.userid == vessel , 'Leg']=legL
184
185     portL = portList(df[df.userid == vessel])
186     df.loc[df.userid == vessel , 'Port']=portL
187
188     waitL = waitList(df[df.userid == vessel])
189     df.loc[df.userid == vessel , 'Wait']=waitL
190
191     return df
192
193 #Lable Static MessageTypes with Draught Ratio
194 def lableStatDraughtRatio(df5 , shipList):
195
196     #Lable Message Type 5 with draught ratio
197     def designDraught(row , df):
198         if df[df['MMSI']==row['userid']].size >0:
199             desD = df[df['MMSI']==row['userid']].Draught.values[0]
200             return row.draught/desD
201         else:
202             return 0
203     df5['dRatio'] = df5.apply(designDraught , args=(shipList ,) , axis=1)
204     return df5
205

```

```

206 #Calculate Time Sail, Port, Wait
207 def timeAnalytics(df, dfSail, dfPort, dfWait):
208
209     vessels = df['userid'].unique()
210     voyageTimes = []
211     waitingTimes = []
212     portTimes = []
213     vesselListS = []
214     vesselListW = []
215     vesselListP = []
216
217     for vessel in vessels:
218         dfTemp=df[df.userid == vessel]
219         #Sailing
220         dfT = dfTemp[dfTemp.Leg != 0]
221         times = (dfT.groupby('Leg').last()['unixtime']-dfT.groupby('Leg').
first()['unixtime'])/(60*60)
222         voyageTimes.extend(times)
223         vesselListS.extend([vessel]*len(times))
224
225         #Waiting
226         dfT = dfTemp[dfTemp.Wait != 0]
227         times = (dfT.groupby('Wait').last()['unixtime']-dfT.groupby('Wait').
first()['unixtime'])/(60*60)
228         waitingTimes.extend(times)
229         vesselListW.extend([vessel]*len(times))
230
231         #Port
232         dfT = dfTemp[dfTemp.Port != 0]
233         times = (dfT.groupby('Port').last()['unixtime']-dfT.groupby('Port').
first()['unixtime'])/(60*60)
234         portTimes.extend(times)
235         vesselListP.extend([vessel]*len(times))
236
237         dfSail['userid']= vesselListS
238         dfSail['voyageTime'] = voyageTimes
239
240         dfWait['userid'] = vesselListW
241         dfWait['waitingTime'] = waitingTimes
242
243         dfPort['userid'] = vesselListP
244         dfPort['portTime'] = portTimes
245
246     return dfSail, dfPort, dfWait
247
248
249 def legAnalytics(df, df5, dfSail):
250     vessels = df['userid'].unique()
251
252     maxMessageIntervall = []
253     voyageDistance = []
254     dRatio = []

```

```

255     area = []
256     heading = []
257     speed = []
258
259     for vessel in vessels:
260         dfTemp=df[df.userid == vessel]
261         dfT = dfTemp[dfTemp.Leg != 0]
262
263         #Find each leg's max message intervall
264         intervalls = dfT.groupby('Leg').unixtime.apply(lambda x: x.diff().max
265         ())/(60*60)
266         maxMessageIntervall.extend(intervalls)
267
268         #Sailinglennngts
269         voyageDistance.extend(dfT.groupby('Leg').apply(lambda x:haversineDF(x.
270         longitude,\
271         x.latitude ,x.longitude.shift(),x.latitude.shift()).sum()))
272
273         #DraughtRatio
274         startTimes = dfT.groupby('Leg').first()['unixtime']
275         endTimes = dfT.groupby('Leg').last()['unixtime']
276         for i in range(1,len(startTimes)+1):
277             dRatio.extend([df5[(df5.userid == vessel) & (df5.unixtime>=
278             startTimes[i])\
279             & (df5.unixtime<=endTimes[i])]['dRatio'].median()
280             ])
281
282         #Operational Area
283         area.extend(dfT.groupby('Leg').apply(lambda x: x.Area.iloc[len(x)//2]
284         ))
285         #Heading
286         heading.extend(dfT.groupby('Leg').apply(lambda x: x.cog.iloc[len(x)
287         //2]))
288         #Avg Speed
289         speed.extend(dfT.groupby('Leg').sog.mean())
290
291         dfSail['maxMsgInt'] = maxMessageIntervall
292         dfSail['voyageDist'] = voyageDistance
293         dfSail['dRatio'] = dRatio
294         dfSail['dir'] = heading
295         dfSail['area'] = area
296         dfSail['avgSpeed'] = speed
297         return dfSail
298
299 def speedAnalytics(speed):
300     plt.figure()
301     sns.distplot(speed, kde=False, norm_hist=True, hist_kws=dict(alpha=0.9))
302     plt.show()
303
304 def speedAnalyticsYearly(df):
305     year = []

```

