NTNU
Norwegian University of
Science and Technology

Ship emissions calculation from AIS

Stian Glomvik Rakke

Master of Science in Engineering and ICT	
Submission date:	June 2016
Supervisor:	Bjørn Egil Asbjørnslett, IMT
Co-supervisor:	Ørnulf Jan Rødseth, MARINTEK

Norwegian University of Science and Technology
Department of Marine Technology

Master Thesis

SHIP EMISSIONS CALCULATION FROM AIS

June 24, 2016

Stian Glomvik Rakke
Norwegian University of Science and Technology
Department of Marine Technology
Supervisor: Bjørn Egil Asbjørslett
Co-supervisor: Ørnulf Jan Rødseth

Preface

This thesis was written during the spring of 2016 at the Norwegian University of Science and Technology (NTNU) in Trondheim, Department of Maritime Engineering. The report is written as a specialization project in the master degree program of Engineering and ICT.

I would like to thank my supervisor, Professor Bjørn Egil Asbjørnslett and co-supervisor at NTNU, Professor Ørnulf Jan Rødseth, for guidance and useful feedback during this thesis writing.

I would also like to give acknowledgment to classmates and other study acquaintances who helped me in the theory of ship design and shipping, as well as computer programming and maritime engineering in general.

Trondheim, June 24, 2016

Stian Glomvik Rakke

Abstract

A methodology, named ECAIS, is presented to calculate ship emissions based on their fuel consumption from AIS data. This was done to avoid use of commercial ship databases, which can be expensive for research on sizable fleets. Using the approximation method Holtrop-Mennen it ws possible to find a distinct ships propulsion power requirements for different speeds. This empirical method uses main ship characteristics for calculation. From AIS data several main ship characteristics could be derived. Remaining characteristics was found by generic ship approximation found in literature surveys. This was used in combination with power prediction and specific fuel consumption, and applied to different ship size categories, as fuel consumption is calculated from speeds in dynamic AIS data. Fuel consumption and CO2 emission were derived and compared to earlier studies.

Results show a sizable difference from Third IMO GHG study. As this study has only been made for a limited number of data, calculations contains substantial uncertainties which should be investigated further. Further improvements for ECAIS method has been emphasized, which is believed to improve results.

Sammendrag

I denne masteroppgaven presenteres en metode, kalt for ECAIS, som regner ut utslipp av miljøgasser fra AIS-data. Dette gjøres for å unngå brukes av kommersielle skipsdatabaser, som det kan være dyrt å hente informasjon til forskning på store mengder skip fra. Ved å benytte Holtrop-Mennen er det mulig å finne effektbehovet for propulsjon for ulike hastigheter i et bestemt skip. Denne metoden benytter seg av skipskarakteristikker for utregninger. Gjennom AIS-data kan man finne noen av disse skipskarakteristikkene. De resterende skipskarakteristikkene ble utledet ved hjelp av generelle approksimasjoner som ble funnet i litteraturstudier. Dette er i en kombinasjon med effektbehovet og et spesifikk forbruksmål satt i sammenheng med skipsstørrelser, brukt til utregning av skipets forbruk ved spesifikke hastigheter. Disse hastighetene er funnet i de dynamiske AIS-dataene. Brenselforbruket og utslippet av miljøgasser ble regnet ut og sammenlignet med tidligere studier.

Resultatet viser en stor forskjell fra utregninger gjort in den tredje IMO GHG-studien. Siden denne studien kun er utført på et mindre datasett er det stor usikkerhet rundt tallene. Dette burde bli analysert of utviklet videre. Videre forbedringer er pekt ut, og dette vil trolig forbedre resultatet til methoden.

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

AIS: Automatic Identification System
DWT: Deadweight Tonnage
EFDB: Emission Factor Database
EEDI: Energy Efficiency Design Index GHG: Green house gases
IMO: International Maritime Organization
IPCC: Intergovernmental Panel on Climate Changes
NTNU: Norwegian University of Science and Technology
TEU: Twenty foot equivalent units
UTC: Coordinated Universal Time
VHF: Very High Frequency
VTS: Vessel Traffic Service

Symbols

V = Velocity

P_D = Power delivered

P_E = Effective power

P_B = Brake power

P_T = Trust power

L_{OA} = Length overall

L_{pp} = Ship length between perpendicular

L_{wl} = Ship length on waterline

B = Ship breadth on waterline

T = Ship draught amidships

∇ = Volume displacement

Δ = Displacement, 1.025∇

g = Acceleration of gravity, $9.81 m/s^2$

C_P = Prismatic Coefficient

C_B = Block Coefficient

C_F = Frictional Coefficient

C_{Wp} = Waterplane Area Coefficient

C_M = Mindship Section Coefficient

C_{Stern} = Stern Shape Parameter

S = Wetted Surface Hull

S_{App} = Wetted Area Appendages

Z = Number of Propeller Blades

A_T = Transom Area

h_B = Center of Bulb Area Above Keel Line

A_{BT} = Transverse Bulb Area

1 | Introduction

As a vital enabler for global trade and prosperity shipping constitutes a large share of the worlds transportation of commodities. Hence, ship emissions has received great focus in recent years. GHG emissions from shipping accounted for approximately 2.4% of global emissions in 2012 (Smith et al., 2014). Several goals for reducing emissions has been introduced, in example regulations that prohibit deliberate emissions of ozone depleting substances (IMO, 2016). Extensive work has been done to implement new regulations in the shipping industry to meet these goals. An effort to measure total emissions in world shipping fleet has consequently needed new research, as shipping data earlier has been insufficient.

Automatic Identification System (AIS) was initially introduced as an anti-collision system; providing live ship tracking along with identification number and several main ship characters. This has later been exploited in different research areas as AIS provide and gather a significant amount of data. Emissions calculation studies has been conducted through AIS analysis of ship journeys. This is combined with ship databases that contains ship main characteristics. An example of this is a modelling system for exhaust emission of marine traffic in the Baltic sea presented by Jalkanen, Brink, Kalli, Pettersson, Kukkonen & Stipa (2009).

1.1 Objectives

This thesis aims to calculate global emissions from ship traffic. More specifically, the target is to utilize AIS data for development of a method that estimate fuel consumption. It should take use of known ship design rules or approximations, and power prediction approximations. Specific fuel oil consumption for different ship type should be utilised in calculations of fuel consumptions and emissions. This method should only make use of input from AIS data.

1.2 Approach

Development of ship categorization on AIS data by Smestad (2015), opens possibilities for differentiate by ship group when extracting AIS data. This is exploited together with power prediction estimation methods in a computer program for calculation of fuel consumption. As AIS does not provide all ship characteristics, missing characteristics are derived from literature survey. The study done by Smith et al. (2014) is used for comparison, together with real authentic ship fuel consumptions.

1.3 Contributions

This thesis develops a method for estimating global fuel consumption without ship databases. Emissions is derived from this consumption. In this context main contributions from this work are:

- Avoid unaffordable expenses from buying and retrieving commercial digital databases for whole fleets.
- Simplifies emissions calculation
- Independent of commercial parties

1.4 Literature survey

Heuristics for categorizing ships by AIS data was presented in a master thesis in 2015 by Smestad (Smestad, 2015). This opened up several other possibilities in AIS research. An approach to adopt this heuristics to emission calculation was enabled. Third IMO GHG study is a global emissions study by International Maritime Organisation (Smith et al., 2014). This is an updated of earlier research with regards to shipping. This study evaluates shipping emissions during the period 2007-2012. Key findings in this study is shipping emissions relative to other anthropogenic emissions, quality and uncertainties of the emission inventories, comparison of emissions to second IMO GHG study, fuel trends and future scenarios. Third IMO GHG study is used as basis for comparison to this thesis.

1.5 Thesis organization

Chapter 1 gives an introduction to the thesis. It displays different objectives and contributions, as well as the approach and literature survey carried out to accomplish this.

Chapter 2 presents theory dealt with in this thesis. This includes AIS, World fleet resistance, power prediction, fuel consumption emissions and Ship dimensions

Chapter 3 presents the method for simplified ship emissions calculation from AIS called ECAIS. This chapter describes the ECAIS Method put forth in this thesis. Each part of the method is explained. This includes values for all main characteristics used in Holtrop-Mennen and other essential ship characteristics.

Chapter 4 presents results from calculations done by developed computer program. This includes different tests of constraints. AIS data from a May 1. 2014 until September 15. 2014 are processed. Results are discussed in chapter 5.

Chapter 5 contains discussion concerning the ECAIS method as a whole. Weaknesses of ECAIS method are discussed and further work is presented.

Chapter 6 presents conclusion of this thesis and ECAIS method.

Appendices

2 | Theory

In this chapter, theory is presented to give a brief introduction to AIS and to other key elements that are essential to understand how shipping emissions are calculated. The chapter is divided into sections to help the reader get a quicker overview of the issue in hand, giving only brief discussions to each section. Heuristics presented later in this thesis is based on theory from this chapter.

2.1 Introduction to AIS Data

Automatic Identification System (AIS) is a communication system introduced in 2002, to enhance: safety of life at sea; the safety and efficiency of navigation; and the protection of the marine environment (IMO, 2003). Messages between ships, and with a base station on shore, will be received by either ships directly, buoys, land based station, and satellites.
AIS uses Very High Frequency(VHF) system. A specified protocol for communication with specific information is transmitted from the ship, which is divided in static data and dynamical data. Recent year specially dedicated satellites has been launched, receiving larger number of AIS messages and wider coverage. This is called S-AIS.

Ruling guidelines for use of AIS reporting is given by SOLAS. [1] It states that ships above 300 gross tonnage engaged in international voyages, cargo ships of 500 gross tonnages and upward not engaged on international voyages as well as all passenger ships built after 2002, or operated after 2008, should have an AIS system installed (IMO, 2003). By May 31. 2013 ships of more than 18 meters were also required to install AIS class A . This was later to apply to all ships of more than 15 meters by May 31. 2014 (European CommissionC, 2011). For VHF transmitters and receivers range of AIS is nominally a little less than 40 km (Navcen, 2016). Coverage will mainly depend upon height of the antenna, while the surrounding geographical landscape and heights also contributes with regards to range.

---

[1] International convention for the Safety of lives at sea

2.1.1 Types of AIS

AIS differentiate between class A and class B equipment. **Class A** (Message type 1, 2 and 3) autonomously report their position every 2-10 seconds. This will depend on their speed and course. Reporting is less frequent when moored, only every three minutes. Vessel static and voyage related information (Message type 5) will be reported every 6 minutes. Class A may also send safety related information, meteorological and hydrological data, electronic broadcast to mariners, and other information marine safety messages.

Class B equipment can also be used together with all AIS base stations, but does not meet all the performance standards adopted by IMO. As for this class they report every third minute or less when moored, similar to Class A stations. As for the their position (message 6/8), messages are sent less often and at a lower power. Static data (message 18/24) will be reported every 6 minutes. They can only receive safety related messages, not send them (Navcen, 2016a).

In all AIS contains of 27 different messages types that can be transmitted. Message type 1-4 are the most frequently used. Message type 28-63 is reserved for future use. From static data information such as destination and ship characteristics are given, whereas dynamic data navigational details, speed and data from sensors are some of the essential data.

2.1.2 Use

With the introduction of mandatory AIS reporting in 2002. AIS was primarily introduced as a anti-collision system. However, today there are several areas of AIS usage. Furthermore additional information may also be added to the AIS message, to get an even greater usage. The third IMO GHG study (Smith et al., 2014) uses AIS as a tool to estimate global shipping emissions inventories. Mandatory AIS reporting has lead to a greatly improved shipping emissions estimations.

Position in BitVector	Description
1-6	Message type
7-8	Repeat indicator
9-38	UserID (MMSI)
39-40	AIS Version
41-70	Imo Number
71-112	Call Signal
113-232	Vessel Name
233-240	Ship type
241-249	Dimension to Bow (m)
250-258	Dimension to Stern (m)
259-264	Dimension to Port(m)
256-270	Dimension to Starboard(m)
271-294	ETA at destination (MMDDHHMM)
295-302	Draught (m)
303-422	Destination
423-423	DTE
424-424	Spare

Table 2.1: Message Type 5: Static message (ITU, 2014).

Marine traffic[2] is one of many places where you can get information about ships and ship movement from AIS. It presents a list of research areas where AIS is used. Examples given are; Study of marine telecommunications in respect of efficiency and propagation parameters, and secondly simulation of vessel movements in order to contribute to the safety of navigation and to cope with critical incidents. Moreover other examples are interactive information systems design, design of databases providing real-time information and statistical processing of ports' traffic with applications in operational research. Additional examples given are design of models for the spotting of the origin of pollution-related incidents, design of efficient algorithms for sea path evaluation and for determining the estimated time of ship arrivals. Last examples given are correlation of the collected information with weather data, and cooperation with institutes dedicated to the protection of the environment.

DNV-GL[3] presents another list of AIS usage, with regards to business decisions. Some examples given; how do partners/competitors run their networks? How many direct connections and transhipment do they offer? Which charter vessels have a higher chance of marine growth? Which ports/terminals have congestion issues? How do partners/competitors perform in terms of slow steaming and constant speed profile? How does this affect their fuel bill? What is the operational cost breakdown of other players? (DNV GL, 2016) This shows the wide range of areas where AIS is applicable.

2.2 World Fleet

World cargo fleet is about 65% of world fleet in total. About 90% of all transportation is carried by international shipping. Table 2.2 represent the difference between world cargo fleet and total world fleet and Figure 2.1 represent different ship types and fleet size within this category.

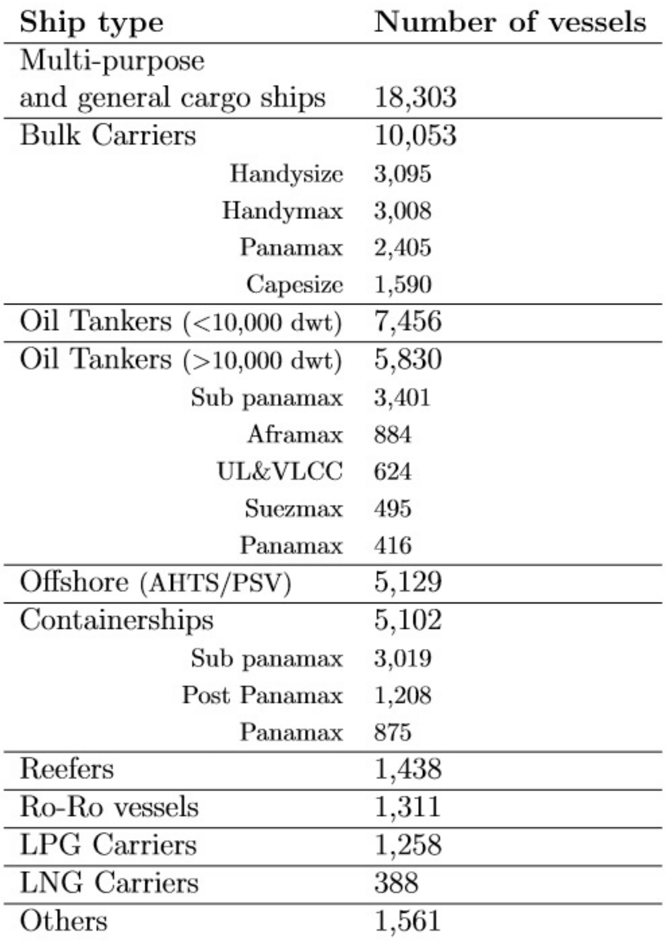
---

[2] A website that provides free near real-time information to the public regarding vessels' positions and movements as well as other related information for ships. http://www.marinetraffic.com/

[3] The world's leading classification society and a recognized advisor for the maritime industry.

World Cargo Fleet	57, 829
Total World Fleet	88, 483

Table 2.2: Number of vessels in the world fleet 2. May 2014 (Smestad, 2015).



Ship type	Number of vessels
Multi-purpose and general cargo ships	18,303
Bulk Carriers	10,053
Handysize	3,095
Handymax	3,008
Panamax	2,405
Capesize	1,590
Oil Tankers (<10,000 dwt)	7,456
Oil Tankers (>10,000 dwt)	5,830
Sub panamax	3,401
Aframax	884
UL&VLCC	624
Suezmax	495
Panamax	416
Offshore (AHTS/PSV)	5,129
Containerships	5,102
Sub panamax	3,019
Post Panamax	1,208
Panamax	875
Reefers	1,438
Ro-Ro vessels	1,311
LPG Carriers	1,258
LNG Carriers	388
Others	1,561

Figure 2.1: World fleet cargo (Smestad, 2015)

Bulk Carriers

Bulk Carrier are defined by carrying bulk cargo, such as grains, coal, ore and cement. It is about 10000 bulk carriers, which is classified into size categories. Design speed is usually between 13-15 knots (MAN Diesel & Turbo, 2013)

Oil Tankers

Oil Tankers are designed for bulk carry of oil. According to MAN Diesel & Turbo (2011) there are two basic types of Oil tankers: Crude Tankers and Product tankers, which move unrefined oil to refinery and refined oil to point near consuming markets, respectively. Oil tankers are defined by size and occupation. Design speed is normally between 13-16 knots (MAN Diesel & Turbo, 2013)

Container ships

Container ships are transporting containers, and are measured in twenty-foot equivalent unit(TEU). Container fleet contains of about 5000 ship and the design speed are between 15-25 knots (MAN Diesel & Turbo, 2011).

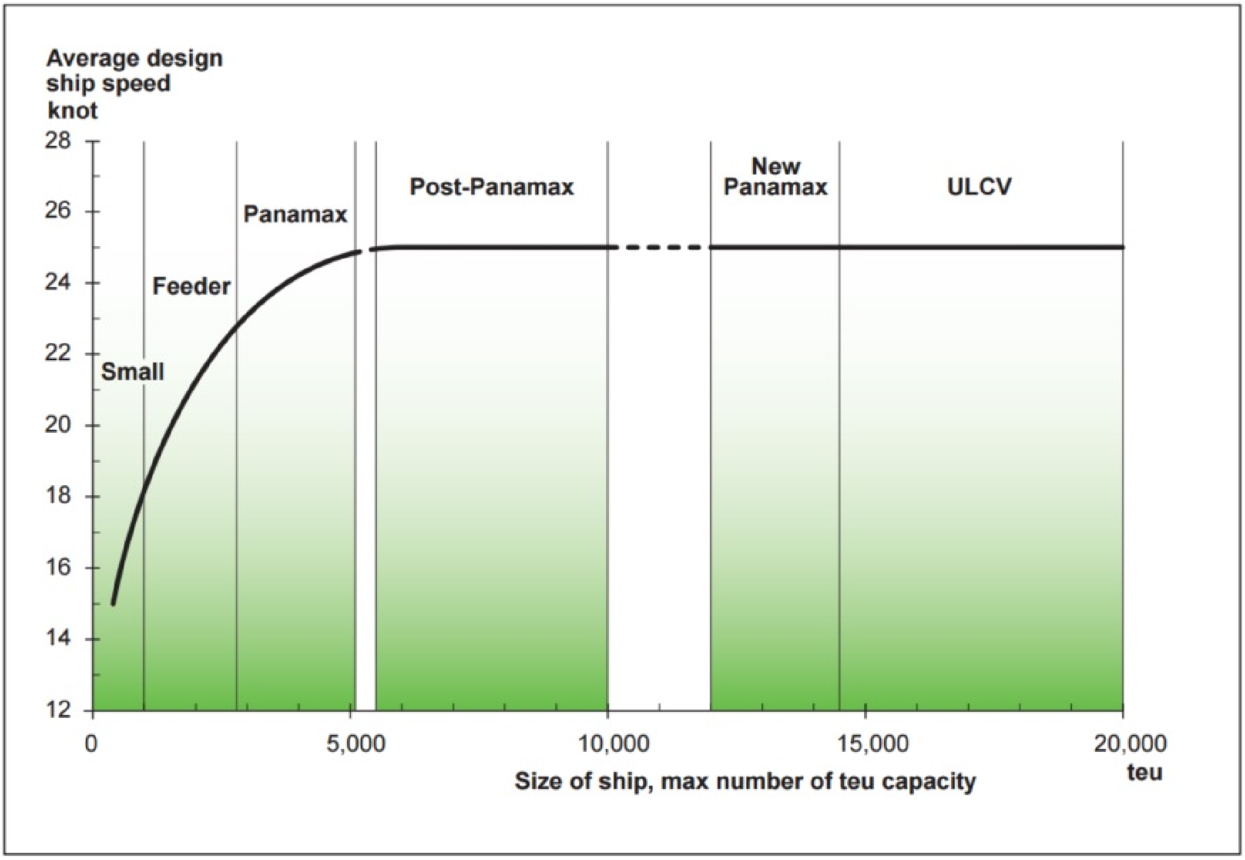


Figure 2.2: Design speed for container vessels (MAN Diesel & Turbo, 2011).

RO-RO

RO-RO ships has its name from Roll on-Roll off, and are transporting wheeled cargo, such as cars and trains.

Offshore(AHTS/PSV)vessel

Offshore vessel are vessels that handle offshore service. Vessel are designed with Dynamic Positioning[4] due to difficult working condition and high precision work. Work capabilities often require more installed power and it is often installed with a controllable pitch propeller.

---

[4] Computer-controlled system to automatically maintain a vessel's position and heading

Others

Consists of Reefers, LPG Carriers, LNG carriers (tankers in S-AIS) and other smaller vessels.

Ship type	Number of vessels
Multi-purpose and general cargo ships	18,303
Bulk Carrier	10,053
Handy size	3095
Handy Max	3,008
Panamax	2405
Capsize	7456
Oil Tankers (<10,000 dwt)	7456
Oil Tankers (>10,000 dwt)	5830
Sub panamax	3,401
Aframax	881
ULCC&VLCC	624
Suezmax	495
Panamax	416
Offshore(AHTS/PSV)	5,129
Container ships	5,102
Sub Panamax	3,019
Post Panamax	1,208
Panamax	875
Reefers	1438
Ro-Ro Vessels	1344
LPG Carriers	1258
LNG Carriers	388
Others	1561

Table 2.3: World fleet (Mantell, Benson, Stopfrod, Crowe & Gordon, 2014).

2.2.1 Ship sizes

A different solution from sizing ship types in dimensions sizes is offered in Smith et al. (2014). Table 2.4 divide ship sizes into under-groups in respect to capacity.

Vessel	**Capacity bin**	**Capacity unit**
Bulk carrier	0 - 9,999	DWT
	10,000–34,999	
	35,000–59999	
	60,000–99,999	
	100,000–199,999	
	200,000–+	
Chemical tanker	0-4,999	DWT
	5,000–9,999	
	10,000–19,999	
	20,000–+	
Container	0-999	TEU
	1,000–1,999	
	2,000–2,999	
	3,000–4,999	
	5,000–7,999	
	8,000–11,999+	
	12,000–14,500	
	14,500–+	
Cruise	0 - 1,999	GT
	2,000–9,999	
	10,000–59,999	
	60,000–99,999	
	100,000–+	
Ferry - pax only	2,000-+	GT
Ferry - ro-pax	2,000-+	GT
General cargo	0 - 4,999	DWT
	5,000–9,999	
	10,000–+	
Liquefied gas tanker	0 - 49,999	Cubic metres (m^3)
	50,000–199,999	
	200,000–+	
Oil tanker	0 - 4,999	DWT
	5,000–9,999	
	10,000–19,999	
	20,000–59,999	
	60,000–79,999	
	80,000–119,999	
	120,000–199,999	
	200,000–+	
Other liquids tankers	0-+	DWT
Refrigerated cargo	0-1,999	DWT
Ro-Ro	0-4,999	GT
	5,000–+	
Vehicle	3,999	Vehicles
	4,000–+	

Table 2.4: Vessel type and sizes (Smith et al., 2014).