```

301 speed = []
302 sY = datetime.datetime.utcnow().year
303 eY = datetime.datetime.utcnow().year
304 print(sY)
305 print(eY)
306
307 for i in range(sY,eY+1):
308     low = calendar.timegm(datetime.datetime(i,1,1,0,0).timetuple())
309     high = calendar.timegm(datetime.datetime(i+1,1,1,0,0).timetuple())
310     year.append(i)
311     speed.append(df[(df['unixtime']>=low) & (df['unixtime']<high)]['sog'].
values)
312
313
314 ax = sns.boxplot(data=speed, color="lightsteelblue")
315 ax.set_xticklabels(year)
316 ax.set(xlabel='Year', ylabel='Speed [knots]')
```

## B.6 machineLearning.py

```

1
2 #!/usr/bin/env python3
3 # -*- coding: utf-8 -*-
4 """
5 Created on Wed Mar 13 10:18:34 2019
6
7 @author: havard
8 """
9
10 import numpy as np
11 import pandas as pd
12 import math
13 import matplotlib.pyplot as plt
14 import sqlite3 as sql
15 import time
16 import datetime
17 import seaborn as sns
18 import matplotlib.pyplot as plt
19 import sklearn.cluster as cluster
20 from mpl_toolkits.basemap import Basemap
21
22
23 def clusterCentre(df):
24     dfRes = pd.DataFrame(columns=['longitude', 'latitude', 'portNr'])
25     ports = df['portNr'].unique()
26     for port in ports:
27         dfT = df[df['portNr']==port]
28         dfRes.loc[port]=[dfT['longitude'].mean(),dfT['latitude'].mean(),dfT['
portNr'].mean()]
29     return dfRes
30
```

```

31 def clusterDBSCAN(x,y,r):
32     '''
33     run DBSCAN clustering on input(x,y,r)
34     '''
35     #Haversine to convert radius in km to radians
36     km_pr_rad = 6371.0088
37     eps = r/km_pr_rad
38     points = list(zip(x,y))
39     dbscan = cluster.DBSCAN(eps=eps,min_samples = 5, algorithm = 'ball_tree',\
40                             metric='haversine').fit(np.radians(points))
41     labels = dbscan.labels_
42     n_clusters_ = len(set(labels))-1 if -1 in labels else 0
43     n_noise_ = list(labels).count(-1)
44
45
46     print('Number of clusters: ',n_clusters_)
47     print('Number of noise poitns: ',n_noise_)
48     print(labels)
49     df = pd.DataFrame()
50     df['longitude']= list(x)
51     df['latitude'] = list(y)
52     df['portNr']= list(labels)
53     df = df[df['portNr'] != -1]
54     df = clusterCentre(df)
55     return df
56
57
58 def clusterKMEANS(x,y,n):
59     '''
60     K-means clustering (x,y,number of clusters)
61     '''
62     points = list(zip(x,y))
63     k_means = cluster.KMeans(n_clusters = n).fit(points)
64     cluster_centres = k_means.cluster_centers_
65     labels = k_means.labels_
66     clusterlon = cluster_centres[:,0]
67     clusterlat = cluster_centres[:,1]
68
69
70     return clusterlon ,clusterlat ,labels

```

## B.7 ais\_Plotting.py

```

1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Wed Feb 27 12:11:57 2019
5
6 @author: havard
7 """
8 import numpy as np

```



```

9 import pandas as pd
10 import math
11 import matplotlib.pyplot as plt
12 import sqlite3 as sql
13 import time
14 import datetime
15 import calendar
16 import seaborn as sns
17 import matplotlib.pyplot as plt
18 import sklearn.cluster as cluster
19 from mpl_toolkits.basemap import Basemap
20
21 c1 = 'steelblue'
22 c2 = 'darkorange'
23 c3 = 'lightseagreen'
24 c11 = 'lightblue'
25 c21 = 'bisque'
26 c31 = 'paleturquoise'
27
28
29 def mapScatterGlobal(x,y):
30     '''
31     Plots scatter map for coordinates x(longitude), y(latitude)
32     '''
33     minlon = -180
34     minlat = -70
35     maxlon = 180
36     maxlat = 90
37     lat0 = (maxlat+minlat)/2
38     lon0 = (maxlon+minlon)/2
39     lat1 = (maxlat+minlat)/2-20
40
41     fig ,ax = plt.subplots(figsize=(15,7))
42     m = Basemap(llcrnrlon=minlon, llcrnrlat=minlat, urcrnrlon=maxlon,
43               urcrnrlat=maxlat, rsphere=(6378137.00,6356752.3142),
44               resolution='l', projection='cyl', lat_0=lat0, lon_0=lon0,
45               lat_ts=lat1)
46     m.drawmapboundary(fill_color='white')
47     m.fillcontinents(color='lightgrey',lake_color='white',zorder=1)
48     m.scatter(x,y,5,marker='o',c='blue',alpha = 0.5,zorder=2)
49     fig.tight_layout()
50
51
52 def mapScatterGlobalMsgType(df):
53     '''
54     Plots scatter map for coordinates x(longitude), y(latitude)
55     '''
56     minlon = -180
57     minlat = -70
58     maxlon = 180
59     maxlat = 90
60     lat0 = (maxlat+minlat)/2

```

```

61     lon0 = (maxlon+minlon)/2
62     lat1 = (maxlat+minlat)/2-20
63
64     fig , ax = plt . subplots ( figsize =( 15 , 7 ) )
65     m = Basemap ( llcrnrlon = minlon , llcrnrlat = minlat , urcrnrlon = maxlon ,
66                 urcrnrlat = maxlat , rsphere = ( 6378137.00 , 6356752.3142 ) ,
67                 resolution = 'l' , projection = 'cyl' , lat_0 = lat0 , lon_0 = lon0 ,
68                 lat_ts = lat1 )
69     m . drawmapboundary ( fill_color = 'white' )
70     m . fillcontinents ( color = 'lightgrey' , lake_color = 'white' , zorder = 1 )
71     m . scatter ( df [ df . msgType == 1 ] . longitude , df [ df . msgType == 1 ] . latitude , 1 , marker =
72                 'o' , c = c1 , alpha = 0.4 , zorder = 2 )
73     m . scatter ( df [ df . msgType == 2 ] . longitude , df [ df . msgType == 2 ] . latitude , 1 , marker =
74                 'o' , c = c3 , alpha = 0.8 , zorder = 2 )
75     m . scatter ( df [ df . msgType == 3 ] . longitude , df [ df . msgType == 3 ] . latitude , 1 , marker =
76                 'o' , c = c2 , alpha = 0.8 , zorder = 2 )
77     fig . tight_layout ()
78
79 def mapScatterLocal ( x , y ) :
80     '''
81     Plots scatter map for coordinates x(longitude) , y(latitude)
82     '''
83     minlon = max ( -180 , min ( x ) - 6 )
84     minlat = max ( -90 , min ( y ) - 6 )
85     maxlon = min ( 180 , max ( x ) + 6 )
86     maxlat = min ( 90 , max ( y ) + 6 )
87     lat0 = ( maxlat + minlat ) / 2
88     lon0 = ( maxlon + minlon ) / 2
89     lat1 = ( maxlat + minlat ) / 2 - 20
90
91     fig , ax = plt . subplots ( figsize =( 15 , 7 ) )
92     m = Basemap ( llcrnrlon = minlon , llcrnrlat = minlat , urcrnrlon = maxlon ,
93                 urcrnrlat = maxlat , rsphere = ( 6378137.00 , 6356752.3142 ) ,
94                 resolution = 'l' , projection = 'cyl' , lat_0 = lat0 , lon_0 = lon0 ,
95                 lat_ts = lat1 )
96     m . drawmapboundary ( fill_color = 'white' )
97     m . fillcontinents ( color = 'lightgrey' , lake_color = 'white' , zorder = 1 )
98     m . scatter ( x , y , 5 , marker = 'o' , c = 'blue' , alpha = 0.5 , zorder = 2 )
99     fig . tight_layout ()
100
101 def mapErroneousStatus ( df ) :
102     '''
103     Plots scatter map for coordinates x(longitude) , y(latitude) ,
104     '''
105     minlon = -180
106     minlat = -65
107     maxlon = 180
108     maxlat = 70
109     lat0 = ( maxlat + minlat ) / 2
110     lon0 = ( maxlon + minlon ) / 2
111     lat1 = ( maxlat + minlat ) / 2 - 20

```