2.3 Resistance

In fluid mechanics resistance is the opposing force on a moving object with respect to a surrounding object. Hull[5] resistance can be found from basic principles of ship propulsion.

2.3.1 Total Resistance

Total resistance can be divided into three parts; Frictional resistance, residual resistance and air resistance. This can further divided so that total resistance equation is:

$$\text{Total resistance, } R_T = R_V + R_W + R_A + R_{others} \tag{2.1}$$

Where:
R_T = Total Resistance
R_V = Viscosity resistance
R_W = Wave making resistance
R_A = Correlation allowance
R_{others} = (Air resistance, Appendage resistance, R_b resistance of bulbous bow, R_{rt} immersed transom stern)

---

[5] Body of ship

Viscosity resistance

Viscous resistance is the predominating resistance force for low-speed ships like bulk, carriers and tankers where it accounts for between 70% to 90% of all resistance (MAN Diesel & Turbo, 2011). Viscous resistance is mainly made out of frictional resistance.

$$\text{Frictional resistance, } R_F = C_F * 1/2 * \rho * S * V^2 \tag{2.2}$$

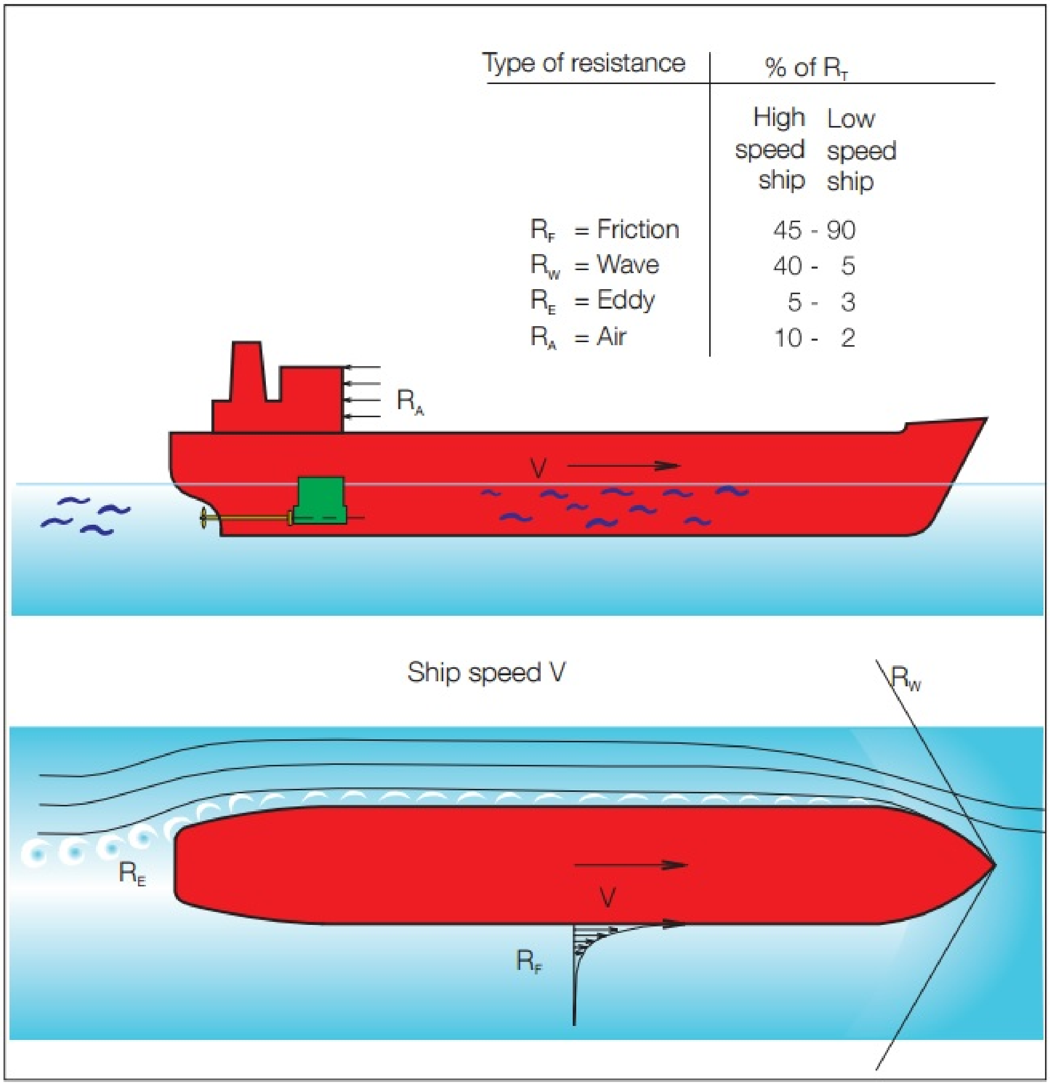
Where:
R_F = Frictional Resistance
C_F = Frictional Coefficient
S = Hull Wetted Surface
V = Ship speed

Residual resistance

Residual resistance is the resistance from waves and eddy-making, hence it is from the loss of energy cause by wave making and flow separation. Residual resistance can be found by using model testing since C_{Rmodel}=C_{Rship}. Speed of ship is the main factor for how influential residual resistance is. In a low-speed ship it represent from 8% - 25% of total resistance, while in high-speed ship it could be up to 40% - 60%. Furthermore shallow water will affect the impact from residual resistance. However, assuming seawater depth 10 times more than ship draught, residual resistance will not be influencing.

$$R_R = C_R * 1/2 * \rho * S * V^2 \tag{2.3}$$

Where:
R_R = Residual Resistance
C_R = Resistance Coefficient



Type of resistance	% of R_T	
	High speed ship	Low speed ship
R_F = Friction	45 -	90
R_W = Wave	40 -	5
R_E = Eddy	5 -	3
R_A = Air	10 -	2

Figure 2.3: Total ship towing resistance (MAN Diesel & Turbo, 2011).

Air resistance

Air resistance is given by

$$R_A = C_{AA} * 1/2 * \rho * S * V^2 \tag{2.4}$$

Where:
R_A = Air Resistance
C_{AA} = Air resistance Coefficient

Air resistance may also be based on dynamic pressure of air:

$$R_A = 0.90 * 0.5 * \rho_{air} * V^2 * A_{air} \tag{2.5}$$

Where:
A_{air} = Cross section area of the ship above the water

Navigational resistance

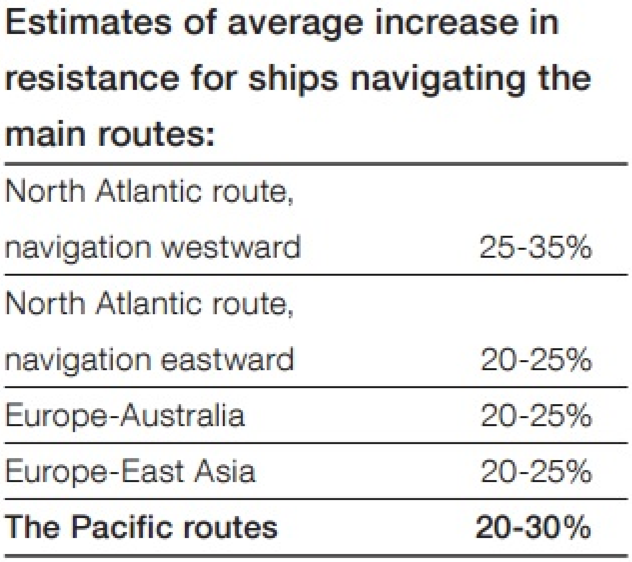
Due to sea, current and wind, an additional navigational resistance has to be added to total resistance (MAN Diesel & Turbo, 2011). In figure 2.4 estimations of increases resistance for main routes are presented. This shows the importance of navigational resistance.

2.3.2 Model coefficients

Model testing is used in an early stage for finding ship resistance. Ship model resistance is represented through towing resistance.

$$\text{Total resistance coefficient, } C_T = \frac{R_T}{0.5 * \rho * V^2 * S} = (1 + k) * C_F \text{for } F_N < 0.1 \tag{2.6}$$

Where:
F_N = Froude number



Estimates of average increase in resistance for ships navigating the main routes:

North Atlantic route, navigation westward	25-35%
North Atlantic route, navigation eastward	20-25%
Europe-Australia	20-25%
Europe-East Asia	20-25%
The Pacific routes	**20-30%**

Figure 2.4: Navigational resistance from main routes (MAN Diesel & Turbo, 2011).

Froude Number gives an understanding of the relationship between total resistance and viscous resistance. By assuming that wave resistance (due to F_N), air resistance and base drag are negligible, frictional resistance equals total resistance. $C_F = C_T$

$$\text{Froude number, } F_n = \frac{V}{\sqrt{gL_{wl}}} \tag{2.7}$$

For a ship model viscous resistance from viscosity coefficient:

$$\text{Viscosity coefficient, } C_V = C_F + k \times C_F \tag{2.8}$$

Where:
k = Form factor

Form factor can be found by model tests, numerical equations and empirical equations, hence there is several ways to approximate form factor.

This is one example:

$$\text{Form factor 1+k, } k = 19\frac{\nabla}{L \times B \times T} \times \frac{B}{L}^2 \tag{2.9}$$

$$\text{Frictional coefficient, } C_F = \frac{0.075}{(logR_n - 2)^2} \tag{2.10}$$

Where :

$$\text{Reynolds Number, } R_N = \frac{V * L_{WL}}{\nu} \tag{2.11}$$

$$\nu = 10^{-6} \text{ for } 20^\circ \tag{2.12}$$

[6] This means that frictional resistance depends on the length of the ship at waterline, L_{WL}.

$$\text{Residual resistance coefficient, } C_{Rm} = C_{Tm} - C_{Fm} \tag{2.13}$$

Correlation allowance is a factor for systematic errors in scaling method and the value C_A is between -0.15×10^{-3} and $-0.3 * 10^{-3}$ (Steen, Unknown) There are also several other resistance coefficients. For low speed ships these resistance coefficients are for the most part negligible.

$$\text{Transom stern, } C_A = \frac{0.029 * S_B/S^{3/2}}{C_F^{1/2}} \tag{2.14}$$

$$\text{Appendix Resistance coefficient, } C_{BD} = \frac{0.029 * (S_B/S)^{3/2}}{C_F^{1/2}} \tag{2.15}$$

$$\text{Roughness allowance, } \Delta C_F = [110.31 \times H \times {V_S}^{0.21} - 403.33] \times C_{Fs}^2 \tag{2.16}$$

2.3.3 Propellers

Ships are traditionally using propellers for propulsion. This can be either fixed or controllable pitch propellers and normally the ship has one or two propellers to move the ship. For fixed propellers pitch is normally 70% of D/2.

---

[6] Mean value for sea ocean temperature are varying and kinematic viscosity have a little higher value in salt water than fresh water.

There exists several types of propeller types, and their implementation depends on the ships design purpose. The propeller contains from 2 to 6 blades (z). Despite that fewer blades gives more efficiency, normally the blade number is 4 or even 5 and 6 for bigger vessel types. This is due to lack of strength in the blades as they are applied heavy loads (MAN Diesel & Turbo, 2011). Thrust power delivered by the propeller to water is given by the propeller thrust in water with a given speed, V_A.

$$\text{Thrust power, } P_T = P_E/\eta_H = V_A * T \tag{2.17}$$

where: η_H is the hull efficiency.

Propeller efficiency, η_O , can be found through an open water test carried out in a towing tank. The test measures thrust, torque and speed of advance at fixed revolution rate. This is expressed with dimensionless constants:

$$K_T = \frac{T}{\rho * n^2 * D^4} \tag{2.18}$$

$$K_Q = \frac{Q}{\rho * n^2 * D^5} \tag{2.19}$$

$$J_A = \frac{V_A}{n * D} \tag{2.20}$$

Where:
J_A: Advance number
V_A: Speed of advance
n: Revolutions per minute
D : Propeller diameter
K_T: Thrust Coefficient
K_Q: Torque Coefficient
T: Thrust
Q: Torque

The friction of the hull makes a friction belt around the hull, which causes wake in the aft part of the ship, around the propellers. This results in lower speed around the propeller area than ship speed, equal to speed of advance. Speed of advance can be found from this formula:

$$V_A = V_S(1 - w) \tag{2.21}$$

Where w is the wake fraction coefficient

The rotation of the propellers causes the water to be drawn towards the propeller, adding resistance to the propeller and causes trust reduction.

$$t = \frac{T - R_T}{T} \tag{2.22}$$

Where t is the trust deduction coefficient.

2.4 Power prediction

In the course of ship design process, power prediction for ship can be approximated for a given hull form and resistance coefficients. In addition the given characteristics of the proposed hull effective power, P_E, can be calculated.

Effective power is the power needed for pulling the hull through water.

$$\text{Effective power, } P_E = V * R_T \tag{2.23}$$

Thrust power is the power delivered by the propeller to water in a given speed. η_H is the relationship between thrust power and effective power:

$$\text{Thrust power, } P_T = P_E/\eta_B = V_A * T \tag{2.24}$$

η_B is the relationship between thrust power and delivered power to propellers.

$$\text{Delivered power, } P_D = P_T/\eta_B \tag{2.25}$$

where: $\eta_B = \eta_O * \eta_H$

Brake power of main engine is derived from the relationship between delivered power and brake power. This is the power produced from the engine to deliver a given effective power output.

$$\text{Brake power, } P_B = P_D/\eta_S \tag{2.26}$$

2.4.1 Propulsion efficiency

Propulsion efficiency is a measure of total power loss from propulsion engine to water. This is expressed with efficiency coefficients.

$$\text{Total efficiency, } \eta_T = \frac{P_E}{P_B} = \frac{P_E}{P_T} * \frac{P_T}{P_D} * \frac{P_D}{P_B} = \eta_H * \eta_B * \eta_S = \eta_H * \eta_0 * \eta_R * \eta_S \tag{2.27}$$

where:
η_B = Propeller efficiency - behind hull
η_S = Shaft efficiency
η_H = Hull efficiency
η_0 = Propeller efficiency -open water
η_R = Relative rotative efficiency

Quasi Propulsion Efficiency, η_D, is the ratio between effective power and power delivered to the propellers.

$$\text{Propulsive efficiency - open water, } \eta_D = \eta_0 * \eta_H * \eta_R = P_E/P_D \tag{2.28}$$

Propeller efficiency behind hull is the efficiency of the propeller work behind the ship.

$$\text{Propeller efficiency - behind hull , } \eta_B = \eta_0 * \eta_R = P_T/P_D \tag{2.29}$$

Open water propeller efficiency is the efficiency of the propeller work in open water.

$$\text{Propeller efficiency - behind hull } = \eta_0 \tag{2.30}$$

Efficiency of the hull described the ratio between effective power and thrust power.

$$\text{Hull efficiency, } \eta_H = P_E/P_T = \frac{R_T * V}{T * V_A} = \frac{1-t}{1-w} \tag{2.31}$$

Relative rotative efficiency comes from the water flowing to the propeller. The rotation of the water gives a beneficial effect tho the propulsion.

$$\text{Relative rotative efficiency, } \eta_R = P_E/P_T = \frac{R_T * V}{T * V_A} = \frac{1-t}{1-w} \tag{2.32}$$

Shaft efficiency, η_S, is the loss from ie. shaft and gearbox losses. This may also be expressed as mechanical efficiency. This efficiency can be from 0.96 to 0.995, but normally around 0.99. The efficiency expresses the ratio between power delivered and brake power delivered by the engine.

$$\text{Shaft efficiency, } \eta_S = P_D/P_B \tag{2.33}$$

Values for the given efficiencies will be discussed further in the next chapter.

2.4.2 Power prediction using empirical methods

There are several empirical methods for approximation of power prediction. Primarily empirical methods are used for calculating hull resistance in early stages of design phases. This includes the form factor, a way to separate viscous resistance and wave resistance and is a correction method for displaced water by the hull.

Renown empirical methods for resistance prediction are Holtrop-Menn, Guldhammer, Lap - Keller, Series-60, Hollenbach and MARINTEK's Formula. Table 2.5 is the deviation between model tests and empirical resistance approximations and shows the numerical difference between the methods (Steen, 2011). Some of the different methods are described shortly in the underlying text.

	Single-screw design draft		Single-screw ballast draft		Twin-screw design draft	
	Mean	Standard deviation	Mean	Standard deviation	Mean	Standard deviation
Holtrop-Mennen	-0.5%	12.8%	6.3%	16.1%	5.8%	18.4%
Guldhammer	-0.8%	11.0%	10.5%	17.9%	11.2%	19.2%
Lap-Keller	-0.5%	12.9%	27.9%	32.9%	14.0%	23.4%
Series - 60	-1.0%	11.6%	37.3%	42.7%	15.2%	23.3%
Hollenbach	-1.0%	9.4%	-0.2%	11.2%	3.5%	13.3%

Table 2.5: Deviation between model tests and empirical methods (Steen, 2011).

Holtrop-Mennen

Holtrop-Mennen is a method for calculating propulsive power of ships. This is done by using basic hull dimensions. Total ship resistance is divided into component for regression analysis. This was done using an extensive number of models test and trial measures (Holtrop & Mennen, 1982). From Holtrop-Mennen effective power (P_E) and resistance R_T are estimated (Holtrop & Mennen, 1982).

Guldhammer and Harvald

A ship calculation method was developed by Guldhammer and Harvald from 1965 - 1974 (Guldhammer & Harvald, 1974). Their heuristics uses an extensive analysis of published model tests, and takes relativly few parameters (Kristensen & Lützen, 2012). Residual resistance is approximated with a function using only three parameters; length/displacement-ratio, prismatic coefficient and Froude number and is given without correction of hull form, bulbous bow or position of LCB (Kristensen & Lützen, 2012).

Lap - Keller

Lap presented diagrams for determining resistance for single screw ships. This was later extended by Keller for resistance and power prediction for single screw ships (Keller, 1973; Lap, 1954).

Hollenbach

Hollenbach estimating resistance and propulsion for single screw and twin screw ships. It is base on a extensive number of model tests, and the newest published method for conventional ships (Steen, 2011).

$$C_R Hollenbach = C_{R,Standard} * C_{R,Fnkrit} \tag{2.34}$$

$$\text{Residual resistance, } C_R \text{ Hollenbach} = \frac{R_R}{\frac{\rho}{2} * V_0^2 * B * T} \tag{2.35}$$

$$C_{TS} = (C_{FS} + \Delta C_F) + \frac{B * T}{S} * C_{RHollenbach} \tag{2.36}$$

Marinteks formula for formfaktor

Formula base on experimentally decided form factors from regression. (Steen, 2011)

$$k = 0.6\phi + 145\phi^{3.5}\phi = \frac{C_B}{L_{WL}} * \sqrt{T_{AP} + T_{FP} * B} \tag{2.37}$$

2.5 Fuel consumption

Fuel cost accounts for a big part of expenses for a voyage. GHG emissions are also a direct consequence of amount of fuel used. Hence fuel consumption is paid close attention from all who have interests in shipping.

A measure for how much fuel engine uses per produced power is called specific fuel consumption (SFC). This is also called Brake specific fuel consumption. SFC is the measure of fuel efficiency for engines, in this case main engine for ships. SFC varies with speed and loads on the vessel. Figure 2.5 shows distribution of SFC compared to engine shaft power.

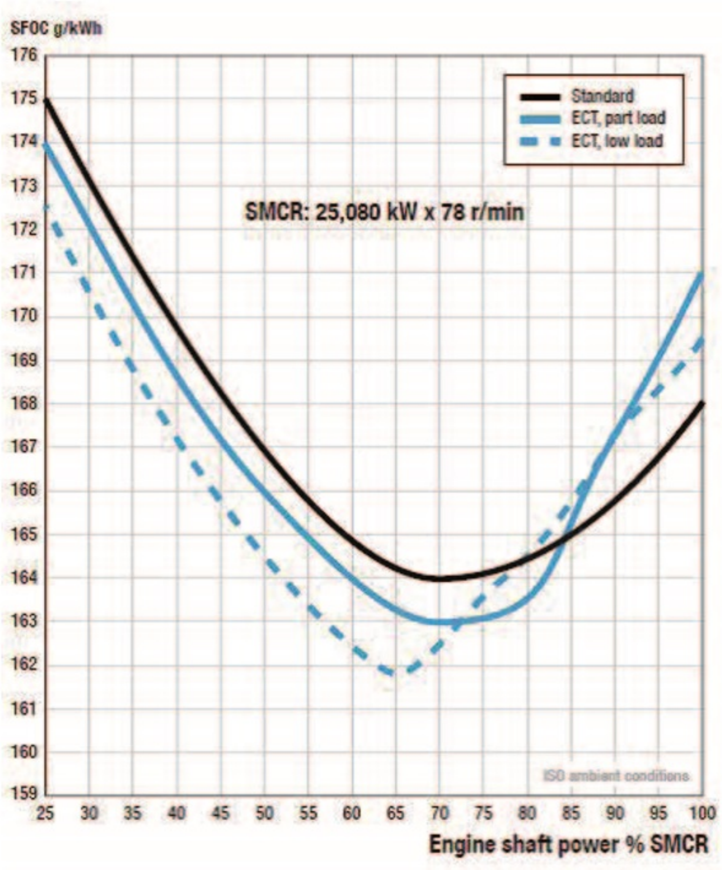


Figure 2.5: SFOC curve with engine control tuning (ECT) (MAN Diesel & Turbo, 2012).

Possible influential factors for SFC are engine type, engine rating, fuel type and wheather it meets pre-IMO tier, IMO 1 or 2 requirements (Marakogianni, Papaefthimiou & Zopounidis , 2016).

$$\text{Break specific fuel consumption, } BSFC = \frac{r}{P} \tag{2.38}$$

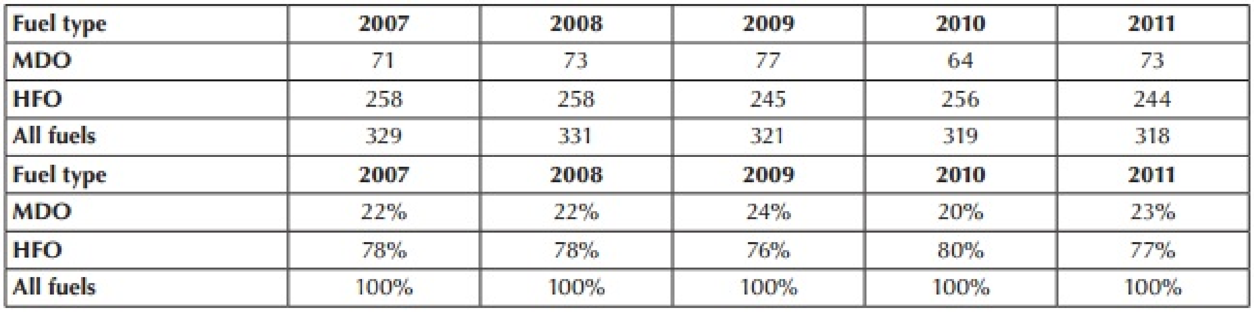
where:
r is the fuel consumption rate in grams per second (g/s)
P is the power produced in watts where $P = \tau\omega$
ω is the engine speed in radians per second (rad/s)
τ is the engine torque in newton meters (N·m)

BSFC, or SFC, is measures in g/kwh, which is the same as $3.6 * 10^6$ g/J
SFC depends on engine type size and load, and building year(as engines has become more effective), which will be discussed shortly in this section.

All cargo ships	210
Tankers	191-229
Containers	194-222
Bulk and compined carriers	192-202
General cargo vessels	200-230

Table 2.6: SFC cargo vessels (Eyring, Köhler, van Aardenne & Lauer, 2005).

Fuel types used in shipping are: Marine gas oil (MGO), Marine diesel oil (MDO), Intermediate fuel oil (IFO), Marine fuel oil (MFO) and Heavy fuel oil (HFO). These are often combined into two groups, MDO and HFO. A distribution of fuel type usage in shipping from 2007 to 2011 can be found in table 2.6.



Fuel type	2007	2008	2009	2010	2011
MDO	71	73	77	64	73
HFO	258	258	245	256	244
All fuels	329	331	321	319	318
Fuel type	**2007**	**2008**	**2009**	**2010**	**2011**
MDO	22%	22%	24%	20%	23%
HFO	78%	78%	76%	80%	77%
All fuels	100%	100%	100%	100%	100%

Figure 2.6: Upper range of top -down fuel type consumption (in million tons) (Smith et al., 2014).

This can be be compared with IEA[7] fuel sales (Smith et al., 2014). In marine sector fuel sold distinct types of fuel in 2011 for HFO equal to 177.9, MDO equal to 29.6 and LNG equal to 0 (all in million tons).Possible ways to monitor fuel consumption is to use AIS combined with ship data, bunker delivery note (BDN), installing fuel flow meters or collecting Noon reports (Faber, Nelissen & Smit, 2013).

Engine age effects efficiency. This is shown in table 2.7. It could account for as much as 10 % of SFC.

Engine age	SSD	MSD	HSD
before 1983	205	215	225
1984 -2000	185	195	205
post 2001	175	185	195

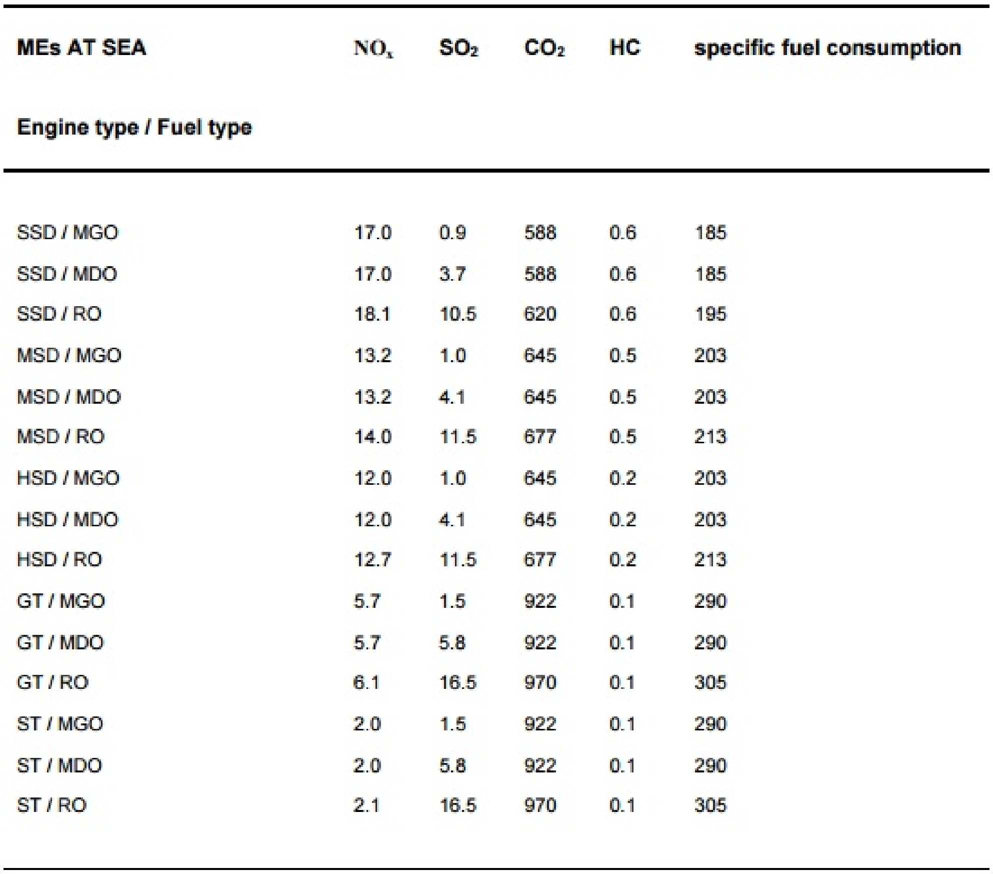
Table 2.7: SFOC (Smith et al., 2014).

Specific fuel consumption for combination of engine type and fuel type is shown in table

---

[7] International Energy Agency

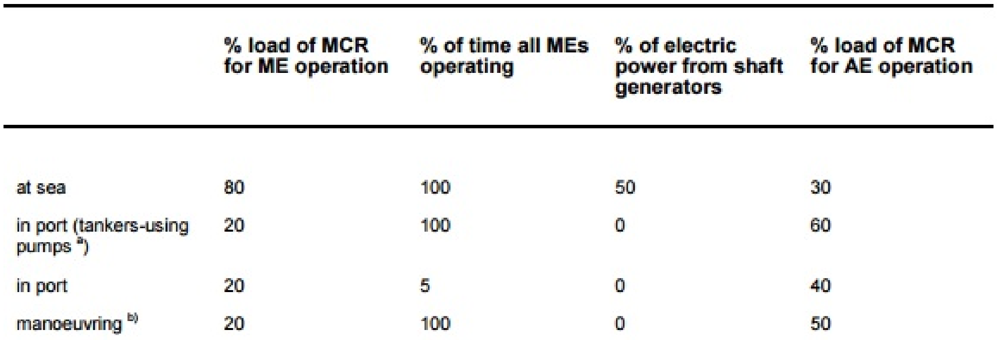
2.7, where: SSD(Slow speed diesel engine), MSD (Medium speed diesel engine), and HSD (High speed diesel engine).



MEs AT SEA **Engine type / Fuel type**	NO_x	**SO_2**	**CO_2**	**HC**	**specific fuel consumption**
SSD / MGO	17.0	0.9	588	0.6	185
SSD / MDO	17.0	3.7	588	0.6	185
SSD / RO	18.1	10.5	620	0.6	195
MSD / MGO	13.2	1.0	645	0.5	203
MSD / MDO	13.2	4.1	645	0.5	203
MSD / RO	14.0	11.5	677	0.5	213
HSD / MGO	12.0	1.0	645	0.2	203
HSD / MDO	12.0	4.1	645	0.2	203
HSD / RO	12.7	11.5	677	0.2	213
GT / MGO	5.7	1.5	922	0.1	290
GT / MDO	5.7	5.8	922	0.1	290
GT / RO	6.1	16.5	970	0.1	305
ST / MGO	2.0	1.5	922	0.1	290
ST / MDO	2.0	5.8	922	0.1	290
ST / RO	2.1	16.5	970	0.1	305

Figure 2.7: Combination engine fuel/type in ships (Whall et al., 2002).

Engines in a ship is exposed to different loads during different operations. Table 2.8 is an assumption done by Whall et al. (2002) on load distributions during different operations.



	% load of MCR for ME operation	% of time all MEs operating	% of electric power from shaft generators	% load of MCR for AE operation
at sea	80	100	50	30
in port (tankers-using pumps [a])	20	100	0	60
in port	20	5	0	40
manoeuvring [b)]	20	100	0	50

Figure 2.8: Assumptions for engine operations (Whall et al., 2002).

2.6 Emissions

This section gives a short introduction to emissions and its contributions from shipping.

2.6.1 Emissions in general and in shipping

As of year 2010 global anthropogenic emissions of GHG[8] was 49 ± 4.5 GT CO2- equivalents/year (Pachauri et al., 2014). Out of this, CO2 accounted for 76%. Fossil fuel and industrial processes accounted for 65%, and forestry and other land use for 11%. Figure 2.9 shows a growth in from 27GT in 1970 to 49GT in 2010. Consequently global anthropogenic emissions has almost doubled in this period.

---

[8] Green house gases

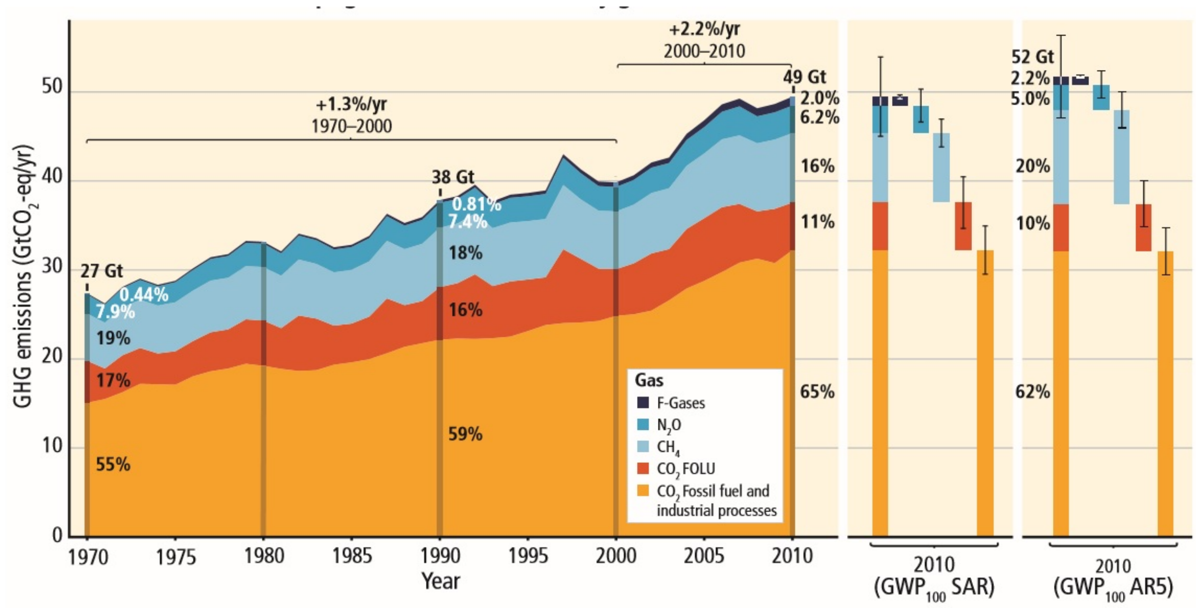


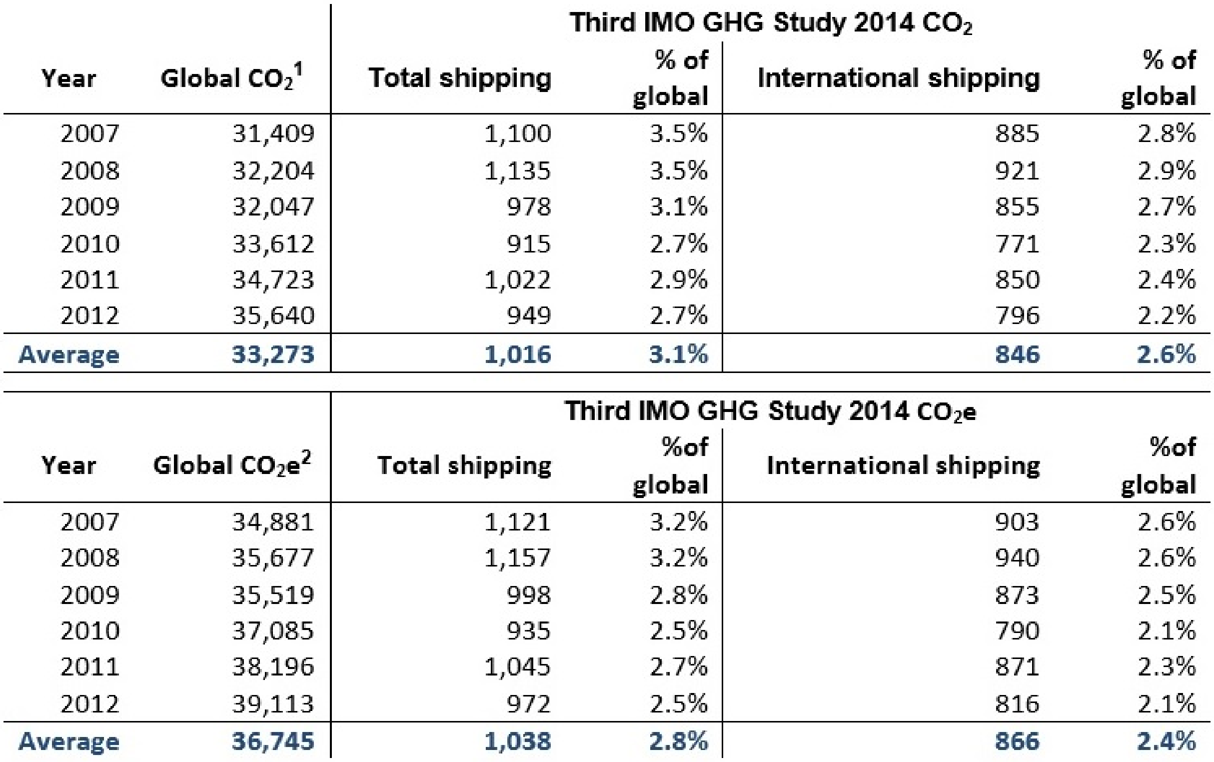
Figure 2.9: Total annual anthropogenic GHG emissions by gases 1970–2010 (Whall et al., 2002).

Further development in global emissions of GHG will depend on both socio-economic development and climate policies. The Paris Climate agreement following COP21[9], showed that there most likely will be an increase in policies with regards to GHG-emissions in the future.

From 2007 to 2012 shipping accounted for 3.1% of global GHG emissions, and 2.6% of global CO2 emissions (Smith et al., 2014). Third IMO GHG Study showed a slight reduction in emissions from the Second IMO GHG study in 2009. This can be seen in figure 2.11.

---

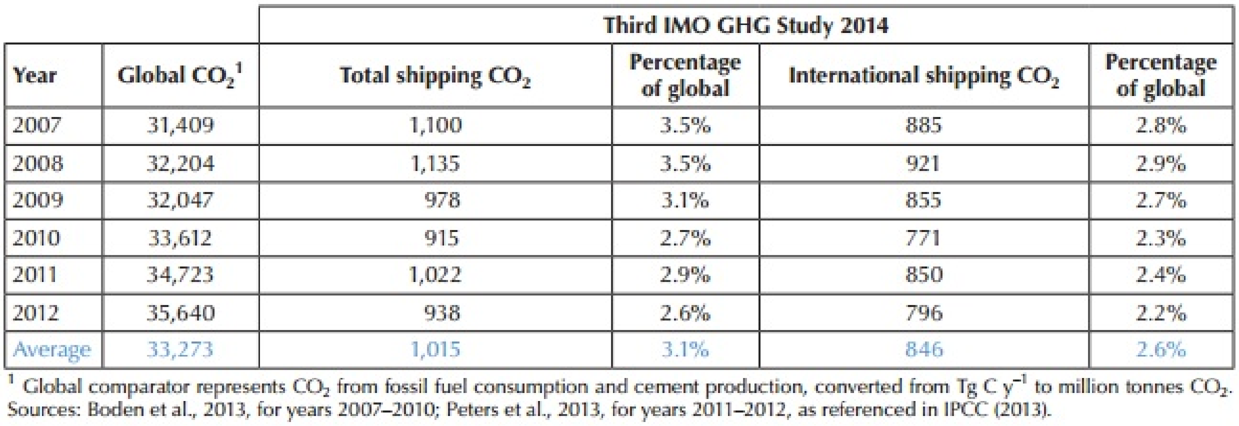
[9] United Nations climate conference, 2015



Year	Global CO_2[1]	Third IMO GHG Study 2014 CO_2 Total shipping	% of global	International shipping	% of global
2007	31,409	1,100	3.5%	885	2.8%
2008	32,204	1,135	3.5%	921	2.9%
2009	32,047	978	3.1%	855	2.7%
2010	33,612	915	2.7%	771	2.3%
2011	34,723	1,022	2.9%	850	2.4%
2012	35,640	949	2.7%	796	2.2%
Average	**33,273**	**1,016**	**3.1%**	**846**	**2.6%**

Year	Global CO_2e[2]	Third IMO GHG Study 2014 CO_2e Total shipping	%of global	International shipping	%of global
2007	34,881	1,121	3.2%	903	2.6%
2008	35,677	1,157	3.2%	940	2.6%
2009	35,519	998	2.8%	873	2.5%
2010	37,085	935	2.5%	790	2.1%
2011	38,196	1,045	2.7%	871	2.3%
2012	39,113	972	2.5%	816	2.1%
Average	**36,745**	**1,038**	**2.8%**	**866**	**2.4%**

Figure 2.10: Shipping emissions 2007-2012 (Smith et al., 2014).



		Third IMO GHG Study 2014			
Year	**Global CO_2[1]**	**Total shipping CO_2**	**Percentage of global**	**International shipping CO_2**	**Percentage of global**
2007	31,409	1,100	3.5%	885	2.8%
2008	32,204	1,135	3.5%	921	2.9%
2009	32,047	978	3.1%	855	2.7%
2010	33,612	915	2.7%	771	2.3%
2011	34,723	1,022	2.9%	850	2.4%
2012	35,640	938	2.6%	796	2.2%
Average	33,273	1,015	3.1%	846	2.6%

[1] Global comparator represents CO_2 from fossil fuel consumption and cement production, converted from Tg C y^{-1} to million tonnes CO_2.
Sources: Boden et al., 2013, for years 2007–2010; Peters et al., 2013, for years 2011–2012, as referenced in IPCC (2013).

Figure 2.11: Shipping CO2 emissions 2007-2012 (Smith et al., 2014).

2.6.2 Emission factors

Relationship between emission and fuel consumption can be found through SFOC[10].

$$\text{Emission, } EF_{baseline}(g\,pollutant/g\,fuel) = \frac{EF_{baseline}(g\,pollutant/kWh)}{SFOC_{baseline}(g\,fuel/kWh)} \tag{2.39}$$

Emissions factors were developed for GHG species by third GHG study 2014 (Smith et al., 2014). This can be used directly with fuel consumption for emission calculations. CO2 Baseline for the different fuel types used in marine shipping are shown in equations 2.40, 2.41 and 2.42. As for same fuel consumption by a distinct ship, emission for each of the GHG will be decided by the fuel type. Table 2.8 shows CO_2 emissions factor for all the different fuels in marine shipping. This is transmissible to other GHG emissions.

$$\text{HFO } EF_{baseline,CO^2} = 3,114 kgCO^2/tonnefuel \tag{2.40}$$

$$\text{MDO/MGO } EF_{baseline,CO^2} = 3,206 kgCO^2/tonnefuel \tag{2.41}$$

$$\text{LNG } EF_{baseline,CO^2} = 2,750 kgCO^2/tonnefuel \tag{2.42}$$

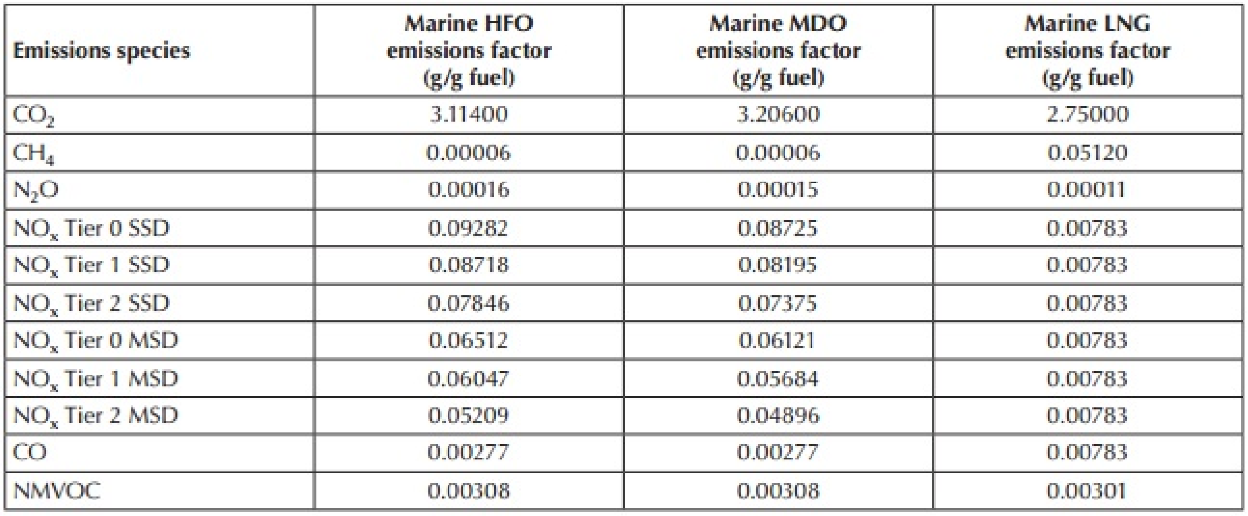
Type of fuel	Reference	Carbon Content	C_F (t-CO^2/t-Fuel)
1. Diesel/Gas oil	ISO 8217 Grades DMX through DMB	0.8744	3.206
2. Light Fuel Oil(LFO)	ISO 8217 Grades RMA through RMD	0.8594	3.151
3. Heavy Fuel Oil (HFO)	ISO 8217 Grades RME through RMK	0.8493	3.114
4. Liquefied Petroleum Gas (LNG)	Propan/Butane	0.8182 / 0.8264	3.000 / 3.030
5. Liquefied Natural Gas (LNG)		0.7500	2.750
6. Methanol		0.3750	1.375
7. Ethanol		0.5217	1.913

Table 2.8: Petroleum product CO_2 C_F factors (Smith et al., 2014).

---

[10] Specific fuel oil consumption, also named SFC and BSFC

Figure 2.12 contains emission factors for different GHG species. It shows that the amount of emissions depends on both type and species.



Emissions species	Marine HFO emissions factor (g/g fuel)	Marine MDO emissions factor (g/g fuel)	Marine LNG emissions factor (g/g fuel)
CO_2	3.11400	3.20600	2.75000
CH_4	0.00006	0.00006	0.05120
N_2O	0.00016	0.00015	0.00011
NO_x Tier 0 SSD	0.09282	0.08725	0.00783
NO_x Tier 1 SSD	0.08718	0.08195	0.00783
NO_x Tier 2 SSD	0.07846	0.07375	0.00783
NO_x Tier 0 MSD	0.06512	0.06121	0.00783
NO_x Tier 1 MSD	0.06047	0.05684	0.00783
NO_x Tier 2 MSD	0.05209	0.04896	0.00783
CO	0.00277	0.00277	0.00783
NMVOC	0.00308	0.00308	0.00301

Figure 2.12: Emissions factor for GHG (Smith et al., 2014).

2.6.3 Climate changes

Climate changes poses a significant risk for human and natural systems. There is strong scientific evidence of climate change that largely is caused by human activities. Global warming is closely linked to climate changes such as:rising sea water levels,increases in intense rainfall events and decrease in snow cover and sea ice. Furthermore global warming are linked to more frequented intense heat waves, increases in wildfires, longer growing seasons and ocean acidification (Matson et al., 2010).

2.6.4 Emission prevention

MARPOL

MARPOL is the main convension for prevention of pollution from ship at sea and was adapted by IMO[11] in 1973. It is meant to cover pollution from operational and accidental causes. As of October 2. 1983 the MARPOL convention entered into force, after the 1978 Protocol absorbed the 1973 Convension (IMO, 2011). Today's MARPOL includes the 1997 protocol and includes in total six Annexes.

- Annex 1 - Prevention of pollution of oil
- Annex 2 - Control of pollution by noxious liquid substance carried in bulk
- Annex 3 - Prevention of pollution by harmful substances carried by sea in packaged form
- Annex 4 - Prevention of pollution by sewage from ships
- Annex 5 - Prevention of pollution by garbage from ships
- Annex 6 - Prevention of air pollution from ships

MARPOL convention needs to be ratified by more than 50 percent of world fleets collective GT to be officially valid, which is also the case for all six annexes.