```

110     fig ,ax = plt.subplots(figsize=(15,7))
111     m = Basemap(llcrnrlon=minlon, llcrnrlat=minlat, urcrnrlon=maxlon,
112               urcrnrlat=maxlat, rsphere=(6378137.00,6356752.3142),
113               resolution='l', projection='cyl', lat_0=lat0, lon_0=lon0,
114               lat_ts=lat1)
115     m.drawmapboundary(fill_color='white')
116     m.fillcontinents(color='lightgrey',lake_color='white',zorder=1)
117
118     df[df['nav_status']!=0]['nav_status'].value_counts()
119     print('INcude Table with distinct vessel sending these messages')
120
121     count = []
122
123     df1 = df[(df['nav_status']==5)&(df['sog']>0.5)]
124     count.append(df1.size)
125     m.scatter(df1['longitude'],df1['latitude'],1,marker='o',c=c1,alpha = 1,
126             zorder=2,label='Moored and Speed over 0.5: '+str(len(df1)))
127
128     df1 = df[(df['nav_status']==0)&(df['sog']==0)]
129     count.append(df1.size)
130     m.scatter(df1['longitude'],df1['latitude'],1,marker='o',c=c2,alpha = 1,
131             zorder=2,label='Underway sialing and speed =0: '+str(len(df1)))
132
133     df1 = df[(df['nav_status']==1)&(df['sog']>1)]
134     count.append(df1.size)
135     m.scatter(df1['longitude'],df1['latitude'],1,marker='o',c=c3,alpha = 1,
136             zorder=2,label='At anchor and Speed over 1: '+str(len(df1)))
137
138     df1 = df[(df['nav_status']!=0)&(df['nav_status']!=1)&(df['nav_status']!=5)
139             ]
140     m.scatter(df1['longitude'],df1['latitude'],1,marker='o',c=c11,alpha = 0.5,
141             zorder=2,label='Not status 0,1 or 5: '+str(len(df1)))
142
143     print('Number of Errounous stauts = '+ str(sum(count))+ ' Percent of total
144           : '+ str(sum(count)/df.size))
145
146     #lgnd = plt.legend(bbox_to_anchor=(0., 1.02, 1., .102), loc=3,ncol=2, mode
147     ="expand", borderaxespad=0.)
148     lgnd=ax.legend(loc='upper center', bbox_to_anchor=(0.5, -0.01),prop={'size
149           ': 11.5}, shadow=True,ncol=4)
150     lgnd.draw_frame(False)
151     lgnd.legendHandles[0]._sizes = [30]
152     lgnd.legendHandles[1]._sizes = [30]
153     lgnd.legendHandles[2]._sizes = [30]
154     lgnd.legendHandles[3]._sizes = [30]
155
156     fig.tight_layout()
157
158 def mapPlotDensity(df):

```

```

154     '''
155     Density plot for dataframe, df[2] = x,df[3] = y
156     '''
157     y = df[ 'latitude' ]
158     x = df[ 'longitude' ]
159     #     minlon = max(-180,min(x)-10)
160     #     minlat = max(-90,min(y)-10)
161     #     maxlon = min(180,max(x)+10)
162     #     maxlat = min(90,max(y)+10)
163     minlon = -180
164     minlat = -65
165     maxlon = 180
166     maxlat = 80
167     lat0 = (maxlat+minlat)/2
168     lon0 = (maxlon+minlon)/2
169     lat1 = (maxlat+minlat)/2-20
170
171     fig ,ax = plt.subplots(figsize=(15,7))
172     m = Basemap(llcrnrlon=minlon, llcrnrlat=minlat , urcrnrlon=maxlon ,
173               urcrnrlat=maxlat , rsphere=(6378137.00,6356752.3142) ,
174               resolution='l' , projection='cyl' , lat_0=lat0 , lon_0=lon0 ,
175               lat_ts=lat1)
176     m.drawmapboundary(fill_color='white')
177     m.fillcontinents(color='lightgrey' , lake_color='white' , zorder=1)
178     m.drawcountries(color='white')
179
180     vessels = df[ 'userid' ].unique()
181     for MML in vessels:
182         df_temp = df[df[ 'userid' ] == MML].copy()
183         df_diff = df_temp[['latitude' , 'longitude']].diff().abs() #Plot only
184         with dist close to eachoder
185         df_temp[(df_diff[ 'latitude' ] > 5) | (df_diff[ 'longitude' ] > 5)] = np.
186         nan
187         m.plot(df_temp[ 'longitude' ] , df_temp[ 'latitude' ] , 0.01 , c=c1 , alpha =
188         0.02 , zorder=2)
189     plt.tight_layout()
190
191 def mapPlotDensityType(df , df2 , df3):
192     y = df[ 'latitude' ]
193     x = df[ 'longitude' ]
194     minlon = -180
195     minlat = -65
196     maxlon = 180
197     maxlat = 70
198     lat0 = (maxlat+minlat)/2
199     lon0 = (maxlon+minlon)/2
200     lat1 = (maxlat+minlat)/2-20
201
202     fig ,ax = plt.subplots(figsize=(15,7))
203     m = Basemap(llcrnrlon=minlon, llcrnrlat=minlat , urcrnrlon=maxlon ,
204               urcrnrlat=maxlat , rsphere=(6378137.00,6356752.3142) ,

```

```

203         resolution='l', projection='cyl', lat_0=lat0, lon_0=lon0,
204         lat_ts=lat1)
205     m.drawmapboundary(fill_color='white')
206     m.fillcontinents(color='lightgrey',lake_color='white',zorder=1)
207     m.drawcountries(color='white')
208
209     vessels = df['userid'].unique()
210     for MML in vessels:
211         df_temp = df[df['userid'] == MML].copy()
212         df_diff = df_temp[['latitude', 'longitude']].diff().abs() #Plot only
213     with dist close to eachoder
214         df_temp[(df_diff['latitude'] > 5) | (df_diff['longitude'] > 5)] = np.
nan
215         m.plot(df_temp['longitude'],df_temp['latitude'],0.02,c=c2,alpha =
0.02,zorder=2)
216
217     vessels = df2['userid'].unique()
218     for MML in vessels:
219         df_temp = df2[df2['userid'] == MML].copy()
220         df_diff = df_temp[['latitude', 'longitude']].diff().abs() #Plot only
221     with dist close to eachoder
222         df_temp[(df_diff['latitude'] > 5) | (df_diff['longitude'] > 5)] = np.
nan
223         m.plot(df_temp['longitude'],df_temp['latitude'],0.02,c=c1,alpha =
0.02,zorder=2)
224
225     vessels = df3['userid'].unique()
226     for MML in vessels:
227         df_temp = df3[df3['userid'] == MML].copy()
228         df_diff = df_temp[['latitude', 'longitude']].diff().abs() #Plot only
229     with dist close to eachoder
230         df_temp[(df_diff['latitude'] > 5) | (df_diff['longitude'] > 5)] = np.
nan
231         m.plot(df_temp['longitude'],df_temp['latitude'],0.02,c=c3,alpha =
0.02,zorder=2)
232     plt.tight_layout()
233
234 def plotOceanPolygons(polygons):
235     fig,ax = plt.subplots(figsize=(15,7))
236     m = Basemap(rsphere=(6378137.00,6356752.3142), resolution='l', projection='
cyl',lon_0=0)
237     m.drawmapboundary(fill_color='white')
238     m.fillcontinents(color='lightgrey',lake_color='white')
239     for polygon in polygons:
240         x,y = zip(*polygon)
241         m.plot(x,y,markersize = 0)
242         ax.fill(x,y,alpha=0.2)
243     plt.show()
244

```