EEDI

EEDI [12] is a mandatory technical measure by MEPC [13] of the IMO organization. EEDI was adopted by MARPOL ANNEX 6 in 2011, and put in place to reduces GHG[14] from ships. This a legal binding treaty, the first since Kyoto Protocol from 1997. The EEDI has requirement for different size and ship segments to follow a minimum of efficiency in level per capacity

---

[11] International Maritime Organization

[12] Energy Efficiency design index

[13] Marine Environment Protection Committee

[14] Green House Gases

mile. This is requirements for new ship design. The level of efficiency is adjusted higher every five year. EEDI is accordingly a mean for technological and operational development.

$EEDI = a * b^c$ (GL, 2013). EEDI Calculation formula for CO_2 emissions:

$$EEDI_{attained} = \left(\overbrace{(\prod_{j=1}^{n} f_j)(\sum_{i=1}^{n_{ME}} P_{ME(i)} * C_{FME(i)} * SFC_{ME(i)})}^{\text{Main Engine(s) } CO_2 \text{ emissions}} + \overbrace{(P_{AE} * C_{FAE} * SFC_{AE}) + ((\prod_{j=1}^{n} f_j * \sum_{i=1}^{n_{PTI}} P_{PTI(i)} - \sum f_{eff(i)} * P_{AEeff(i)} * C_{FAE} * SFC_{AE})}^{\text{Auxiliary engine(s) } CO_2 \text{ emissions}} - \overbrace{(\sum_{i=1}^{n_{eff}} f_{eff(i)} * P_{eff(i)} * C_{FME} * SFC_{ME})}^{CO_2 \text{ emission reduction due to Innovative technology(s)}} \right) \underbrace{\frac{1}{f_i * f_l * f_w * f_c * Capacity * v_{ref}}}_{\text{Transport work}} \tag{2.43}$$

Emission reduction

IMO agreement on technical regulations is mandatory for all ships and enter into 94% of world fleet. Introduction of SEEMP[15], which was adapted in 2013, improve efficiency in a cost effective manner (IMO, 2011). This helps ship owners to actively manage ship and fleet efficiency over time. Within 2020 industry goal is 20% per tonne/km CO2 reduction and within 2050 industry is 50% per tonne/km.[16] (ICS, 2014). Whereas goals for reduction is set, shipping emissions are predicted to increase between 50% - 250% within 2050 (Smith et al., 2014).

Here are a number of suggestions found from ICS (2014) for which reductions can be made:

---

[15] Ship Energy Efficiency Management Plan

[16] Compared to 2005

bigger ships to improve fuel efficiency, better ship operational measures(for instance speed management), reduced fuel consumption with SEEMP and alternative fuel sources.

European Parliament's framework for reduction of CO_2 emissions from maritime transport can be found in MRV[17] Regulation adopted on April 29. 2015. This demands all big ships using EU ports to obey the MRV Regulations from start of 2018 (Whall et al., 2002).

2.7 Ship measurements

Describing size and capacity of a ship can be done by linear dimensions or tonnage. Linear dimensions are in three dimensions; length, breadth and depth. In this case the most important linear dimensions for a ship is Length on Waterline, L_{WL}, Breadth (also named Beam), B, and Drought, D. These dimensions are critical for a ships performance in water, as resistance from water constitutes majority of total resistance. In addition ship hull has several other dimensions which are shown in figure 2.13 below.

Length

Overall length (L_{OA}) is the absolute length of the ship. The length on waterline (L_{WL}) is the length from aft to fore at the waterline and the length between perpendiculars (L_{pp}) is the length from fore to aft perpendicular.

Breadth

Breadth or Beam (B) is the absolute breadth of the ship hull. Breadth Moulded (B_m) is the breadth measured inside the inner shells of plating.

---

[17] Monitoring, reporting and verification

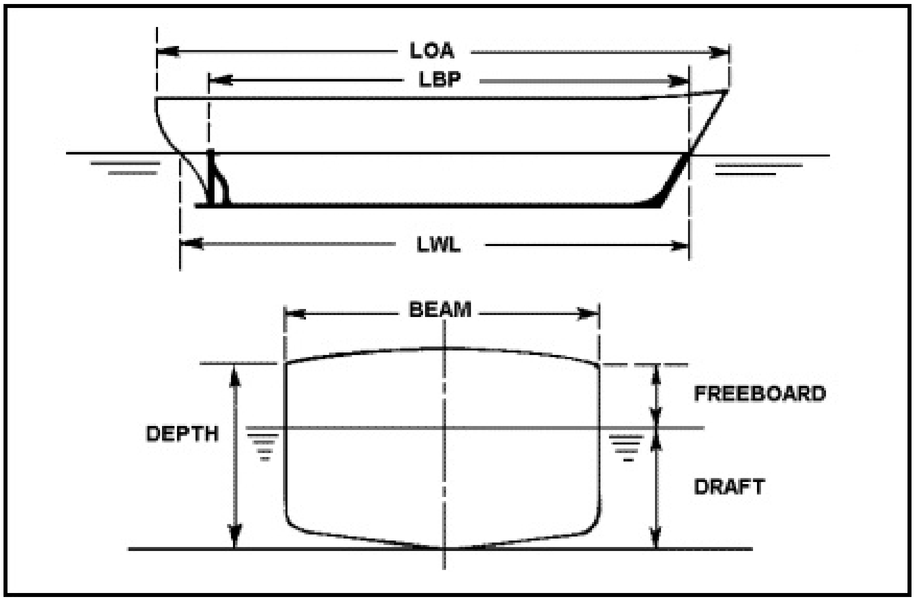


Figure 2.13: Hull dimensions (MarineStudy, 2015).

Draught

Draught moulded on Fore Perpendicular (T_F) is the draught at the fore of the ship where the perpendicular is. Draught moulded on Aft Perpendicular (T_A) is the draught at the aft of the ship where the perpendicular is.

Weight and volume

Displacement(∇) is the suppressed volume of the ship. This can also be expressed as weight. Deadweight (D_{WT}) is a ships carrying capacity, and is the difference between lightweight and displacement loaded. Lightweight is a ships weight without cargo, crew, fuel, passengers etc. and displacement refers to the weight of water pushed away by the ships hull.

Positions at ship hull

Longitudinal center of buoyancy is where the centroid of the displayed water in the horizontal direction. Center of bulb area above keel (h_T) is the distance from keel to the center of bulb in horizontally.

Area, shape of hull and coefficients

Transverse bulb area (A_{BT}) is the area of bulb in direction of the breadth. Transom area (A_T) is the area aft of the ship. Submerged area is the part which is of interest. Wetted area of hull (S) is the area of the hull which is submerged. Stern shape (C_{STERN})is the shape of aft most part of the ship expressed as a coefficient. Wetted area of appendages (S_{APP}) is the total wetted area of all appendages.

Propeller

Propeller diameter is the length from the circle made from a rotating propeller. Clearance propeller with keel is the vertical distance from propeller to keel. Number of propeller blades can be between 2-6. A normal number of propeller blades is four.

2.7.1 Hull form

Shape of the hull is given trough different coefficients describing different part of ship hull. This is useful when designing ship hull for calculating hull resistance, loading of ship etc Considerations to be taken with regards to hull form is hull load in different services, hence an understanding of hull form is necessary to know the significance of effects from speed and displacement.

Block coefficient C_B

C_B is the relationship between hull displacement volume and volume of the dimensions from waterline. See figure 2.14

$$C_{B_{WL}} = \frac{\nabla}{L_{WL} * B_{WL} * D} \tag{2.44}$$

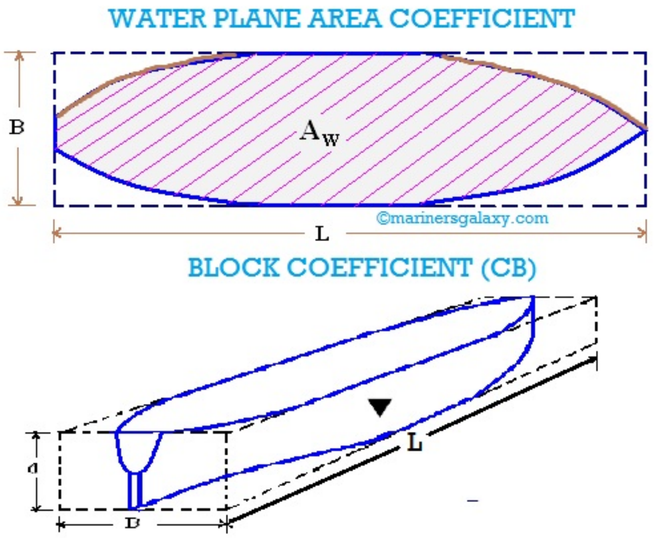


Figure 2.14: Hull form coefficients (Sharma, 2015).

Longitudinal prismatic coefficient C_P

C_P is the ratio between hull displacement volume and the product of the midship frame section area and the waterline length.

$$C_P = \frac{\nabla}{L_{WL} * A_X} \tag{2.45}$$

Midship section coefficient C_M

C_M is the relationship between the immersed midship section area and the aft perpendiculars:

$$C_M = \frac{A_M}{A_M * L_{WL}} \tag{2.46}$$

Waterplane area coefficient, C_{WP}

C_{WP} is the ratio between the ship's waterline area and the product of the breadth and the length of the ship on the waterline,

$$C_{WP} = \frac{A_{WL}}{L_{WL} * B_{WL}} \tag{2.47}$$

3 | Heuristic for ship emissions calculation based on AIS

The objective for this thesis is to make a simplification of emissions calculation. AIS data provides information about ship main characteristics in addition to position and voyage related messages. This information shall be used with power prediction methods. Furthermore this is used for calculation of fuel consumptions. From fuel consumptions, emissions will be derived and gathered in a global inventory ship emissions calculation. A descriptive overall flow chart for ECAIS model is found in figure 3.1.

Emissions calculation from AIS data (ECAIS) method estimates fuel consumption by using Holtrop-Mennen for ship resistance and power prediction. Holtrop-Mennen takes main ship characteristics as input. Some of these characteristics are given directly from AIS. Rest of the characteristics are found through literature survey which gives an approximate value for given ship sizes and types. From ship resistance power prediction is derived. An approximate fuel consumption table based on the estimated engine power are applied, Ship speeds given from AIS data are used for finding the specific fuel oil consumption for each distinct ship. Computer scripts are applied for calculations for all AIS messages. This gives a an approximated ship consumption for the given data input, and from this emissions are derived.

Total CO2 emission can be found from fuel consumption together with emission factor for CO2. From the use of empirical methods, the heuristic is expected to show some inaccuracy in emission calculation. Those inaccuracies will have to be put into context from earlier inaccurate methods for emission calculation, as well as with ship design methods.

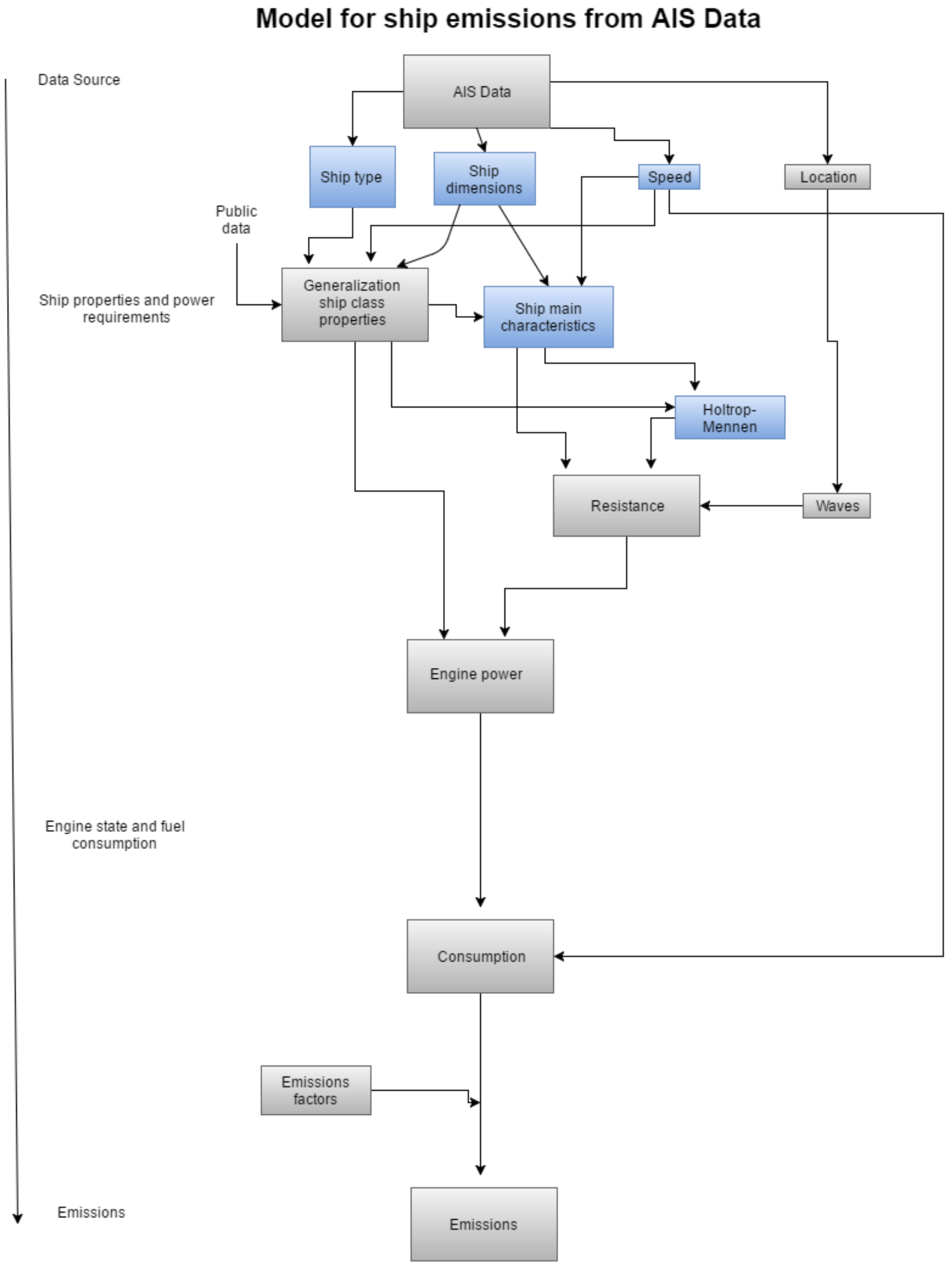


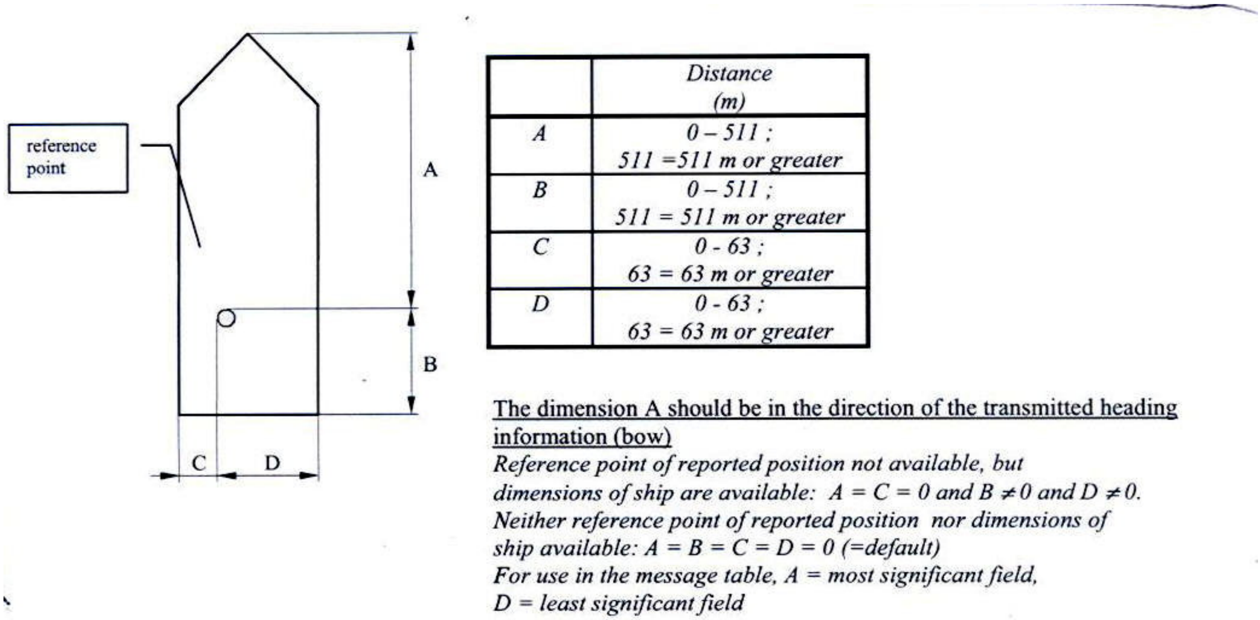
Figure 3.1: Ship emissions model from AIS data

3.1 Holtrop-Mennen calculations

Holtrop-Mennen is a well recognized power prediction estimation method, and for that purpose chosen for this theses calculation. The objective is calculate effective power needed for each individual ship from the S-AIS data collection, which can be applied in fuel consumption calculations.

AIS inputs

To make use of Holtrop-Mennen for resistance calculations and power prediction several inputs are needed. As AIS cannot provide all of the inputs directly a heuristic for the required inputs are derived from ship classifications and design rules.



	Distance (m)
A	*0 – 511 ;* *511 =511 m or greater*
B	*0 – 511 ;* *511 = 511 m or greater*
C	*0 - 63 ;* *63 = 63 m or greater*
D	*0 - 63 ;* *63 = 63 m or greater*

The dimension A should be in the direction of the transmitted heading information (bow)

Reference point of reported position not available, but dimensions of ship are available: A = C = 0 and B ≠ 0 and D ≠ 0.
Neither reference point of reported position nor dimensions of ship available: A = B = C = D = 0 (=default)
For use in the message table, A = most significant field, D = least significant field

Figure 3.2: Static ship dimensions from AIS (MCA, 2016).

3.1.1 Ship characteristics

In this section the main characteristics is described. Using Holtrop-Mennen on a ship fleet that includes characteristics normally found in ship databases, requires assumptions for several of the main characteristics and hull coefficients that are used to derive these characteristics.

Ship type		Container Ship	Container ship	Container ship	Bulk carrier
Capacity	TEU	1300	5000	9000	-
Deadweight	DWT	20355	54240	103000	75000
mass displacement	t	26780	76780	145200	82470
Displacement	m^3	26110	74900	141700	84530
Speed	kn	18	22	25	15
Emptyship mass	t	6430	22540	4200	7470
LxBxT	m	152x25.2x11	271x36.5x12	319x44x14.5	210x33x14.1
H	m	14.3	22.9	26.4	18.8
Fn		0.24	0.23	0.24	0.17
Cb		0.62	0.626	0.62	0.86
Cm		0.975	0.978	0.98	0.975
Cp		0.636	0.625	0.69	0.86
S	m^2	5166	11419	12632	10820
Cf		$3.09 * 10^-3$	$2.575 * 10^-3$	$1.32 * 10^-3$	$1.147 * 10^-3$
Rt	kN	1052	1760	2690	802.7
Ne	kW	1550	40000	69200	10000

Table 3.1: Values for hull characteristics (Charcharlis, 2013).

Block coefficient, C_B

C_B can be described as an essential coefficients with regards to resistance, and is used here for calculating several of the main characteristics in Holtrop. It describes the difference between the hull form and the volume of waterline dimensions.

$$\text{Block coefficient, } C_B = \frac{\nabla}{L_{WL} * B_{WL} * T} \tag{3.1}$$

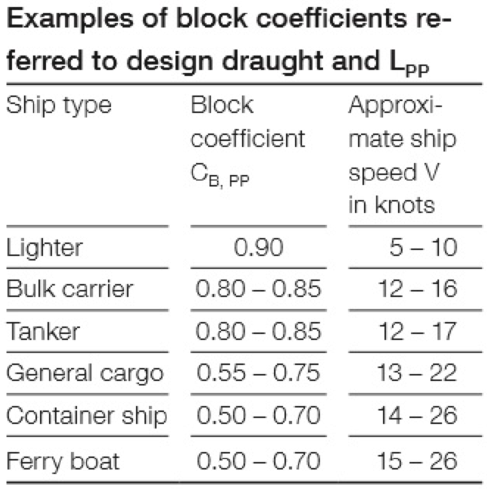
Since displacement is not known from AIS messages, C_B is assumed through literature surveys. There exist several estimation methods, which certainly can be implemented in this method. However, for simplicity of the method, C_B is chosen as a fixed number for each main ship type described in Smestad (2015). Although ship fleet includes ships with a significant variety of dimensions, a single block coefficient is chosen for each ship group. This is done by using median of coefficient range shown in figure 3.3. which were found in MAN Diesel & Turbo (2011). Further literature surveys substantiates coefficients chosen. Examples given are ABS (2013) and Takahashi (2006). This estimation disregards values of ship dimensions and speeds, hence it is likely that deviation from measured block coefficients are significant. However the size of the fleet is believed to make up for that deviation. As for these assumption for block coefficient, further calculations also disregards differences in displacement between voyages.

Block coefficients are chosen as followed:

- LNG = 0.72
- Bulk Carrier = 0.825
- Container Ships = 0.60
- Oil Tankers = 0.825

As the estimation method below uses the ship specific dimensions, further development of the ECAIS-method might include a more narrow estimation. Hence the global formula by Barrass (2004) is shown:'

$$C_B 1.20 - 0.39(V/L^0.5) \tag{3.2}$$



Examples of block coefficients referred to design draught and L_{PP}

Ship type	Block coefficient $C_{B,\,PP}$	Approximate ship speed V in knots
Lighter	0.90	5 – 10
Bulk carrier	0.80 – 0.85	12 – 16
Tanker	0.80 – 0.85	12 – 17
General cargo	0.55 – 0.75	13 – 22
Container ship	0.50 – 0.70	14 – 26
Ferry boat	0.50 – 0.70	15 – 26

Figure 3.3: Block Coefficient example (MAN Diesel & Turbo, 2011).

Midship section coefficient, C_M

As earlier described, midship section coefficent is the ratio between the midship area from waterline and the product of breadth at waterline and draught.

$$\text{Midship coefficient },\ C_M = A_M/(B_{WL} * T), \tag{3.3}$$

Since we do not have the midship area for each ship, an approximation method is used to calculate this coefficient:

$$\text{Benford formula, } C_M = 0.977 + 0.085 * (C_B - 0.60) \tag{3.4}$$

Different estimations for midship section coefficient (Charcharlis, 2013) are shown below:

$$C_M = 0.979 \tag{3.5}$$

$$\text{Equation of Schneekluth and Bertram, } C_M = 1.006 - 0.0056 * C_B^{-3.56} \tag{3.6}$$

$$\text{Jensen Equation , } C_M = (1 + (1 - C_B)^{3.5})^{-1} \tag{3.7}$$

$$\text{Norid Equation , } C_M = 0.928 + 0.080 * C_B \tag{3.8}$$

Prismatic Coefficient, C_P

Prismatic coefficient describes the vertical distribution of the ships hull. Since AIS messages does not distribute midship area, midship section coefficient provides a solution for the prismatic coefficient. This calculation is used as a part of Holtrop-Mennen method:

$$C_P = \nabla/(C_M * B * L * T) \tag{3.9}$$

Length on waterline, L_{WL}

Length on waterline is the length of the ship where it sits in the water. L_{WL} is a percent of ship overall length, which is the length given by AIS messages. As there was not found any studies about the ratio between those lengths, a suggested 97% is used during calculation.

$$L_{WL} = L_{OA} * 0.97 \tag{3.10}$$

Length between perpendiculars, L_{PP}

L_{PP} is the length between perpendiculars and are described here as a percentage of L_{WL}. In MAN Diesel & Turbo (2011) suggested ratio for conventional hulls between L_{PP} and L_{WL} is about 97%. This is also used in the ECAIS method.

$$L_{PP} = L_{WL} * 0.97 \tag{3.11}$$

Breadth moulded, B

Breadth moulded is the maximum beam , normally amid ship. This method uses AIS which contains the distance from the AIS instrument to both port and starboard. Hence the sum of those values gives breadth.