```

245 def plotWorldOceans(x,y,poly):
246     '''
247     Visualisation of polygon, centre in x,y and polygon
248     '''
249     minlon = max(-180,x-2)
250     minlat = max(-90,y-2)
251     maxlon = min(180,x+2)
252     maxlat = min(90,y+2)
253     lat0 = (maxlat+minlat)/2
254     lon0 = (maxlon+minlon)/2
255     lat1 = (maxlat+minlat)/2-20
256
257     fig,ax = plt.subplots(figsize=(15,15))
258     m = Basemap(llcrnrlon=minlon, llcrnrlat=minlat, urcrnrlon=maxlon,
259               urcrnrlat=maxlat, rsphere=(6378137.00,6356752.3142),
260               resolution='l', projection='cyl', lat_0=lat0, lon_0=lon0,
261               lat_ts=lat1)
262     m.drawmapboundary(fill_color='white')
263     m.fillcontinents(color='lightgrey',lake_color='white',zorder=1)
264     x1,y1 = zip(*poly)
265     m.plot(x1,y1,1,c='red',alpha=0.3)
266     m.plot([x1[-1],x1[0]],[y1[-1],y1[0]],1,c='red',alpha=0.2,zorder=2)
267     m.scatter(x1,y1,2,c='red',alpha=0.5,zorder=2)
268     m.scatter(x,y,10,c='red',marker='o',alpha=1,zorder=2)
269     ax.fill(x1,y1,c='blue',alpha=0.1)
270
271
272
273 def plotWorldOceans(polygons):
274     '''
275     Visualisation of polygon, centre in x,y and polygon
276     '''
277     minlon = -180
278     minlat = -65
279     maxlon = 180
280     maxlat = 75
281     lat0 = (maxlat+minlat)/2
282     lon0 = (maxlon+minlon)/2
283     lat1 = (maxlat+minlat)/2-20
284
285     fig,ax = plt.subplots(figsize=(15,15))
286     m = Basemap(llcrnrlon=minlon, llcrnrlat=minlat, urcrnrlon=maxlon,
287               urcrnrlat=maxlat, rsphere=(6378137.00,6356752.3142),
288               resolution='l', projection='cyl', lat_0=lat0, lon_0=lon0,
289               lat_ts=lat1)
290     m.drawmapboundary(fill_color='white')
291     m.fillcontinents(color='lightgrey',lake_color='white',zorder=1)
292     #m.drawcountries(color='white')
293     colors = [c1,c1,c2,c3,c11,c21,c31,c31]
294     for polygon,color in zip(polygons,colors):
295         x1,y1 = zip(*polygon)
296         m.plot(x1,y1,1,c=color,alpha=0.3)

```

```

297     m.plot([x1[-1],x1[0]],[y1[-1],y1[0]],1,c = color,alpha = 0.2,zorder =
298     2)
299     ax.fill(x1,y1,c = color,alpha=0.5)
300
301
302 def plotLegNr(df):
303     fig,ax = plt.subplots(figsize=(15,7))
304     m = Basemap(rsphere=(6378137.00,6356752.3142),resolution='l',projection='
305     cyl',lon_0=0)
306     m.drawmapboundary(fill_color='white')
307     m.fillcontinents(color='lightgrey',lake_color='white')
308     legNr = df.Leg.unique()
309     df = df[df.Leg!=0]
310     for leg in legNr:
311         m.scatter(df[df.Leg == leg].longitude,df[df.Leg == leg].latitude,1,
312         alpha = 1)
313     plt.show()
314
315 def mapStatusScatterLocal(df):
316     '''
317     Plots scatter map for coordinates x(longitude), y(latitude)
318     '''
319     x = df.longitude
320     y = df.latitude
321     minlon = max(-180,min(x)-6)
322     minlat = max(-90,min(y)-6)
323     maxlon = min(180,max(x)+6)
324     maxlat = min(90,max(y)+6)
325     lat0 = (maxlat+minlat)/2
326     lon0 = (maxlon+minlon)/2
327     lat1 = (maxlat+minlat)/2-20
328
329     fig,ax = plt.subplots(figsize=(15,15))
330     m = Basemap(llcrnrlon=minlon,llcrnrlat=minlat,urcrnrlon=maxlon,
331     urcrnrlat=maxlat,rsphere=(6378137.00,6356752.3142),
332     resolution='l',projection='cyl',lat_0=lat0,lon_0=lon0,
333     lat_ts=lat1)
334     m.drawmapboundary(fill_color='white')
335     m.fillcontinents(color='lightgrey',lake_color='white',zorder=1)
336     print('New')
337     m.scatter(df[df.status==1].longitude,df[df.status==1].latitude,1,marker='o
338     ',c=c3,alpha = 0.1,zorder=2,label = 'Sailing')
339     m.scatter(df[df.status==2].longitude,df[df.status==2].latitude,5,marker='o
340     ',c=c1,alpha = 0.8,zorder=2,label = 'In Port')
341     m.scatter(df[df.status==3].longitude,df[df.status==3].latitude,5,marker='o
342     ',c=c2,alpha = 0.8,zorder=2,label = 'Anchor')
343 #     m.scatter(df[df.nav_status==0].longitude,df[df.nav_status==0].latitude,1,
344 #     marker='o',c=c3,alpha = 0.1,zorder=2,label = 'Sailing')
345 #     m.scatter(df[df.nav_status==5].longitude,df[df.nav_status==5].latitude,5,

```