Draught moulded on F.P. and A.P, T_F and T_A

Draught moulded in fore perpendicular is the depth from waterline to flat keel. Draught moulded in aft perpendicular is the depth from waterline to flat keel. For conventional ships, when loaded draught fore and aft of the ship is equal. This value is draught is normally considered as when having summer load. Draught is given by AIS as ten times the breadth, hence it is divided with ten when used in calculations.

Displacement volume moulded, ∇

Displacement of the hull is the water that the hull suppress. This is given by the volume of a block dimensions of the hull and the block coefficient. Volume can be directly derived form AIS data, while C_B must be found from literature with regards to the ship type given by the proposed heuristics of Smestad (2015).

$$\Delta = C_B * L_{WL} * B_{WL} * T \tag{3.12}$$

Longitudinal center of buoyancy

Longitudinal center of buoyancy is normally found in shipping fleet ships behind half of the ships length. This is because of main engine place and weight compared to rest of the ship.

$$lcb = -0.75 * (L_{WL}/2.0)/100.0 \tag{3.13}$$

Transverse bulb area, A_{BT}

Transverse bulb area is found using the ratio between itself and the midship area. Moreover suggested 8% of midship area was found in Charcharlis (2013).

$$A_{BT} = 0.08 * A_M \tag{3.14}$$

Center of bulb area above keel line, h_B

As there was found no literature on this subject center of bulb area above keel line for this method are expressed as a ratio between itself and ships draught. This ratio was found by using values from the original example of Holtrop & Mennen (1982).

$$h_B = 0.4 * D \tag{3.15}$$

Waterplane area coefficient, C_{WP}

Waterplane area coefficient are using the dependency of the block coefficient at maximum draught, found in Kristensen & Lützen (2012).

$$C_{WP} = 0.55 + 0.45 * C_B \tag{3.16}$$

Other approximations found, but not used:

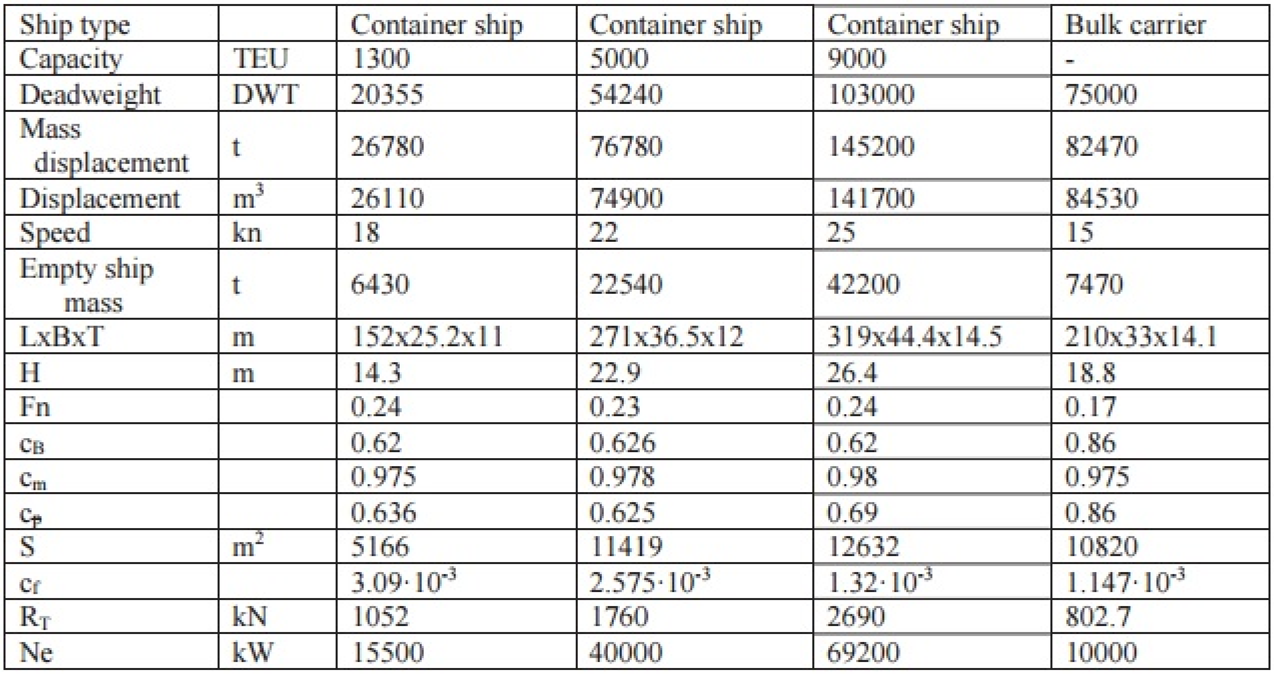
$$C_{WP} = \frac{A_{WL}}{L_{WL} * B_{WL}} \tag{3.17}$$

$$\text{Schhneekluth's equation: } C_{WP} = (1 + 2 * C_B)/3 \tag{3.18}$$

Transom area, A_T

Transom area was described as a ratio between itself and midship area. It was found no literature for a ratio, hence ratio from example given in Holtrop & Mennen (1982) was chosen. This was found to be 0.051.

$$A_T = 0.051 * A_M = 0.051 * C_M * Breadth * Draught \tag{3.19}$$



Ship type		Container ship	Container ship	Container ship	Bulk carrier
Capacity	TEU	1300	5000	9000	-
Deadweight	DWT	20355	54240	103000	75000
Mass displacement	t	26780	76780	145200	82470
Displacement	m^3	26110	74900	141700	84530
Speed	kn	18	22	25	15
Empty ship mass	t	6430	22540	42200	7470
LxBxT	m	152x25.2x11	271x36.5x12	319x44.4x14.5	210x33x14.1
H	m	14.3	22.9	26.4	18.8
Fn		0.24	0.23	0.24	0.17
c_B		0.62	0.626	0.62	0.86
c_m		0.975	0.978	0.98	0.975
c_p		0.636	0.625	0.69	0.86
S	m^2	5166	11419	12632	10820
c_f		$3.09 \cdot 10^{-3}$	$2.575 \cdot 10^{-3}$	$1.32 \cdot 10^{-3}$	$1.147 \cdot 10^{-3}$
R_T	kN	1052	1760	2690	802.7
Ne	kW	15500	40000	69200	10000

Figure 3.4: Values for propulsion coefficients

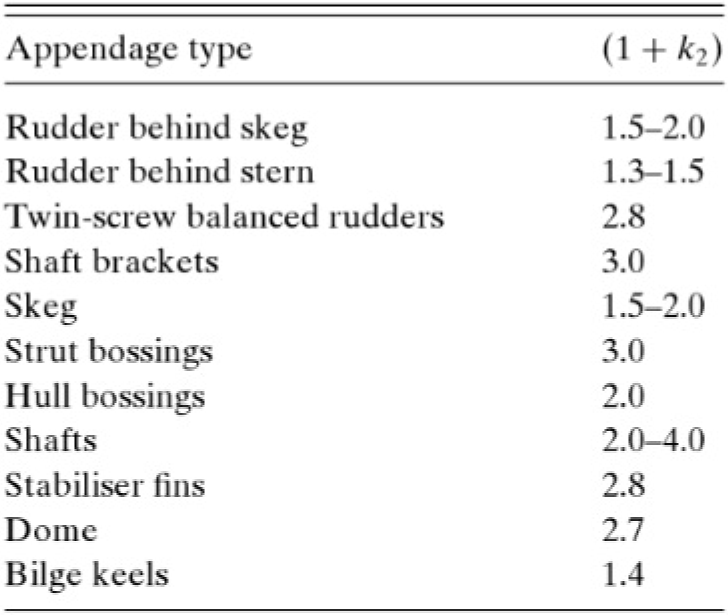
Values for propulsion coefficients (Charcharlis, 2013)

Appendage area

Appendages area was chosen to the same value as example given in Holtrop & Mennen (1982). This value is $50m^2$.

Appendages form factor

Appendages form factor was of simplicity set to 1.5, same as example used by Holtrop & Mennen (1982). The values for each appendage can be found in figure 3.5



Appendage type	$(1 + k_2)$
Rudder behind skeg	1.5–2.0
Rudder behind stern	1.3–1.5
Twin-screw balanced rudders	2.8
Shaft brackets	3.0
Skeg	1.5–2.0
Strut bossings	3.0
Hull bossings	2.0
Shafts	2.0–4.0
Stabiliser fins	2.8
Dome	2.7
Bilge keels	1.4

Figure 3.5: Appendages form factor
Appendages form factor (Molland, 2011)

Stern shape parameter, C_{STERN}

Stern shape parameter was set to 10 described as a U shaped section with Hogner stern, same as example given in Holtrop & Mennen (1982).

- -25 for pram with gondola
- -10 for Vshaped section
- 0 for normal section ship
- 10 for U shaped section with Hogner stern

Propeller diameter, D

Propeller diameter can be found from the ration between draught and diameter. It was found in literature that a expected value of less than 0.65 for Bulk Carrier and Tanker, and a value of less than 0.74 for container ships (MAN Diesel & Turbo, 2011). ECAIS method sets these values to ships within the classification given by Smestad (2015). Other ships not covered by this classification was given the ratio of 0.7.

Number of propeller blades , Z

Number of blades are as described in chapter 2. Normally it is between 4 and 6 on merchant ships, even if fewer blades gives higher efficiency. This is due to strength of the propeller blades(MAN Diesel & Turbo, 2011). Number of blades are set to 4 for all ships calculated by the ECAIS method.

Propeller clearance with keel line

Minimum clearance for construction of new single screw hull was found in DNV GL (2016). This was used as a standard for all ships in this method.

$$\text{Clearance propeller with keel line} = (0.48 - 0.02 * Z) * Radius \tag{3.20}$$

Screw number is set to single screw for all ships, which gives the value 0.2 in Holtrop & Mennen (1982)

3.2 Fuel consumption

Effective power are found in Holtrop-Mennen using ship speeds. Brake power from engine is required to calculate fuel consumption. It is shown in chapter 2 how propulsive efficiencies can be used to calculate the required power from engine engines to achieve appropriate effective power.

$$\text{Brake power, } P_B = \frac{R_T * V}{\eta_T * 0.85} \tag{3.21}$$

where:
R_T is the total resistance
V=Ship speed $\eta_T = \eta_H * \eta_O * \eta_R * \eta_S$
0.85 = Sea margin[1]

---

[1] Factor taking into account extra power required because of rough conditions at sea.

Open water propeller efficiency accounts for a substantial part of total efficiency. In literature there are several ways to calculate propeller efficiencies. For ECAIS method numbers from Wageningen series are applied as a fixed number for different ship groups. Found in figure 3.6 median for cargo ships is approximately equal to 0.65, and approximately 0.58 for tanker ships. This is applied to ships categorized into these groups by heuristics found in Smestad (2015). LNG ships, Bulk Carriers and Container ships are here defined as cargo ships. Furthermore other efficiencies are of less substantial since efficiencies at close to 1.0. Size of η_S depends of propeller shaft length, gearbox and number of bearings.

For shaft systems including a gearbox, numbers found in literature varies between 0.93 to 0.97. For a system directly mounted to propeller a range from 0.98 to 0.99 is found. Applied in ECAIS method, η_S is set to 0.98. Range of values for η_R varies between 0.95 to 1.07. This depends on the shape of hull and number of propellers. For single propeller ships it ranges from 1.00 up to 1.07. For simplicity η_R is assumed to be 1.0 for all ship types in this heuristic. Hull efficiency is found from calculations:

$\eta_H = \frac{1-t}{1-w}$

where: t and w are found from Holtrop-Mennen approximationsHoltrop & Mennen (1982).

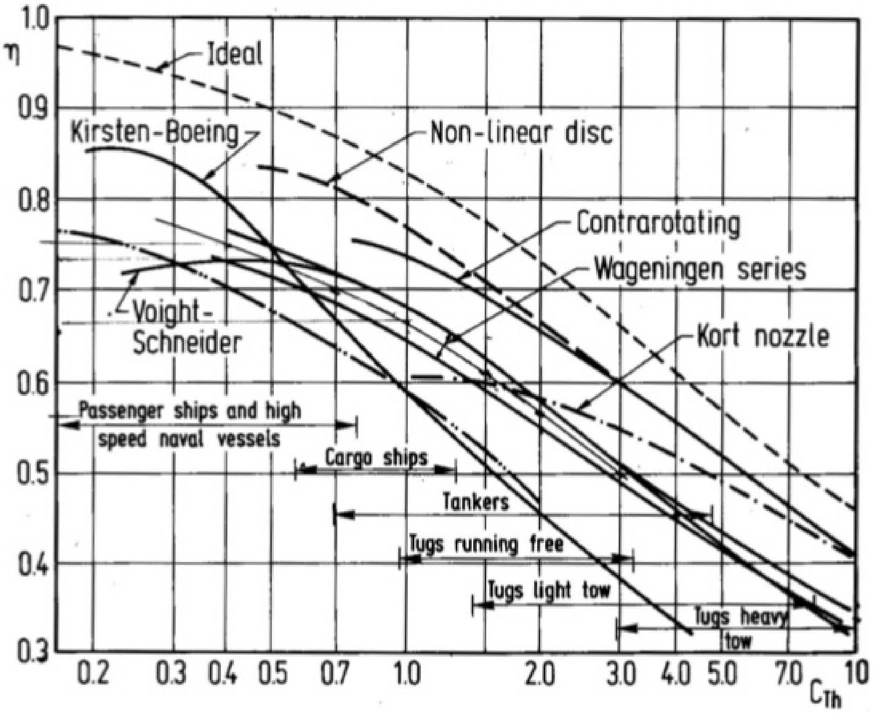


Figure 3.6: Open water efficiency (Kristensen & Lützen, 2012).

For each ship group a specific fuel consumption is given:

- LNG = 215.0 g/kwH
- Container Ships = 208 g/kwH
- Bulk Carrier = 197 g/kwH
- Tanker = 210 g/kwH

This is the median from numbers in 2.6 which was found in MAN Diesel & Turbo (2011)

Fuel consumption:

$$FC = SFC * T * P_B \tag{3.22}$$

where:
T = time span between AIS messages given by a ship.

Fuel consumption is given in tons.

3.3 Emissions

Emissions is derived from fuel consumption, which is explained in Third IMO GHG study (Smith et al., 2014).

$$EF_{baseline}(g\,pollutant/g\,fuel) = \frac{EF_{baseline}(g\,pollutant/kWh)}{SFOC_{baseline}(g\,fuel/kWh)} \tag{3.23}$$

Conversion factor, C_F

Specific baseline emissions factors are used depending on type and fuel type. It is also important to notice that there is various factors for different GHG species. Conversion factor for the list below is found from Smith et al. (2014) using HFO fuel. HFO fuel is by far the most used fuel type in marine sector (6:1 compared to MDO (Smith et al., 2014)), and factors for MDO often near. As of this reason, factors from HFO are used for the whole fleet.

- Carbon dioxide (CO2) = 3.114
- Nitrogen oxides (NOx) = 0.0903
- Sulphur oxides (SOx) = 0.025
- Particulate matter (PM) = 0.00728
- carbon monoxide (CO) = 0.00277
- Methane (CH4) = 0.00006
- Nitrous oxide (N2O) = 0.00015

- Non-methane volatile organic compounds (NMVOC) = 0.00308

Note that different emissions factors can be derived from various studies, i.e. Methodology for Calculating Emissions from Ships, written by Swedish Methodology for Environmental Data's (Smestad, 2015).

Fuel correction factor

Fuel correction factors is used by Smith et al. (2014) to allow for the different fuel types, hence the FCF should be taken into consideration when evaluation emissions numbers. As for now, FCF is not included in the emissions calculation.

Adjusting with FCF,

$$EF_{actual}(g\,pollutant/g\,fuel = EF_{baseline}(g\,pollutant/g\,fuel) \times FCF \tag{3.24}$$

3.4 Ship boundary constraints

Ship type constraints used in this heuristic as proposed in Smestad (2015) :

LNG Carrier Group

Maximum draught:	13 m
Maximum change in draught:	3.5 m
Maximum speed:	>= 16 kn
AIS Ship type:	80-89 (tanker)

Table 3.2: LNG carriers group

LNG Carrier types

Ship size	Breadth (m)	Length (m)
General group	40-42	270-300
Q-Flex	48-50	314-316
Q-Max	46-54	344-345

Table 3.3: LNG carriers

Container Ship Group

Maximum speed:	>= 15.9 kn
AIS Ship type:	70-79 (cargo ship)

Table 3.4: Container ship group

Container ship Panamax types

Size category	Length (m)	Breadth (m)	Draught (m)	Maximum change of draught (m)
Panamax vessel 1	210-269.9	31-33	13	5.5
Panamax vessel 2	270-300	31-33	14	5.5

Table 3.5: Panamax Container

Container ship types

Size category	Length (m)	Breadth (m)	Maximum change of draught (m)	Maximum speed (kn)	AIS Ship type
Post Panamax	270-315	40-43	5.5	>= 15.9 kn	70-79
New Panamax	320-370	46-52	5.5	>= 15.9 kn	70-79
Post New Panamax	380-397	54-58	5.5	>= 15.9 kn	70-79
Trippel E	397-401	58-61	5.5	>= 15.9 kn	70-79

Table 3.6: Post Panamax, New Panamax, Post New Panamax and Trippel E

Bulk carriers group

Bulk carrier	
Maximum speed:	<= 15.0 kn
AIS ship type:	70-79(cargo group)

Table 3.7: Bulk carrier ship group

Bulk carrier Panamax type

Length (m)	Breadth (m)	Minimum draught (m)	Minimum change in draught (m)
180-250	30-34	5	5.5

Table 3.8: Panamax Bulk Carrier

Bulk carrier types

Ship category	Length (m)	Breadth (m)	Minimum change in draught (m)	Maximum speed (kn)
Capsize	320-320	36-50	5	15
Handymax	160-180	29-33	5	15
Handysize	130-180	20-29	5	15

Table 3.9: Capsize, Handymax and Handysize

Tanker ship group

Oil Tanker Group	
Maximum speed	<= 16.0 kn
AIS ship type:	80-89(tanker)

Table 3.10: Oil Tanker ship group

ULCC & VLCC ships

ULCC & VLCC	
Maximum draught:	25 m
Minimum draught:	10 m
Maximum change in draught:	8 m
Breadth (m)	50-70
Length (m)	320-400

Table 3.11: ULCC & VLCC

Tanker ship types

Size category	Length (m)	Breadth (m)	Minimum change in draught (m)	Maximum speed (kn)
Suezmax	265-320	45-50	5(max 20 m draught)	16
Aframax	235-265	38-44	0	16
Panamax	200-235	30-33.5	4	16

Table 3.12: Suezmax, Aframax and Panamax

3.5 Computer program build up

Raw Satelite AIS from Norwegian Coastal Service spanning from May 1, 2014 to September 15, 2014 is decoded and put into a SQLite [2] database. Python [3] was used to do decode this raw data.

For the problem in hand it was developed a program with two classes, a Holtrop class and a Ship class, and it was also written in Python programming language

---

[2] SQL, Structured Query Language is a programming language specifically made to retrieve data from databases. Its development is controlled by the International Electrotechnical Commission and the International Organization for Standardization, ISO. SQLite is a free software library that powers databases that use SQL ($http://www.sqlite.org$).

[3] Python is a programming language that can be found at https://www.python.org/.

It contains four files. First file is a main method, which includes the main query and the total emission calculations. Moreover this query has constrains where used to remove false/disrupted AIS data. This includes length constraint which was used for comparing results if adjusted. Message type 5 file includes a MessageType5 class. For different AIS messages, variables are fetched and passed on to ship class. Ship characteristics are found in static messages from Messages Type 5 in AIS. Speed, which include average speed and max speed, are fetched from dynamic messages, which is found in Message Type 1, 2 & 3.

When ship objects has received required values, they are used by Holtrop class for calculation of resistance and power requirement for different speeds. Power prediction is sent back to Ship class, which uses preset SFOC from ship group to calculate each distinct fuel consumption. Furthermore fuel consumption is used together with distinct emission factors for ship emission for different emission types. Fuel consumption and emission for all ship classes are fetch in main method and summed up in total consumption and emission for fleet in query.

4 | Results

In this chapter results from calculations for fuel consumption and emissions are presented. Furthermore these results are used in comparison to other data found.

4.1 Calculations for original ship heuristics

Total consumption was calculated by utilizing the presented ECAIS method. AIS messages from May 1. 2014 to September 15. 2014 was obtained from Kystverket[1]. This data was decoded and processed for uses in emissions calculation. Results are presented in table 4.1.

Description	Results	Units
Total ships calculated	15,987	(-)
Total ships rejected	4,529	(-)
Ships in total	20,516	(-)
Total estimated fuel consumption	35,420,583	(tons)
Total CO2 emissions	110,299,695	(tons)
Total CH4 emissions	2,125	(tons)
Total N20 emissions	5,313	(tons)
Total NOx emissions	3.198,479	(tons)
Total NMVOC emissions	109,095	(tons)
Total CO emissions	98,115	(tons)
Total PM emissions	257,862	(tons)
Total SO2 emissions	885,515	(tons)

Table 4.1: Ship emissions calculation with original constraints

---

[1] Norwegian Coastal Administration

4.1.1 Adapted constraints

A second calculation was done using new constraints. Max speed for Bulk carriers was set to 16.0 (kn) and max speed for Tankers was set to 18.0 (kn). All other constraints are kept as previous. Results can be found in table 4.2

Description	Results	Units
Total ships calculated	19013	(-)
Total ships rejected	1503	(-)
Ships in total	20516	(-)
Total estimated fuel consumption	39,091,541	(tons)
Total CO2 emissions	121,731,060	(tons)
Total CH4 emissions	2,345	(tons)
Total N20 emissions	5,864	(tons)
Total NOx emissions	3,529,966	(tons)
Total NMVOC emissions	120,402	(tons)
Total CO emissions	108,283	(tons)
Total PM emissions	284,586	(tons)
Total SO2 emissions	977,288	(tons)

Table 4.2: Ship emissions calculation with adapted constraints

4.2 Distributions

This consumption was reviewed closer for consumption distribution by ship types. Results is presented in table 4.3.