```

342 marker='o',c=c1,alpha = 0.8,zorder=2,label = 'In Port')
# m.scatter(df[df.nav_status==1].longitude,df[df.nav_status==1].latitude,5,
marker='o',c=c2,alpha = 0.8,zorder=2,label = 'Anchor')
343 fig.tight_layout()
344
345
346 lgnd=ax.legend(loc='upper center', bbox_to_anchor=(0.5, -0.01),prop={'size
': 11.5}, shadow=True,ncol=3)
347 lgnd.draw_frame(False)
348 lgnd.legendHandles[0]._sizes = [100]
349 lgnd.legendHandles[1]._sizes = [100]
350 lgnd.legendHandles[2]._sizes = [100]

```

## B.8 locationCheck.py

```

1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Thu Mar  7 09:33:47 2019
5
6 @author: havard
7 """
8 import time
9 import datetime
10 import calendar
11 import numpy as np
12 import pandas as pd
13 import matplotlib.pyplot as plt
14 import seaborn as sns
15 from mpl_toolkits.basemap import Basemap
16
17
18 def generatePolygons():
19     '''
20     Generates the oceanc poylgons, and return list containg them
21     '''
22     polygons = list()
23     westPacific = [[-180,90],[-100,90],[-103,22],[-76,7],[-61,-19],
24                   [-80,-90],[-180,-90]]
25     eastPacific = [[180,90],[101,90],[101,15],[145,-29],[145,-90],[180,-90]]
26
27     atlantic = [[25,-90],[32,30.5],[37,32],[37,90],[-100,90],
28               [-103,22],[-76,7],[-61,-19],[-80,-90],[25,-90]]
29     indian =
30     [[25,-90],[32,30.5],[37,32],[37,90],[101,90],[101,15],[145,-29],[145,-90],[25,-90]]
31
32     polygons.append(westPacific)
33     polygons.append(eastPacific)
34     polygons.append(atlantic)
35     polygons.append(indian)

```



```

35
36
37     return polygons
38
39 def pInsidePoly(x,y,polygon):
40     '''
41     Check if point insde polygon, using Ray-Casting algorithm
42     (Longitude, Latityde, polygon)
43     return true if insde, else false
44     '''
45     n = len(polygon)
46     result = False
47     x1,y1 = polygon[0]
48     for i in range(n+1):
49         x2,y2 = polygon[i % n]
50         if y>min(y1,y2):
51             if y <= max(y1,y2):
52                 if x <= max(x1,x2):
53                     if y1 != y2:
54                         xinters = (y-y1)*(x2-x1)/(y2-y1)+x1
55                     if x1 == x2 or x <= xinters:
56                         result = not result
57     x1,y1 = x2,y2
58     return result
59
60
61
62 def dfInsidePoly(dfI,polygon):
63     '''
64     Check if Datafarames points is
65     (DataFrame, polygon)
66     Returns a df with points inside the polygon
67     '''
68     #Minimize search
69     x,y = zip(*polygon)
70     max_x,min_x,max_y,min_y = max(x),min(x),max(y),min(y)
71     dfIn = dfI[(dfI['longitude'] < max_x) & (dfI['longitude'] > min_x) &
72               (dfI['latitude'] < max_y) & (dfI['latitude'] > min_y)].copy()
73
74     #Defining new colum in dataframe
75     dfIn['inside'] = False
76
77     for i in dfIn.index:
78         if pInsidePoly(dfIn['longitude'][i],dfIn['latitude'][i],polygon):
79             dfIn.at[i,'inside'] = True
80
81     dfIn = dfIn[dfIn.inside == True]
82     return dfIn
83
84
85 def lableArea(df):
86     df['Area'] = 'init'

```

```

87 polygons = generatePolygons()
88 lables = ['Pac', 'Pac', 'Atl', 'Ind']
89
90 for polygon, lable in zip(polygons, lables):
91
92     def rayCast(row, polygon):
93         if row.Area == 'init':
94             x,y = zip(*polygon)
95             max_x, min_x, max_y, min_y = max(x), min(x), max(y), min(y)
96             if (row.longitude < max_x) & (row.longitude > min_x) & (row.
latitude < max_y) & (row.latitude > min_y):
97                 if pInsidePoly(row.longitude, row.latitude, polygon):
98                     return lable
99                 return row.Area
100             x,y = zip(*polygon)
101             max_x, min_x, max_y, min_y = max(x), min(x), max(y), min(y)
102             #df.loc [(df.longitude < max_x) & (df.longitude > min_x) &
103             #         (df.latitude < max_y) & (df.latitude > min_y), 'Area']. apply(rayCast,
args=(polygon, ), axis=1)
104
105             #apply(rayCast, args=(polygon, ), axis=1)
106
107             df['Area'] = df.apply(rayCast, args=(polygon, ), axis=1)
108         return df
109
110
111 #For visualization of Geofencing
112 def lableExtremal(df, polygon):
113     x,y = zip(*polygon)
114     max_x, min_x, max_y, min_y = max(x), min(x), max(y), min(y)
115     df.loc [(df.longitude < max_x) & (df.longitude > min_x) &
116             (df.latitude < max_y) & (df.latitude > min_y), 'location'] = 1
117
118     def lableInside(row, polygon):
119         if pInsidePoly(row.longitude, row.latitude, polygon):
120             return 2
121         return row.location
122
123     df['location'] = df.apply(lableInside, args=(polygon, ), axis=1)
124     return df
125
126 def visualizeGeoFence(df):
127     df['location'] = 0
128
129     polygons = generatePolygons()
130
131     df = lableExtremal(df, polygons[0])
132     df = lableExtremal(df, polygons[1])
133
134     c1 = 'steelblue'
135     c2 = 'darkorange'
136     c3 = 'lightseagreen'

```

```
137 c11 = 'lightblue'
138 c21 = 'bisque'
139 c31 = 'paleturquoise'
140
141 minlon = -180
142 minlat = -65
143 maxlon = 180
144 maxlat = 75
145 lat0 = (maxlat+minlat)/2
146 lon0 = (maxlon+minlon)/2
147 lat1 = (maxlat+minlat)/2-20
148
149 fig ,ax = plt.subplots(figsize=(15,7))
150 m = Basemap(llcrnrlon=minlon, llcrnrlat=minlat, urcrnrlon=maxlon,
151            urcrnrlat=maxlat, rsphere=(6378137.00,6356752.3142),
152            resolution='l', projection='cyl', lat_0=lat0, lon_0=lon0,
153            lat_ts=lat1)
154 m.drawmapboundary(fill_color='white')
155 m.fillcontinents(color='lightgrey', lake_color='white', zorder=1)
156 m.scatter(df[df.location==1].longitude, df[df.location==1].latitude, 2,
157          marker='o', c=c2, alpha = 0.5, zorder=2)
158 m.scatter(df[df.location==2].longitude, df[df.location==2].latitude, 2,
159          marker='o', c=c1, alpha = 0.5, zorder=2)
160 m.scatter(df[df.location==0].longitude, df[df.location==0].latitude, 2,
161          marker='o', c='black', alpha = 0.2, zorder=2)
162
163 for polygon in polygons[:2]:
164     x1,y1 = zip(*polygon)
165     m.plot(x1,y1,1,c=c1,alpha=0.3)
166     m.plot([x1[-1],x1[0]],[y1[-1],y1[0]],1,c=c1,alpha=0.2,zorder=2)
167     ax.fill(x1,y1,c=c1,alpha=0.5)
168
169 fig.tight_layout()
170 return df
```