Ship group	**Ship type**	**Ship count**	**Total consumption** (tons)	**Avg. consumption** (tons)	**Avg. displacement** (tons)
Bulk carrier		6,001	**4,536,946**	**756**	**55,448**
	Handysize	10	2,766	277	22,419
	Handymax	11	6,605	600	38,397
	Panamax	1,069	1,196,415	828	49,777
	Capsize	371	433,302	1,167	126,106
	None of above	4,163	2,897,859	696	54,008
Container Ships		8290	**26,348,514**	**3,178**	**49,876**
	Panamax Container 1	748	2,730,521	3,650	45,443
	Panamax Container 2	279	1,510,843	5,415	61,101
	Post Panamax	416	2,893,069	6,954	81,505
	New Panamax	285	3,022,325	10,604	134,544
	Post New Panamax	4	74,071	18,518	187,888
	Trippel E	10	174,283	17,428	195,842
	None of above	4,163	2,897,859	2,434	43,902
LNG Carrier		1069	**3,434,086**	**3,212**	**56,915**
	General Group	31	141,786	4,573	86,911
	Q-Flex	26	226,384	8,707	119,335
	Q-Max	10	116,187	11,619	139,319
	None of above	1002	2,949,730	2,944	53,545
Oil Tankers		3,654	**4,743,751**	**1,299**	**88,886**
	Panamax	208	267,922	1,288	61,814
	Aframax	552	720,382)	1,305	93,079
	Suezmax	311	489,741	1,575	130,577
	ULCC & VLCC	327	987,417	3,020	256,229
	None of above	2256	2,949,730	1,011	60,353
Ships outside ship groups		1,485	**-**	**-**	-
None		16	**-**	**-**	-

Table 4.3: Distribution of AIS

4.3 Case study

This section compare results from ECAIS method with real consumption from selected ships in our S-AIS collection. As these consumptions numbers are difficult to apprehend, it counts only for a small number of ships. A comparison is made between their actual consumption and consumption calculated in ECAIS method. A total of 10 ships are compared. This is presented in table 4.4.

Ship number	**Deviation**
Ship 1	+13.91 %
Ship 2	-0.70 %
Ship 3	-4.62 %
Ship 4	+5.45 %
Ship 5	+11.22 %
Ship 6	-1.78 %
Ship 7	-7.02 %
Ship 8	-20.10 %
Ship 9	+3.35 %
Ship 10	+15.33 %
Total	**-5.19 %**

Table 4.4: Fuel consumption vs actual consumption in percent

5 | Discussion

In this chapter results from fuel consumption and emissions calculation are discussed.

5.1 Case studies

This section addresses the different cases that were showed in Chapter 4. A discussion the different cases is presented with its own subsection.

5.1.1 Third IMO GHG study comparison

Total number of ships in database in hand was 47089 ships. This was reduced to 20516 by adding constraints to program query. This is approximately 2/5 of world cargo fleet, found from table 2.2. All ships below 130 meter was not considered, since heuristics presented by Smestad (2015) was limited to ships above this length. Also, ships above 460 meters was not taken into account since larger ship has never been built. Furthermore, it was noticed that AIS messages contained erroneous MMSI and IMO numbers. Messages did not consist of correct number of digits. These messages could not be considered, although these messages may concern ships within ship heuristics. From the ships evaluated by the constructed computer program, 4529 ships was rejected. This makes out 22.1% of total ships evaluated by ship heuristics. This is a fairly high number of ships rejected, and would account for a significant uncertainty for total fuel consumption.

Overall consumption was calculated to 35,420,583 tons of HFO fuel **??**. Fuel oil statistics from IEA shows that in 2011 it was sold 178.9 million tons of marine fuel in shipping, having a relatively steady sale over several years (Smith et al., 2014). Total estimated fuel consumption compared to one year of sales are about 19.8%. This is as mentioned earlier, in a time period of 5 and a half month during summer. A further review on sailing days will be done in next subsection. Emissions from this consumption was found directly from emissions factors and are also presented in table **??**.

5.1.2 Adapted constraints

During test runs with different length constraints, it came visible that amount of rejects were a sizable share of evaluated ships**??**. Tankers were found to be overrepresented in rejected ships. Although samples picked out matched the heuristic for AIS ship type, it failed at max speed test. Some had speeds above 100 knots, that is clearly not correct. Others had a slightly higher max speed than 16 knots, mostly up to around 18.0 knots and in some occasions 18.5 knots. It is claimed here that a ship with a calculated consumption is better than no consumption numbers, if purpose is to calculate emissions derived from consumptions. While a ship of course can give a presumptive wrong value of fuel consumption, it is far more likely that no consumption will make a more sizable impact on the total consumption number. Hence the ship constraints should try to include as many ships as possible. This must be if a group of ship is not within the constraints. The heuristic should instead mitigate its restrictions, although ships might wrongly be misplaced in another ship group. For this instance it may be that LNG Carriers is wrongfully identified as Tanker, although the overall performance for calculating fuel consumption improves.

Same test are performed for Ore Carriers and Bulk Carriers, with same results. Most values are within the given constraints, while a few are outside. This makes the program reject the ships that should clearly add to the total fuel consumption. A new proposal for max speed for Bulk Carrier is introduced, setting boundary as up to 16.0 (kn) from up to 15.0 (kn). Max speed for Tankers is set from below 16.0 (kn), to below 18.0(kn)

Max speeds of more than 100 knots are still excluded, although rejected ships might well be a real ship. This is most likely messages that has been wrongfully set in some way. Calculation of ships with max speed more than 100 knots will be most likely be much higher than it should, and instead should be added afterwards.

From the ships evaluated by the constructed computer program only 1503 ships was rejected. This makes out 7.3% of total ships evaluated by ship heuristics. A sizable difference from the original heuristics. Of the rejected ships 1485 of those ships did not match any ship groups in the heuristics. 2 ships was rejected with breadth equals 0, 13 ships was rejected for having registered no speeds, and 3 was rejected for having breadth more than half of ships length. Overall consumption was calculated to 39,063,298 tons of HFO fuel. This is 21.8% all fuel consumption compared to sales registered by IEA (Smith et al., 2014).

Findings from calculations was compared with numbers found in G from Third IMO GHG study. Numbers are from 2012. Since only average deadweight is given in these figures, a connection between displacement and deadweight was found from Kristensen (2013). As displacement is the sum of lightweight and deadweight combined, using lightweight factors with regards to deadweight, it was possible to compare the two results. Factor ranges between 0.07 to 0.17, while for this selection of ships the range is smaller. Most ships dealt with in this thesis will be in the area 0.08-0.10. As numbers for LNG was not found, factors for Bulk carriers was used instead. As container ships was measured as TEU in Smith et al. (2014), there was not done any calculation for this group.

The comparison of ECAIS and Smith et al. (2014) shows limited coinciding numbers. If it can be assumed that consumption rate is equal over a whole year, ECAIS calculates only between 20% to 45% of IMO calculations for Bulk Carriers. For Oil Tankers results show results between 43% to 76% of compared numbers. For the last group, LNG, calculated results where from 214% to 298% of IMO calculations. There were a difference between average deadweights in Smith et al. (2014) and compared result from ECAIS. However, this difference is not similar to contrast between fuel consumption.

Although numbers from Smith et al. (2014) are from year 2012, a relatively steady consumption rates from past years makes comparison for fuel consumption of different years feasible.

As size categories are somewhat organized slightly different it was difficult to compare average deadweight and number of ships in each category directly. Fleet size from ECAIS was also compared with table 2.3. Adjusted comparable fleet number for Mantell, Benson, Stopfrod, Crowe & Gordon (2014) was calculated as 19570 ships. Fleet calculated by ECAIS method came to 19015 ships. Compared in groups, Tankers and Bulk carriers give smaller numbers in ECAIS, while Container ships and LNG are greater. Although ship groups did not correlate, total fleet size for ECAIS and Mantell, Benson, Stopfrod, Crowe & Gordon (2014) was comparable.

5.1.3 Real fuel consumption comparison

These ships does not represent a wide range of of ship types, hence the comparison may be constricted to a specific ship type. Results show a overall good match with real fuel consumption, although results varies for distinct ships from +20% to -15%. As number of ships is only 10, it was impossible to conclude on this result.

5.2 Discussion of method in general

5.2.1 AIS Data

For this thesis it has only been conducted research for one data set. This AIS data set was for the period May 1. 2014 to September 15. 2014 and contained 47089 distinct ships. For a proper evaluation of ECAIS method, a research of more than one data set should be conducted, and contain a continuous period for more than 365 days. As these data only covers 5 and a half month of messages, comparison will be affected by the different in summer and winter season. Furthermore, quality of AIS as a data source could not be properly tested with regards erroneous messages without being compared to other data sets.

However, quality of this S-AIS data was tested during calculations. A check for false IMO(7 digits) and MMSI(9 digits) numbers was carried out. This showed that there was 9150 distinct ships which contained erroneous IMO or MMSI numbers. During calculations more erroneous messages was discovered. **?** number of ships did not contain average or max speed, hence they did not have any reports of speed. If a report from static messages is picked up so should speeds from dynamic messages be. Moreover it was found that some messages contained speed values of more that 100 knots, which can not be the case for any larger ship constructed. A few ships were also registered with a length of more than 460 m, which is the size of the biggest ship ever built.

5.2.2 Applying Holtrop-Mennen with

From table 2.5 calculated values using Holtrop-Mennen returns a mean value of -1.0 % greater than model tests, with a standard deviation of 12.8 %. This is the closest mean value to model tests, and the reason for applying this empirical method instead of other mentioned. As Holtrop-Mennen uses ship characteristics not available in AIS data, approximations had to be conducted to be able to carry out the research. This suggests that deviation from actual consumption will be greater than initially mentioned. Holtrop-Mennen was perceived as applicable for this type of computational research, although sources of error were found in the process.

5.2.3 Ship characteristics

As a part of Holtrop-Mennen, ship characteristics are applied as input for performing power calculations. AIS messages only reports of length, breadth, draught and speed as inputs used in Holtrop-Mennen. Remaining characteristics are either attempted to derived from this, or from a literature survey. Some characteristics had plenty of research adequate for what was trying to find. Other characteristics where more difficult. Some characteristics was in the end done by a calculated guess. More research would have to be done to find better approximations for characteristics needed. A characteristic that pinched out as an important number for other calculations, was Block Coefficient. Implementing this for coefficient for each ship type could be suggestions for further work, as this was only chosen as a median for a range for each ship group.

5.2.4 Propulsion efficiencies

Propulsion efficiencies represents a sizable share of brake power. Especially a variation in open water propeller efficiency will have decisive impact on results. These numbers are approximated numbers for ship groups taken from literature studies and should be developed further to adopted the ship types described.

5.2.5 Fuel and efficiency

Fuel type consumed in marine traffic showed a over-representation of HFO. This was considered when choosing to apply features from HFO to all ship classes. Further development of this method would include fuel correction, as implemented in Third IMO GHG study. Specific fuel consumption was available available in different research. A mean value for each group was selected, and used in calculation for fuel consumption. These values were a approximation for all ships represented in ship group. A further development of this would include dividing ships by their engine type and fuel, and to include ship age.

5.2.6 Emissions factors

Emissions factor were directly obtained from Third IMO GHG Study (Smith et al., 2014), and considered as correct if applied with correct engine type and fuel.

5.2.7 Sea margin

As sea margin is a sizable addition to resistance, it was included in power estimation. A sea margin of 15% was chosen for all ships, obtained from literature studies. Initially it was thought of using position for calculation sea margin, while only having limited time for this research, geographical position was not accounted for in calculations. For different areas of ship routes ships experience various weather conditions, there is considerable differences between upstream and downstream, and headwind and tailwind will represent a difference in ship resistance. Further work should include this work for more reliable sea margins.

5.2.8 Suggestions for further work

Improvements for more accurate fuel consumptions, hence emissions calculations can be done. Development of inputs applied, for each ship type could enhance results. This will also allow further research for distinct ship types.Furthermore, development of ship heuristics should include reducing rejected ships as this accounts for more tha 10 % of ship fleet used in calculations. Mentioned above is sea margin, which represents a sizable uncertainty for fewer number of ships. Ship characteristics, including installed power, SFC, age and fuel type could also be included.

This method could be applied for use in smaller, more specific areas. with selective targeting of flag type, positioning, ship types, dates and time would provide new research opportunities. In this method all other activities than service was disregarded. A further development should include port and maneuvering consumption. This could be applied with engine usage factors found in Third IMO GHG study. This includes axillary engines.

Lastly, to improvement this method, studies should be performed on different data sets: This can be used to compare results, with probable

6 Conclusion

Results show a sizable difference from Third IMO GHG study. As this study has only been made for a limited number of data, calculations contains substantial uncertainties which should be investigated further. Further improvements for ECAIS method has been emphasized, which is believed to improve results.

References

ABS (2013). *Ship Energy Efficiency Measures, Status and Guidance*. Downloaded 24.6.2016 from: https://www.eagle.org/eagleExternalPortalWEB/ShowProperty/BEA%20Repository/References/Capability%20Brochures/ShipEnergyEfficiency.

Barrass, B. (2004). *Ship design and performance for masters and mates*. UK: Butterworth-Heinemann.

Charchalis, A. (2013). *Designing constraints in evaluation of ship propulsion power*. Poland: Gdynia Maritime University.

DNV GL (2016). *AIS – Meant for navigational safety, used for business intelligence*. Downloaded 6.6. 2016 from: https://www.dnvgl.com/maritime/energy-efficiency/automatic-identification-system-data-insights.html

European Commission (2011). *Commision Directive of 23 February 2011 amending Directive 2002/59/EC of the European Parliament and of the Council establishing a Community vessel traffic monitoring and information system*. Downloaded 24.6.2016 from: http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A32011L0015

Eyring, V. Köhler, H.W., van Aardenne, J. & Lauer, A. (2005). Emissions from international shipping: 1. The last 50 years. *Jounal of Geophysical Research*, *110*(17).

Faber, J., Nelissen, D. & Smit, M. (2013). *Monitoring of bunker fuel consumption Delft*. The Nederlands: CE Delft.

GL (2013). *Rules for Classification and Construction Additional Rules and Guidelines*. Downloaded 3.5.2016 from: http:$//www.gl-group.com/infoServices/rules/pdfs/gl_vi-13-1_e.pdf$

Guldhammer, H.E. & Harvald, S.A. (1974). *Ship resistance, effect of form and principal dimensions*. Kopenhagen: Akademisk forlag.

Holtrop, J. & Mennen, G.G.J. (1982). An approximate power prediction method. *International Shipbuilding Progress*, *29*(335).

ICS (2014). *Shipping, World Trade and the Reduction of CO2 Emissions , United Nations Framework Convention on Climate Change (UNFCCC)*. Downloaded 24.6.2016 from:

http://www.ics-shipping.org/docs/default-source/resources/policy-tools/shipping-world-trade-and-the-reduction-of-co2-emissionsEE36BCFD2279.pdf?sfvrsn=20

IMO (2002). *Guidelines for the installation of a shipborne automatic identification system (AIS).* Downloaded 14.6.2016 from: http://www.navcen.uscg.gov/pdf/AIS/

IMO (2003). *Guidelines for the installation of a shipborne automatic identification system (AIS).* Downloaded 15.5.2016 from: http://www.imo.org/en/OurWork/Safety/Navigation/Documents/227.pdf

IMO (2011). *Marine Environment Protection Committee (MEPC) – 62nd session: 11 to 15 July 2011.* Downloaded 15.5.2016 from: http://www.imo.org/en/MediaCentre/MeetingSummaries/MEPC/Pages/MEPC-62nd-session.aspx

IMO (2016). *Prevention of Air Pollution from Ships.* Downloaded 24.6.2016 from: http://www.imo.org/en/OurWork/Environment/PollutionPrevention/AirPollution/Pages/Air-Pollution.aspx

ITU (2014). *Technical characteristics for an automatic identification system using time division multiple access in the VHF maritime mobile frequency band.* Geneva: ITU.

Jalkanen, J.P., Brink, A., Kalli, J., Pettersson, H., Kukkonen, J., & Stipa, T. (2009). A modelling system for the exhaust emissions of marine traffic and its application in the Baltic Sea area. *Atmospheric Chemistry and Physics*, *9*(23), 9209-9223.

Keller, W.H.Auf ‘m (1973). Extended Diagrams for Determining the Resistance and Required Power for Single Screw Ships. *International Shipbuilding Progress*, *20*225.

Kristensen, H.O., & Lützen, M. (2012). Prediction of Resistance and Propulsion Power of Ships. *Clean Shipping Currents*, *1*(6).

Kristensen, H.M. (2013). Statistical Analysis and Determination of Regression Formulas for Main Dimensions of Container Ships based on IHS Fairplay Data.*Statistical Analysis and Determination of Regression Formulas for Main Dimensions of Container Ships based on IHS Fairplay Data.* Denmark: University of Southern Denmark

Lap, A.J.W. (1954). Diagrams for Deter- mining the Resistance of Single Screw Ships. *International Shipbuilding Progress*, *1*4.

MAN Diesel & Turbo (2011). *Basic Principles of Ship Propulsion*. Denmark: MAN Diesel & Turbo

MAN Diesel & Turbo (2013). *SFOC Optimisation Methods For MAN B&W Two-stroke IMO Tier II Engines*. Denmark: MAN Diesel & Turbo

MAN Diesel & Turbo (2013). *Propulsion Trends in Tankers*. Denmark: MAN Diesel & Turbo

Mantell, C., Benson, R., Stopford, M., Crowe, T., & Gordon, S. (2014). *Shipping intelligence weekly*. UK: Technical report, Clarkson Research Services Limited.

Maragkogianni, A., Papaefthimiou, S. & Zopounidis, C. (2016). *Mitigating Shipping Emissions in European Ports: Social and Environmental Benefits*. Switzerland: Springer.

Marine Study (2015). *Definitions and Ship's Dimensions*. Downloaded 24.6.2016 from: http://marinestudy.net/coc-oral-exam-preparation-part-10-ship-construction/.

Matson, P.A., Dietz, T., Abdalati, W., Busalacchi, A.J., Caldeira, K., Corell, R.W., Defries, R.S., Fung, I.Y., Gaines, S., Hornberger, G.M. & Lemos, M.C. (2010). *Advancing the science of climate change*. Washington: The National Academy of Sciences.

Maritime & Coastguard Agency (2016). *Automatic Identification System (AIS)*. Downloaded 24.6.2016 from: https://mcanet.mcga.gov.uk/public/c4/solasv/annexes/annex17.htm

Molland A.F. , Turnock, S.R. & Hudson, D.A. (2011). *Ship Resistance and Propulsion: Practical Estimation of Propulsive Power*. Cambridge: Cambridge University Press.

Navcen (2016). *How AIS works*. Downloaded 1.6.2016 from: http://www.navcen.uscg.gov/?pageName=AISworks

Navcen (2016a). *Types of AIS (Per ITU-R M.1371 and TEC Standards)*. Downloaded 2.6.2016 from: http://www.navcen.uscg.gov/?pageName=typesAIS

Pachauri, R.K., Allen, M.R., Barros, V.R., Broome, J., Cramer, W., Christ, R., Church, J.A., Clarke, L., Dahe, Q., Dasgupta, P. and Dubash, N.K. (2014). *Climate change 2014: synthesis Report*. Contribution of working groups I, II and III to the fifth assessment report of the intergovernmental panel on climate change. IPCC.

Sharma, H (2015). *Naval Architecture Terminology and Coefficients of Forms*. Downloaded 24.6.2016 from: http://marinersgalaxy.com/2015/04/ship-dimensions-terminology-and.html

Smestad, B.B. (2015). *A Study of Satellite AIS Data and the Global Ship Traffic Through the Singapore Strait (Masteroppgave, NTNU)*. Trondheim: NTNU.

Smith, T.W.P., Jalkanen, J.P., Anderson, B.A., Corbett, J.J., Faber, J., Hanayama, S., ... & Raucci, C. (2014). *Third IMO GHG study 2014*. London: International Maritime Organization (IMO).

Steen, S. (Unknown). *Making speed-power predictions from model tests (Lecture Notes NTNU)*. Downloaded 20.6.2016 from: http://www.ivt.ntnu.no/imt/courses/tmr7/lecture/Speed-power_pred.pdf

Steen, S. (2011). Marin 3- kompendie, TMR427. Trondheim: NTNU.

Takahashi, H., Goto, A. & Abe, M. (2006). *Study on standards for main dimensions of the Design Ship*. Downloaded 24.6.2016 from: http://www.nilim.go.jp/lab/bcg/siryou/tnn/tnn0309pdf/ks0309011.pdf.

Whall, C., Cooper, D., Archer, K., Twigger, L., Thurston, N., Ockwell, D., McIntyre, A. & Ritchie, A. (2002). *Quantification of emissions from ships associated with ship movements between ports in the European Community (Report for the European Commission). Great Britain: Entec UK Limited.*

A | AIS Data Content

Information item	Information generation, type and quality of information
Static	
Maritime Mobile Service Identity(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
Dynamic	
Ship's position with accuracy indication and integrity status	Automatically updated from the position sensor connected to AIS. The accuracy indication is for better or worse than 10 m.
Position time stamp in UTC	Automatically updated from the 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	Navigational status information has to be manually entered by the OOW1 and changed as necessary, for example: • underway by engines • at anchor • not under command (NUC) • restricted in ability to manoeuvre (RI ATM) • moored • constrained by draught • aground • engaged in fishing • underway by sail - In practice, since all these relate to the COLREGs2, any change that is needed could be undertaken at the same time that the lights or shapes were changed
Rate of turn (ROT)	Automatically updated from the ship's ROT sensor or derived from the gyro. This information might not be available
Voyage-related	

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 confirm whether or not hazardous cargo is beeing carried, namely: • DG (Dangerous goods) • HS (Harmful substances) • MP (Marine pollutants) 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
Safety-related	
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

Table A.1: Data sent by ship (IMO, 2002).

B | AISdecode.py

```
/usr/bin/python

port aisparser
port sqlite3 as lite
port sys
port os


f extractMessages(filepath):
  global messageType1
  global messageType2
  global messageType3
  global messageType4
  global messageType5
  global timeStamps1
  global timeStamps2
  global timeStamps3
  global timeStamps4
  global timeStamps5
  messageType1 = []
  messageType2 = []
  messageType3 = []
  messageType4 = []
  messageType5 = []
  timeStamps1 = []
  timeStamps2 = []
  timeStamps3 = []
  timeStamps4 = []
  timeStamps5 = []
  s = []
  i = 0
  f = open(filepath, ’r’)
  for line in f:
      s.append(’c:’+line.split(’c:’)[1].split(’*’)[0]+’!BSVDM’+line.split(’!BSVDM’)[1])
  ais_state = aisparser.ais_state()
  for p in s:
      #print p
      result = aisparser.assemble_vdm( ais_state, p )
      if( result == 0):
          timestamp = p.split(’c:’)[1].split(’*’)[0].split(’!’)[0]
          ais_state.msgid = aisparser.get_6bit( ais_state.six_state, 6 )
          i = i+1
          if ais_state.msgid == 1:
              msg = aisparser.aismsg_1()
              aisparser.parse_ais_1( ais_state, msg )
              timeStamps1.append(timestamp)
              messageType1.append(msg)
          elif ais_state.msgid == 2:
              msg = aisparser.aismsg_2()
              aisparser.parse_ais_2( ais_state, msg )
              timeStamps2.append(timestamp)
              messageType2.append(msg)
          elif ais_state.msgid == 3:
              msg = aisparser.aismsg_3()
              aisparser.parse_ais_3( ais_state, msg )
              timeStamps3.append(timestamp)
              messageType3.append(msg)
          elif ais_state.msgid == 4:
              msg = aisparser.aismsg_4()
              aisparser.parse_ais_4( ais_state, msg )
              (status,lat_dd,long_ddd) = aisparser.pos2ddd(msg.latitude, msg.longitude)
```

```
            timeStamps4.append(timestamp)
            messageType4.append(msg)
        elif ais_state.msgid == 5:
            msg = aisparser.aismsg_5()
            aisparser.parse_ais_5( ais_state, msg )
            timeStamps5.append(timestamp)
            messageType5.append(msg)


f createDatabase(databasepath):

  con = lite.connect(databasepath)

  with con:

      cur = con.cursor()
      cur.execute("CREATE TABLE MessageType1(unixtime int, cog INT, latitude INT, longitude INT, msgid INT, nav_status INT, pos_acc INT, raim INT, regional INT, repeat INT, rot INT, slot_timeout INT, sog INT, spare INT, sub_message INT, sync_state INT, true INT, userid INT, utc_sec INT)")
      cur.execute("CREATE TABLE MessageType2(unixtime int, cog INT, latitude INT, longitude INT, msgid INT, nav_status INT, pos_acc INT, raim INT, regional INT, repeat INT, rot INT, slot_timeout INT, sog INT, spare INT, sub_message INT, sync_state INT, true INT, userid INT, utc_sec INT)")
      cur.execute("CREATE TABLE MessageType3(unixtime int, cog int,keep INT,latitude INT,longitude INT,msgid INT,nav_status INT,num_slots INT,pos_acc INT,raim INT,regional INT,repeat INT,rot INT,slot_increment int,sog int,spare INT,sync_state INT, true int,userid INT, utc_sec INT)")
      cur.execute("CREATE TABLE MessageType4(unixtime int, latitude INT,longitude INT,msgid INT,pos_acc INT,pos_type INT,raim INT,repeat INT,slot_timeout INT,spare int,sub_message int,sync_state INT,userid INT,utc_day INT,utc_hour INT,utc_minute INT,utc_month INT,utc_second INT,utc_year int)")
      cur.execute("CREATE TABLE MessageType5(unixtime int, callsign string,dest string,dim_bow int,dim_port INT,dim_starboard INT,dim_stern int,draught INT,dte INT,eta INT,imo INT,msgid INT,name text,pos_type INT,repeat INT,ship_type INT,spare INT ,userid INT,version INT)")
      cur.execute("CREATE INDEX userid_index ON MessageType1 (userid)")
      cur.execute("CREATE INDEX userid_index2 ON MessageType2 (userid)")
      cur.execute("CREATE INDEX userid_index3 ON MessageType3 (userid)")
      cur.execute("CREATE INDEX userid_index4 ON MessageType4 (userid)")
      cur.execute("CREATE INDEX userid_index5   ON MessageType5 (userid)")
      cur.execute("CREATE INDEX unixtime_index ON MessageType1 (unixtime)")
      cur.execute("CREATE INDEX unixtime_index2 ON MessageType2 (unixtime)")
      cur.execute("CREATE INDEX unixtime_index3 ON MessageType3 (unixtime)")
      cur.execute("CREATE INDEX unixtime_index4 ON MessageType4 (unixtime)")
      cur.execute("CREATE INDEX unixtime_index5   ON MessageType5 (unixtime)")

f writeToDatabase(databasepath):
  con = lite.connect(databasepath)
  con.isolation_level = None

  with con:
      cur = con.cursor()
      cur.execute(’BEGIN TRANSACTION’)
      for i in range(0, len(messageType1)):
          (status,lat_dd,long_ddd) = aisparser.pos2ddd(messageType1[i].latitude, messageType1[i].longitude)
          cur.execute("INSERT OR IGNORE INTO MessageType1(unixtime, cog , latitude , longitude , msgid , nav_status , pos_acc , raim , regional , repeat , rot , slot_timeout , sog, spare, sub_message, sync_state, true, userid, utc_sec) VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)",
              (timeStamps1[i], messageType1[i].cog, lat_dd, long_ddd, ord(messageType1[i].msgid),ord(messageType1[i].nav_status),
                  ord(messageType1[i].pos_acc), ord(messageType1[i].raim), ord(messageType1[i].regional), ord(messageType1[i].repeat), messageType1[i].rot,
                  ord(messageType1[i].slot_timeout),messageType1[i].sog, ord(messageType1[i].spare), messageType1[i].sub_message, ord(messageType1[i].sync_state),
                  messageType1[i].true, messageType1[i].userid, ord(messageType1[i].utc_sec)))
      for i in range(0, len(messageType2)):
          (status,lat_dd,long_ddd) = aisparser.pos2ddd(messageType2[i].latitude, messageType2[i].longitude)
          cur.execute("INSERT OR IGNORE INTO MessageType2(unixtime, cog , latitude , longitude , msgid , nav_status , pos_acc , raim , regional , repeat , rot , slot_timeout , sog, spare, sub_message, sync_state, true, userid, utc_sec) VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)",
              (timeStamps2[i], messageType2[i].cog, lat_dd, long_ddd, ord(messageType2[i].msgid),ord(messageType2[i].nav_status),
                  ord(messageType2[i].pos_acc), ord(messageType2[i].raim), ord(messageType2[i].regional), ord(messageType2[i].repeat), messageType2[i].rot,
```

```
                ord(messageType2[i].slot_timeout),messageType2[i].sog, ord(messageType2[i].spare), messageType2[i].sub_message, ord(messageType2[i].sync_state),
                messageType2[i].true, messageType2[i].userid, ord(messageType2[i].utc_sec)))
    for i in range(0, len(messageType3)):
        (status,lat_dd,long_ddd) = aisparser.pos2ddd(messageType3[i].latitude, messageType3[i].longitude)
        cur.execute("INSERT OR IGNORE INTO MessageType3(unixtime, cog,keep,latitude,longitude,msgid,nav_status,num_slots,pos_acc,raim,regional,repeat,rot,slot_increment,sog,spare,sync_state, true,userid, utc_sec) VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)", (timeStamps3[i], messageType3[i].cog, ord(messageType3[i].keep) ,
            lat_dd, long_ddd, ord(messageType3[i].msgid) , ord(messageType3[i].nav_status) , ord(messageType3[i].num_slots) , ord(messageType3[i].pos_acc) ,
            ord(messageType3[i].raim) , ord(messageType3[i].regional) , ord(messageType3[i].repeat) , messageType3[i].rot, messageType3[i].slot_increment, messageType3[i].sog,
            ord(messageType3[i].spare) , ord(messageType3[i].sync_state) ,messageType3[i].true, messageType3[i].userid, ord(messageType3[i].utc_sec)))
    for i in range(0, len(messageType4)):
        (status,lat_dd,long_ddd) = aisparser.pos2ddd(messageType4[i].latitude, messageType4[i].longitude)
        cur.execute("INSERT OR IGNORE INTO MessageType4(unixtime, latitude,longitude,msgid,pos_acc,pos_type,raim,repeat,slot_timeout,spare,sub_message,sync_state,userid,utc_day,utc_hour,utc_minute,utc_month,utc_second,utc_year) VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)", (timeStamps4[i], lat_dd, long_ddd, ord(messageType4[i].msgid) , ord(messageType4[i].pos_acc) ,
            ord(messageType4[i].pos_type) , ord(messageType4[i].raim) , ord(messageType4[i].repeat) , ord(messageType4[i].slot_timeout) , messageType4[i].spare, messageType4[i].sub_message,
            ord(messageType4[i].sync_state) ,  messageType4[i].userid, ord(messageType4[i].utc_day) , ord(messageType4[i].utc_hour) , ord(messageType4[i].utc_minute) , ord(messageType4[i].utc_month) ,
            ord(messageType4[i].utc_second) , messageType4[i].utc_year))
    for i in range(0, len(messageType5)):
        messageType5[i].dest = messageType5[i].dest.replace(" ","")
        messageType5[i].dest = messageType5[i].dest.replace("@","")
        messageType5[i].callsign = messageType5[i].callsign.replace(" ","")
        messageType5[i].callsign = messageType5[i].callsign.replace("@","")
        #messageType5[i].name = messageType5[i].name.replace(" ","")
        messageType5[i].name = messageType5[i].name.replace("@","")
        cur.execute("INSERT OR IGNORE INTO MessageType5(unixtime, callsign, dest, dim_bow, dim_port, dim_starboard, dim_stern, draught, dte, eta, imo, msgid, name, pos_type, repeat, ship_type, spare , userid, version) VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)", (timeStamps5[i], messageType5[i].callsign , messageType5[i].dest, messageType5[i].dim_bow, ord(messageType5[i].dim_port) ,
            ord(messageType5[i].dim_starboard) , messageType5[i].dim_stern, messageType5[i].draught, ord(messageType5[i].dte) , messageType5[i].eta, messageType5[i].imo, ord(messageType5[i].msgid) , messageType5[i].name ,
            ord(messageType5[i].pos_type) , ord(messageType5[i].repeat) , messageType5[i].ship_type, ord(messageType5[i].spare) , messageType5[i].userid, ord(messageType5[i].version)))
    cur.execute(’COMMIT’)
__name__ == "__main__":
 #databasepath =’’
 createDatabase("masterdatabase1")
 for foldername in os.listdir(’../AIS-Data/Kystverket/’):
    print foldername
    for filename in os.listdir(’../AIS-Data/Kystverket/’+foldername):
        #print filename
        if filename.split(’.’)[1]==’dat’:
            extractMessages(’../AIS-Data/Kystverket/’+foldername+’/’+filename)
            print len(messageType5)
            writeToDatabase("masterdatabase1")
            #print filename
        else:
            print ’Dropped file: ’+foldername+’/’+filename
```

C | Ship.py

```python
port getFlagState
port datetime


ass Ship(object):
  def __init__(self, mmsi):
      self._mmsi = mmsi
      self._name = None
      self._imo = None
      self._status = None
      self._status_description = None
      self._numSpeedRecords = None
      self._length = None
      self._lpp = None
      self._breadth = None
      self._shiptype = None
      self._displacement = None
      self._callsign = None
      self._draught = None
      self._msgID = None
      self._flagstate = None
      self._pitch = 0.7
      self._maxdraught = None
      self._mindraught = None
      self._maxspeed = None
      self._avgspeed = None
      self._shiptypename = None
      self._shiptypegroup = None
      self._cb = None
      self._sfc = None
      self._pe = None
      self._totalconsumption = None
      self._longcofb = None
      self._transbulbarea = None
      self._cobkeel = None
      self._cm = None
      self._cwp = None
      self._transom = None
      self._sapp = None
      self._sternparameter = None
      self._propdiameter = None
      self._numberOfPropellerBlades = None
      self._propellerClearance = None
      self._co2factor = 3.114
      self._ch4factor = 0.00006
      self._n2Ofactor = 0.00015
      self._noxfactor = 0.0903
      self._nmvocfactor = 0.00308
      self._cofactor = 0.00277
      self._pmfactor = 0.00728
      self._so2factor = 0.025
      self._noeff = None
      self._nseff = 0.98
      self._nreff = 1.0
      self._nheff = None
      self._pb = None
      self._totalSailingDays = None

  @property
  def mmsi(self):
      return self._mmsi
```

```

    @mmsi.setter
    def mmsi(self, mmsi):
        self._mmsi = mmsi

    @property
    def name(self):
        return self._name

    @name.setter
    def name(self, name):
        self._name = name

    @property
    def imo(self):
        return self._imo

    @imo.setter
    def imo(self, imo):
        self._imo = imo

    @property
    def status(self):
        return self._status

    @status.setter
    def status(self, status):
        self._status = status

    @property
    def statusDescription(self):
        return self._status_description

    @statusDescription.setter
    def statusDescription(self, statusDescription):
        self._status_description = statusDescription

    @property
    def numSpeedRecords(self):
        return self._numSpeedRecords

    @numSpeedRecords.setter
    def numSpeedRecords(self, numSpeedRecords):
        self._numSpeedRecords = numSpeedRecords

    @property
    def shiptype(self):
        return self._shiptype

    @shiptype.setter
    def shiptype(self, type):
        self._shiptype = type

    # @property
    @property
    def cb(self):
        return self._cb

    @cb.setter
    def cb(self, cb):
        self._cb = cb

    @property
    def deltadraught(self):
        maxdraught = self.maxdraught
```

```
        mindraught = self.mindraught

        if maxdraught == None or mindraught == None:
            return None
        deltadraught = maxdraught - mindraught
        return deltadraught

    @property
    def flagstate(self):
        return self._flagstate

    @flagstate.setter
    def flagstate(self, flagstate):
        self._flagstate = flagstate

    @property
    def callsign(self):
        return self._callsign

    @callsign.setter
    def callsign(self, callsign):
        self._callsign = callsign

    @property
    def msgid(self):
        return self._msgID

    @msgid.setter
    def msgid(self, msgid):
        self._msgID = msgid

    @property
    def maxspeed(self):
        return self._maxspeed

    @maxspeed.setter
    def maxspeed(self, maxspeed):
        self._maxspeed = maxspeed

    @property
    def avgspeed(self):
        return self._avgspeed

    @avgspeed.setter
    def avgspeed(self, avgspeed):
        self._avgspeed = avgspeed

    # Length
    @property
    def length(self):
        return self._length

    @length.setter
    def length(self, length):
        self._length = length

    @property
    def lengthwl(self):
        length = self._length
        if length == None:
            return None
        lwl = length * 0.97
        return lwl

    # Length between perpendiculars
```

```python
@property
def lpp(self):
    if self.lengthwl == None:
        return None
    lpp = self.lengthwl * 0.97
    return lpp

@lpp.setter
def lpp(self, lpp):
    self._lpp = lpp

# Breadth
@property
def breadth(self):
    return self._breadth

@breadth.setter
def breadth(self, breadth):
    self._breadth = breadth

# Draught
@property
def draught(self):
    return self._draught

@draught.setter
def draught(self, draught):
    self._draught = draught

@property
def maxdraught(self):
    return self._maxdraught

@maxdraught.setter
# draughtforep=10.0 #AIS
# draughtaftp=10.0  #AIS
def maxdraught(self, maxdraught):
    self._maxdraught = maxdraught

@property
def mindraught(self):
    return self._mindraught

@mindraught.setter
# draughtforep=10.0 #AIS
# draughtaftp=10.0  #AIS
def mindraught(self, mindraught):
    self._mindraught = mindraught

# Discplacement volume moulded
@property
def displacement(self):
    density = 1.025
    if self.cb == None or self.lpp == None or self.breadth == None or self.draught == None:
        return None
    displacement = self.cb * self.lpp * self.breadth * self.draught * density
    return displacement

@displacement.setter
def displacement(self, displacement):
    self._displacement = displacement

# Longitudinal center of buoyancy
@property
def longcofb(self):
```

```
        if self.lengthwl == None:
            return None
        longcofb = -0.75 * (self.lengthwl / 2.0) / 100.0
        return longcofb

    @longcofb.setter
    def longcofb(self, longcofb):
        self._longcofb = longcofb

    @property
    def noeff(self):
        return self._noeff

    @noeff.setter
    def noeff(self, no):
        self._noeff = no

    @property
    def nreff(self):
        return self._nreff

    @nreff.setter
    def nreff(self, nr):
        self._nreff = nr

    @property
    def nseff(self):
        return self._nseff

    @nseff.setter
    def nseff(self, ns):
        self._nseff = ns

    @property
    def nheff(self):
        return self._nheff

    @nheff.setter
    def nheff(self, nh):
        self._nheff = nh

    # Midship section Coefficient
    @property
    def cm(self):
        # Ax/(BWL*Draught) BWL = beam waterline
        if self.cb == None:
            return None
        cm = 0.977 + 0.085 * (self._cb - 0.60)
        return cm

    @cm.setter
    def cm(self, cm):
        self._cm = cm

    # Center of bulb area above the keel line
    @property
    def cobkeel(self):
        if self.draught == None:
            return None
        cobkeel = self.draught * 0.4
        return cobkeel

    @cobkeel.setter
    def cobkeel(self, cobkeel):
        self._cobkeel = cobkeel
```

```

# Waterplane area coefficient
@property
def cwp(self):
    # Midship section area/WLBeam *Body draught
    if self._cb == None:
        return None

    cwp = 0.55 + 0.45 * self._cb  # Results of statistical analysis of IHS Fairplay data

    return cwp
    # return 0.75

@cwp.setter
def cwp(self, cwp):
    self._cwp = cwp

# Transveres Bulb area
@property
def transbulbarea(self):
    # file: // / C: / Users / DagIvar / Downloads / CHARCHALIS % 20(1).pdf
    if self.cm == None or self.breadth == None or self.draught == None:
        return None
    transbulbarea = 0.08 * self.cm * self.breadth * self.draught
    return transbulbarea

@transbulbarea.setter
def transbulbarea(self, transbulbarea):
    self._transbulbarea = transbulbarea

# Transom Area
@property
def transom(self):
    if self.cm == None or self.breadth == None or self.draught == None:
        return None
    transom = 0.051 * self.cm * self.breadth * self.draught
    if transom == None:
        return None
    return transom

@transom.setter
def transom(self, transom):
    self._transom = transom

# Wetted appendages Areas, Sapp
@property
def sapp(self):
    # ?
    return 50.0

@sapp.setter
def sapp(self, sapp):
    self._sapp = sapp

# stern shape parameter
@property
def sternparameter(self):
    return 10.0

@sternparameter.setter
def sternparameter(self, sternparameter):
    self._sternparameter = sternparameter

@property
def shiptypename(self):
```

```
    return self._shiptypename

@shiptypename.setter
def shiptypename(self, name):
    self._shiptypename = name

@property
def shiptypegroup(self):
    return self._shiptypegroup

@shiptypegroup.setter
def shiptypegroup(self, name):
    self._shiptypegroup = name

# Propeller diameter
@property
def propdiameter(self):
    # Propeller diameter from ship type and size d/D
    if self.draught == None:
        return None
    if self._shiptypename == "Conteiner ship":
        return 0.74 * self.draught
    elif self._shiptypename == "Bulk carrier":
        return 0.65 * self.draught
    elif self._shiptypename == "Oil tanker":
        return 0.65 * self.draught
    return self.draught * 0.7

@propdiameter.setter
def propdiameter(self, propdiameter):
    self._propdiameter = propdiameter

# Number of Propeller blades
@property
def numberOfPropellerBlades(self):
    # Number of propeller from ship type and size
    return 4.0

@numberOfPropellerBlades.setter
def numberOfPropellerBlades(self, numberofblades):
    self._numberOfPropellerBlades = numberofblades

# Clearance propeller with keel line
@property
def propellerClearance(self):
    # Clearance propeller from keel line from ship type and size
    if self.numberOfPropellerBlades == None or self.propdiameter == None:
        return None
    clear = (0.48 - 0.02 * self.numberOfPropellerBlades) * self.propdiameter / 2
    return clear

@propellerClearance.setter
def propellerClearance(self, clearance):
    self._propellerClearance = clearance

# Prismatic Coefficient
@property
def cp(self):
    if self.displacement == None or self.cm == None or \
                    self.breadth == None or self.draught == None or self.length == None:
        return None
    if self.cm == 0 or self.breadth == 0 or self.draught == 0 or self.length == 0:
        return None
    cp = self.displacement / (self.cm * self.breadth * self.draught * self.length)
    return cp
```

```

    @property
    def sfc(self):
        return self._sfc

    @sfc.setter
    def sfc(self, sfc):
        self._sfc = sfc

    # Pitch
    @property
    def pitch(self):
        return self.pitch

    @pitch.setter
    def pitch(self, pitch):
        self._pitch = pitch
        return

    # Coefficient Screw
    def calck(self):
        k = 0.2
        return k

    @property
    def pe(self):
        return self.pe

    @pe.setter
    def pe(self, pe):
        self._pe = pe

    @property
    def totalSailingDays(self):
        return self._totalSailingDays

    @totalSailingDays.setter
    def totalSailingDays(self, days):
        self._totalSailingDays = days

    @property
    def totalconsumption(self):
        if self._totalconsumption > 80000.0:
            return None

        return self._totalconsumption

    @totalconsumption.setter
    def totalconsumption(self, totalconsumption):
        self._totalconsumption = totalconsumption

    # Emission
    @property
    def co2emission(self):
        if self.totalconsumption == None:
            return None
        return self.totalconsumption * self._co2factor

    @property
    def ch4emission(self):
        if self.totalconsumption == None:
            return None
        return self._totalconsumption * self._ch4factor

    @property
```

```
def n20emission(self):
    if self.totalconsumption == None:
        return None
    return self._totalconsumption * self._n20factor

@property
def noxemission(self):
    if self.totalconsumption == None:
        return None
    return self._totalconsumption * self._noxfactor

@property
def nmvocemission(self):
    if self.totalconsumption == None:
        return None
    return self._totalconsumption * self._nmvocfactor

@property
def coemission(self):
    if self.totalconsumption == None:
        return None
    return self._totalconsumption * self._cofactor

@property
def pmemission(self):
    if self.totalconsumption == None:
        return None
    return self._totalconsumption * self._pmfactor

@property
def so2emission(self):
    if self.totalconsumption == None:
        return None
    return self._totalconsumption * self._so2factor

def __str__(self):
    shipstring = ''
    shipstring += 'Ships MMSI number: ' + str(self._mmsi) + '\n'
    shipstring += 'Ships IMO number: ' + str(self._imo) + '\n'
    shipstring += 'Ship name: ' + str(self._name) + '\n'
    shipstring += 'Ships call sign: ' + str(self._callsign) + '\n'
    shipstring += 'This ship is of Ship Type (defined by AIS message): ' + str(self._shiptype) + '\n'
    shipstring += 'This ship is from : ' + str(self._flagstate) + '\n'
    #  shipstring+='This message was sent: ',self.Unixtime()
    #  shipstring+='Destiantion of ship for this message: ',self.Dest()
    shipstring += 'Length of ship LOA is:' + str(self._length) + 'm \n'
    shipstring += 'Length of ship LWL is:' + str(self.lengthwl) + 'm \n'
    shipstring += 'Length between perpendiculars of ship Lpp is:' + str(self.lpp) + 'm \n'
    shipstring += 'Breadth of ship, B : ' + str(self.breadth) + 'm \n'
    shipstring += 'Draught of ship, T: ' + str(self._draught) + 'm \n'
    shipstring += 'Displacement of ship is:' + str(self.displacement) + 'm3 \n'
    shipstring += 'Longitudinal C of B from AP: ' + str(self.longcofb) + 'm \n'
    shipstring += 'Transveres bulb area: ' + str(self.transbulbarea) + 'm \n'
    shipstring += 'Center of bulb area above keel line: ' + str(self.cobkeel) + 'm \n'
    shipstring += 'Midship section Coefficient: ' + str(self.cm) + ' - \n'
    shipstring += 'Waterplane area coefficient: ' + str(self.cwp) + ' - \n'
    shipstring += 'Transom Area: ' + str(self.transom) + 'm2 \n'
    shipstring += 'Wetted area appendages: ' + str(self.sapp) + 'm2 \n'
    shipstring += 'Stern shape parameter: ' + str(self.sternparameter) + ' - \n'
    shipstring += 'Propeller diameter: ' + str(self.propdiameter) + 'm \n'
    shipstring += 'Number of propeller blades: ' + str(self.numberOfPropellerBlades) + ' - \n'
    shipstring += 'Clearance propeller with keel line: ' + str(self.propellerClearance) + 'm \n'
    shipstring += 'CP : ' + str(self.cp) + ' - \n'
    shipstring += 'CB : ' + str(self.cb) + ' - \n'
    shipstring += 'Delta draught : ' + str(self.deltadraught) + ' m \n'
```

```
        shipstring += ’Max draught : ’ + str(self.maxdraught) + ’ m \n’
        shipstring += ’Min draught : ’ + str(self.mindraught) + ’ m \n’
        shipstring += ’Shiptypename : ’ + str(self.shiptypename) + ’ \n’
        shipstring += ’Shiptypegroup : ’ + str(self.shiptypegroup) + ’ \n’
        shipstring += ’Total consumption for this ship : ’ + str(self.totalconsumption) + ’ tonn \n’
        shipstring += ’Flagstate : ’ + str(self.flagstate) + ’ \n’
        shipstring += ’Max speed : ’ + str(self.maxspeed) + ’ \n’
        shipstring += ’Average speed messagetype 1 : ’ + str(self.avgspeed) + ’ \n’
        # shipstring+=’CP1 : ’+str(self.cp1) +’ - \n’


        return shipstring

    def getCSV(self):
        shipstring = ’’
        shipstring += str(self.mmsi) + ’,’
        shipstring += str(self.imo) + ’,’
        shipstring += str(self.name) + ’,’
        shipstring += str(self.status) + ’,’
        shipstring += str(self.statusDescription) + ’,’
        shipstring += str(self.callsign) + ’,’
        shipstring += str(self.shiptype) + ’,’
        shipstring += str(self.shiptypename) + ’,’
        shipstring += str(self.shiptypegroup) + ’,’
        shipstring += str(self.totalSailingDays) + ’,’
        shipstring += str(self.totalconsumption) + ’,’
        shipstring += str(self.co2emission) + ’,’
        shipstring += str(self.ch4emission) + ’,’
        shipstring += str(self.n2Oemission) + ’,’
        shipstring += str(self.noxemission) + ’,’
        shipstring += str(self.nmvocemission) + ’,’
        shipstring += str(self.coemission) + ’,’
        shipstring += str(self.pmemission) + ’,’
        shipstring += str(self.so2emission) + ’,’

        shipstring += str(self.flagstate) + ’,’
        shipstring += str(self.maxspeed) + ’,’
        shipstring += str(self.avgspeed) + ’,’
        shipstring += str(self.numSpeedRecords) + ’,’
        shipstring += str(self.length) + ’,’
        shipstring += str(self.lengthwl) + ’,’
        shipstring += str(self.lpp) + ’,’
        shipstring += str(self.breadth) + ’,’
        shipstring += str(self.draught) + ’,’
        shipstring += str(self.deltadraught) + ’,’
        shipstring += str(self.maxdraught) + ’,’
        shipstring += str(self.mindraught) + ’,’
        shipstring += str(self.displacement) + ’,’
        shipstring += str(self.longcofb) + ’,’
        shipstring += str(self.transbulbarea) + ’,’
        shipstring += str(self.cobkeel) + ’,’
        shipstring += str(self.cm) + ’,’
        shipstring += str(self.cwp) + ’,’
        shipstring += str(self.transom) + ’,’
        shipstring += str(self.sapp) + ’,’
        shipstring += str(self.sternparameter) + ’,’
        shipstring += str(self.propdiameter) + ’,’
        shipstring += str(self.numberOfPropellerBlades) + ’,’
        shipstring += str(self.propellerClearance) + ’,’
        shipstring += str(self.cp) + ’,’
        shipstring += str(self.cb) + ’,’

        return shipstring

    def getCsvHeader(self):
```

```
        shipstring = ''
        shipstring += 'mmsi' + ','
        shipstring += 'imo' + ','
        shipstring += 'name' + ','
        shipstring += 'status' + ','
        shipstring += 'statusDescription' + ','
        shipstring += 'callsign' + ','
        shipstring += 'shiptype' + ','
        shipstring += 'shiptypename' + ','
        shipstring += 'shiptypegroup' + ','
        shipstring += 'total sailing days' + ','
        shipstring += 'totalconsumption' + ','
        shipstring += 'co2emission' + ','
        shipstring += 'ch4emission' + ','
        shipstring += 'n2Oemission' + ','
        shipstring += 'noxemission' + ','
        shipstring += 'nmvocemission' + ','
        shipstring += 'coemission' + ','
        shipstring += 'pmemission' + ','
        shipstring += 'so2emission' + ','
        shipstring += 'flagstate' + ','
        shipstring += 'maxspeed' + ','
        shipstring += 'avgspeed' + ','
        shipstring += 'numSpeedRecords' + ','
        shipstring += 'length' + ','
        shipstring += 'lengthwl' + ','
        shipstring += 'lpp' + ','
        shipstring += 'breadth' + ','
        shipstring += 'draught' + ','
        shipstring += 'deltadraught' + ','
        shipstring += 'maxdraught' + ','
        shipstring += 'mindraught' + ','
        shipstring += 'displacement' + ','
        shipstring += 'longcofb' + ','
        shipstring += 'transbulbarea' + ','
        shipstring += 'cobkeel' + ','
        shipstring += 'cm' + ','
        shipstring += 'cwp' + ','
        shipstring += 'transom' + ','
        shipstring += 'sapp' + ','
        shipstring += 'sternparameter' + ','
        shipstring += 'propdiameter' + ','
        shipstring += 'numberOfPropellerBlades' + ','
        shipstring += 'propellerClearance' + ','
        shipstring += 'cp' + ','
        shipstring += 'cb' + ','

        return shipstring
```

D | Holtrop.py

```
Holtrop.py
master thesis
Author: Stian Rakke
Spring 2016
port math
port sqlite3 as lite
port sys
om Ship import Ship
port numpy as np


ass Holtrop(object):
  def __init__(self, ship, speeds):
      self._ship = ship

      self._speeds = speeds
      self.shipspeed = None  # AIS
      self.gravity = 9.802  # Greenwich
      self.rho = 1025.0
      self.kinviscosity = 0.00000118831

  def getBrakePowers(self):
      brakepower = {}
      for speed in self._speeds:
          self.shipspeed = speed[0] / 10.0
          brakepower[speed[0] / 10.0] = self.brakePower()

      return brakepower

  #########  Functions Holtrop #########

  def calcc1(self):
      breadth = self._ship.breadth
      draughtforep = self._ship.draught
      ie = self.calcie()
      c1 = 2223105.0 * self.calcc7() ** (3.78613) * (draughtforep / breadth) ** (1.07961) * (90.0 - ie) ** (-1.37566)
      return (c1)

  def calcc2(self):
      c2 = math.exp(-1.89 * math.sqrt(self.calcc3()))
      return (c2)

  def calcc3(self):
      transbulb = self._ship.transbulbarea
      breadth = self._ship.breadth
      draughtforep = self._ship.draught
      centerofbulb = self._ship.cobkeel

      c3 = 0.56 * transbulb ** 1.5 / \
           (breadth * draughtforep * (0.31 * math.sqrt(transbulb) + draughtforep - centerofbulb))
      return (c3)

  def calcc4(self):
      lengthwl = self._ship.lengthwl
      draughtforep = self._ship.draught
      if draughtforep / lengthwl <= 0.04:
          c4 = draughtforep / lengthwl
      else:
          c4 = 0.04
      return c4

```

```python
    def calcc5(self):
        transom = self._ship.transom
        breadth = self._ship.breadth
        midshipC = self._ship.cm
        draughtforep = self._ship.draught
        c5 = 1.0 - 0.8 * transom / (breadth * draughtforep * midshipC)
        return c5

    def calcc6(self):
        if self.calcfnt() < 5.0:
            c6 = 0.2 * (1.0 - 0.2 * self.calcfnt())
        else:
            c6 = 0.0
        return c6

    def calcc7(self):
        breadth = self._ship.breadth
        lengthwl = self._ship.lengthwl

        if (breadth / lengthwl) < 0.11:
            c7 = 0.229577 * (breadth / lengthwl) ** 0.33333
        elif (breadth / lengthwl) > 0.11 and (breadth / lengthwl) < 0.25:
            c7 = (breadth / lengthwl)
        else:
            c7 = 0.5 - 0.625 * (breadth / lengthwl)
        return c7

    def calcc8(self):
        breadth = self._ship.breadth
        lengthwl = self._ship.lengthwl
        draughtaftp = self._ship.draught
        propdiameter = self._ship.propdiameter
        if breadth / draughtaftp < 5.0:
            c8 = breadth * self.calcs() / (lengthwl * propdiameter * draughtaftp)
        else:
            c8 = self.calcs() * (7.0 * breadth / draughtaftp - 25.0) / (
            lengthwl * propdiameter * (breadth / draughtaftp - 3.0))
        return c8

    def calcc9(self):
        if self.calcc8() < 28.0:
            c9 = self.calcc8()
        else:
            c9 = 32.0 - 16.0 / (self.calcc8() - 24.0)
        return c9

    def calcc10(self):
        breadth = self._ship.breadth
        lengthwl = self._ship.lengthwl
        if lengthwl / breadth > 5.0:
            c10 = breadth / lengthwl
        else:
            c10 = 0.25 - 0.003328402 / (breadth / lengthwl - 0.134615385)
        return c10

    def calcc11(self):
        breadth = self._ship.breadth
        draughtaftp = self._ship.draught
        propdiameter = self._ship.propdiameter
        if draughtaftp / propdiameter < 2.0:
            c11 = draughtaftp / propdiameter
        else:
            c11 = 0.0833333 * (draughtaftp / propdiameter) ** 3.0 + 1.33333
        return c11

```

```
def calcc12(self):
    breadth = self._ship.breadth
    lengthwl = self._ship.lengthwl
    draughtforep = self._ship.draught
    propdiameter = self._ship.propdiameter
    if draughtforep / lengthwl > 0.05:
        c12 = (draughtforep / lengthwl) ** 0.2228446
    elif draughtforep / lengthwl > 0.02 and draughtforep / lengthwl < 0.05:
        c12 = 48.2 * (draughtforep / lengthwl - 0.02) ** 2.078 + 0.479948
    else:
        c12 = 0.479948
    return c12

def calcc13(self):
    c13 = 1.0 + 0.003 * self.calcsternc()
    return c13

def calcc15(self):
    disp = self._ship.displacement
    lengthwl = self._ship.lengthwl
    if lengthwl ** 3 / disp < 512:
        c15 = -1.69385
    elif lengthwl ** 3 / disp > 1727.0:
        c15 = 0
    else:
        c15 = -1.69385 + (lengthwl / disp ** (1.0 / 3.0) - 8.0 / 2.36)
    return c15

def calcc16(self):
    cp = self._ship.cp
    if cp < 0.8:
        c16 = 8.07981 * cp - 13.8673 * cp ** 2.0 + 6.984388 * cp ** 3
    else:
        c16 = 1.73014 - 0.7067 * cp
    return c16

def calcp0minpv(self):
    p0minpv = 99047.0
    return p0minpv

def calcrgh(self):
    rgh = self.rho * self.gravity * self._ship.cobkeel
    return rgh

def calcshipspeed(self):
    speed = self.shipspeed * 0.5144
    return speed

def calcs(self):
    lengthwl = self._ship.lengthwl
    draughtforep = self._ship.draught
    breadth = self._ship.breadth
    midshipC = self._ship.cm
    cb = self._ship.cb
    cwp = self._ship.cwp
    transbulb = self._ship.transbulbarea
    s = lengthwl * (2 * draughtforep + breadth) * math.sqrt(midshipC) * (
    0.453 + 0.4425 * cb - 0.2862 * midshipC - 0.003467 * (
    breadth / draughtforep) + 0.3696 * cwp) + 2.38 * transbulb / cb
    return s

def calcsapp(self):
    sapp = self._ship.sapp
    return sapp

```

```python
    def calcsternc(self):
        sternCo = self._ship.sternparameter
        return sternCo

    def calcdisp(self):
        disp = self._ship.displacement
        return disp

    def calccp(self):
        cp = self._ship.cp
        return cp

    def calccp1(self):
        cp1 = 1.45 * self.calccp() - 0.315 - 0.0225 * self.calclcb()
        return cp1

    def calcm1(self):
        lengthwl = self._ship.lengthwl
        breadth = self._ship.breadth
        draughtforep = self._ship.draught
        disp = self._ship.displacement
        m1 = 0.0140407 * lengthwl / draughtforep - 1.75254 * disp ** (
        1.0 / 3.0) / lengthwl - 4.79323 * breadth / lengthwl - self.calcc16()
        return m1

    def calcm2(self):
        cp = self._ship.cp
        m2 = self.calcc15() * cp ** 2.0 * math.exp(-0.1 * self.calcfn() ** -2.0)
        return m2

    def calcreyn(self):
        lengthwl = self._ship.lengthwl
        reyn = self.calcshipspeed() * lengthwl / self.kinviscosity
        return reyn

    def calclambda(self):
        lengthwl = self._ship.lengthwl
        breadth = self._ship.breadth
        cp = self._ship.cp
        if lengthwl / breadth < 12.0:
            lambda1 = 1.446 * cp - 0.03 * lengthwl / breadth
        else:
            lambda1 = 1.446 * cp - 0.36
        return lambda1

    def calcfn(self):
        lengthwl = self._ship.lengthwl
        fn = self.calcshipspeed() / math.sqrt(self.gravity * lengthwl)
        return fn

    def calcfni(self):
        draughtforep = self._ship.draught
        centerofbulb = self._ship.cobkeel
        transbulb = self._ship.transbulbarea

        fni = self.calcshipspeed() / (math.sqrt(self.gravity * (
        draughtforep - centerofbulb - 0.25 * math.sqrt(transbulb)) + 0.15 * self.calcshipspeed() ** 2))
        return fni

    def calcfnt(self):
        transomarea = self._ship.transom
        breadth = self._ship.breadth
        cwp = self._ship.cwp
        fnt = self.calcshipspeed() / math.sqrt(2 * self.gravity * transomarea / (breadth + breadth * cwp))
        return fnt
```

```

    def calcca(self):
        lengthwl = self._ship.lengthwl
        cb = self._ship.cb
        ca = 0.006 * (lengthwl + 100.0) ** -0.16 - 0.00205 + 0.003 * math.sqrt(
            lengthwl / 7.5) * cb ** 4 * self.calcc1() * (0.04 - self.calcc4())
        return ca

    def calccf(self):
        cf = 0.075 / (math.log10(self.calcreyn()) - 2.0) ** 2.0
        return cf

    def calcct(self):
        ct = 0
        return ct

    def calccb(self):
        cb = self._ship.cb
        return cb

    def calccv(self):
        cv = self.calcformfactor() * self.calccf() + self.calcca()
        return cv

    def calcie(self):
        breadth = self._ship.breadth
        lengthwl = self._ship.lengthwl
        cwp = self._ship.cwp
        cp = self._ship.cp
        lpp = self._ship.lpp
        lcb = -0.75 * (lpp / 2.0) / 100.0
        lr = lengthwl * (1.0 - cp + ((0.06 * cp * lcb) / (4.0 * cp - 1.0)))
        displacement = self._ship.displacement
        ie = 1.0 + 89.0 * math.exp(
            -(lengthwl / breadth) ** 0.80856 * (1.0 - cwp) ** 0.30484 * (1.0 - cp - 0.0225 * lcb) ** 0.6367 * (
            lr / breadth) ** 0.34574 * (100.0 * displacement / lengthwl ** 3.0) ** 0.16302)
        return ie

    def calclcb(self):
        lpp = self._ship.lpp
        lcb = -0.75 * (lpp / 2) / 100
        return lcb

    def calclengthofrun(self):
        lengthwl = self._ship.lengthwl
        cp = self._ship.cp
        cb = self._ship.cb
        lcb = self.calclcb()
        lr = lengthwl * (1.0 - cp + ((0.06 * cp * lcb) / (4.0 * cp - 1.0)))
        return lr

    def calcpitch(self):
        pitch = self._ship.pitch
        return pitch

    def calcd(self):
        # from Holtrop approximation sheet
        d = -0.9
        return d

    def calcformfactor(self):
        breadth = self._ship.breadth
        cp = self._ship.cp
        formfactor = self.calcc13() * (
        0.93 + self.calcc12() * ((breadth / self.calclengthofrun()) ** 0.92497) * ((0.95 - cp) ** -0.521448) * (
```

```
        (1.0 - cp + 0.0225 * self.calclcb()) ** 0.6906))
        return formfactor

    def calcformfactor2(self, sapprudderbehindskeg, sapprudderbehindstern, sapptwinscrewbalancerudder,
                        sappshaftbrackets, sappskeg, sappstrutbossings, sapphullbossings, sappshafts,
                        sappstabilazerfins, sappdome, sappbilgekeels):
        return 1.50

    def calct(self, a=’’):
        lengthwl = self._ship.lengthwl
        breadth = self._ship.breadth
        cb = self._ship.cb
        propdiameter = self._ship.propdiameter
        draughtforep = self._ship.draught
        cp1 = self.calccp1()
        sternprop = self._ship.sternparameter
        if a is ’twinscrew’:  # twinscrew
            t = 0.325 * cb - 0.1885 * propdiameter / math.sqrt(breadth * draughtforep)
        elif a is ’singlescrewopenstern’:  # singlescrew, open sternon slender fast sailing ships:
            t = 0.1
        elif a is ’singlescrewconventionalstern’:  # single screw, conventional stern
            t = 0.001979 * lengthwl / (
            breadth - breadth * cp1) + 1.0585 * self.calcc10() - 0.00524 - 0.1418 * propdiameter ** 2.0 / (
            breadth * draughtforep) + 0.0015 * sternprop
        else:
            t = 0.001979 * lengthwl / (
            breadth - breadth * cp1) + 1.0585 * self.calcc10() - 0.00524 - 0.1418 * propdiameter ** 2.0 / (
            breadth * draughtforep) + 0.0015 * sternprop
        return t

    def calcw(self, a=’’):
        cb = self._ship.cb
        propdiameter = self._ship.propdiameter
        breadth = self._ship.breadth
        draughtforep = self._ship.draught
        lengthwl = self._ship.lengthwl
        cp1 = self.calccp1()
        cp = self._ship.cp
        sternc = self._ship.sternparameter
        if a is ’twinscrew’:  # twinscrew :#twinscrew
            w = 0.3095 * cb + 10.0 * self.calccv() * cb - 0.23 * propdiameter / math.sqrt(breadth * draughtforep)
        elif a is ’singlescrewopenstern’:  # singlescrew, open sternon slender fast sailing ships:
            w = 0.3 * cb + 10.0 * self.calccv() * cb - 0.1
        elif a is ’singlescrewconventionalstern’:  # single screw, conventional stern
            w = self.calcc9() * self.calccv() * lengthwl / draughtforep * (
            0.0661875 + 1.21756 * self.calcc11() * self.calccv() / (1.0 - cp1)) + 0.24558 * math.sqrt(
                breadth / (lengthwl * (1.0 - cp1))) - 0.09726 / (0.95 - cp) + 0.11434 / (
            0.95 - cb) + 0.75 * sternc * self.calccv() + 0.002 * sternc
        else:
            w = self.calcc9() * self.calccv() * lengthwl / draughtforep * (
            0.0661875 + 1.21756 * self.calcc11() * self.calccv() / (1.0 - cp1)) + 0.24558 * math.sqrt(
                breadth / (lengthwl * (1.0 - cp1))) - 0.09726 / (0.95 - cp) + 0.11434 / (
            0.95 - cb) + 0.75 * sternc * self.calccv() + 0.002 * sternc
        return w

    def calclrdll(self):
        cp = self._ship.cp
        cb = self._ship.cb
        lrdll = 1.0 - cp + 0.06 * cp * cb / (4.0 * cp - 1.0)
        return lrdll

    def calcdeltacd(self):
        delcd = (2.0 + 4.0 * self.calctdlc075()) * (
        0.003605 - (1.89 + 1.62 * math.log10(self.calcc075() / self.calckp())) ** -2.5)
        return delcd
```

```python

    def calctdlc075(self):
        nopropblades = self._ship.numberOfPropellerBlades
        propdiameter = self._ship.propdiameter
        tdlc075 = (0.0185 - 0.00125 * nopropblades) * propdiameter / self.calcc075()
        return tdlc075

    def calcc075(self):
        propdiameter = self._ship.propdiameter
        nopropblades = self._ship.numberOfPropellerBlades
        c075 = 2.073 * (self.calcaedlao()) * propdiameter / nopropblades
        return c075

    def calcaedlao(self):
        nopropblades = self._ship.numberOfPropellerBlades
        propdiameter = self._ship.propdiameter
        k = self.calck()
        aedlao1 = k + (1.3 + 0.3 * nopropblades) * (self.calcthrust() * 1000) / (
        propdiameter ** 2.0 * (self.calcp0minpv() + self.calcrgh()))
        return aedlao1

    def calcz(self):
        z = self._ship.numberOfPropellerBlades
        return z

    def calck(self):
        # k=0.1
        k = self._ship.calck()
        return k

    def calckp(self):
        # standard
        kp = 0.00003
        return kp

    def calckts(self):
        #
        kts = 0
        return kts

    def calckqs(self):
        #
        kqs = 0
        return kqs

    def calcnr(self):
        # From B-series polynomials
        nr = self._ship.nreff
        return nr

    def calcn0(self):
        # From B-series polynomials
        n0 = self._ship.noeff
        return n0

    def calcns(self):
        #
        ns = self._ship.noeff
        return ns

    def calcnh(self):
        nh = (1.0 - self.calct()) / (1.0 - self.calcw())
        return nh

    def calcnd(self):
```

```
        nd = self.calcnh() * self.calcn0() * self.calcnr()
        return nd

    def calcrf(self):
        rf = 0.5 * self.rho * self.calcshipspeed() ** 2.0 * self.calcs() * self.calccf()
        rf = rf / 1000.0
        return rf

    def calcrapp(self):
        rapp = 0.5 * self.rho * self.calcshipspeed() ** 2.0 * self.calcsapp() * (
        self.calcformfactor2(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)) * self.calccf()
        rapp = rapp / 1000.0
        return rapp

    def calcrw(self):
        rw = self.calcc1() * self.calcc2() * self.calcc5() * self.calcdisp() * self.rho * self.gravity * math.exp(
            self.calcm1() * self.calcfn() ** (self.calcd()) + self.calcm2() * math.cos(
                self.calclambda() * self.calcfn() ** (-2.0)))
        rw = rw / 1000.0
        return rw

    def calcrb(self):
        transbulb = self._ship.transbulbarea
        rb = 0.11 * math.exp(-3.0 * self.calcpb() ** (-2.0)) * self.calcfni() ** 3.0 * transbulb ** (
        1.5) * self.rho * self.gravity / (1.0 + self.calcfni() ** 2.0)
        rb = rb / 1000.0
        return rb

    def calcrtr(self):
        transomearea = self._ship.transbulbarea
        rtr = 0.5 * self.rho * self.calcshipspeed() ** 2.0 * transomearea * self.calcc6()
        rtr = rtr / 1000.0
        return rtr

    def calcra(self):
        ra = 0.5 * self.rho * self.calcshipspeed() ** 2 * self.calcs() * self.calcca()
        ra = ra / 1000.0
        return ra

    def calcrt(self):
        rt = (
             self.calcrf() * self.calcformfactor()) + self.calcrapp() + self.calcrw() + self.calcrb() + self.calcrtr() + self.calcra()
        return rt

    def calcpe(self):
        pe = self.calcrt() * self.calcshipspeed()
        return pe

    def calcpb(self):
        transbulb = self._ship.transbulbarea
        draughtforep = self._ship.draught
        centerofbulb = self._ship.cobkeel
        pb = 0.56 * math.sqrt(transbulb) / (draughtforep - 1.5 * centerofbulb)
        return pb

    def calcthrust(self):
        thrust = self.calcpe() / self.calcshipspeed() * 1.0 / (1.0 - self.calct())
        return thrust

    def calcps(self):
        ps = self.calcpe() / (
        self.calcnr() * self.calcn0() * self.calcns() * (1.0 - self.calct()) / (1.0 - self.calcw()))
        return ps

    def calcpt(self):
```

```
        pt = self.calcpe() / self.calcnh()
        return pt

    def calcpd(self):
        # Arguments

        pd = self.calcpe() / self.calcnd()
        return pd

    def addseamargin(self):
        # Arguments
        seamargin = 0.85
        sm = self.calcpd() / seamargin
        return sm

    def add85MCRoperation(self):
        # Arguments
        mcr85 = 0.85
        mcr = self.addseamargin() / mcr85
        return mcr

    # Find ship brakepower
    def brakePower(self):
        ns = self._ship.nseff
        brakePower = self.addseamargin() * ns
        return brakePower
```

E | MessageType5.py

```
MessageType 5

port sqlite3 as lite
port sys
om Ship import Ship
om holtrop3 import Holtrop
om collections import Counter
port time
port datetime


f getShipMessage5(databasepath, MMSI, length):
  con = lite.connect(databasepath)
  messagetype = ("MessageType1", "MessageType2", "MessageType3", "MessageType4", "MessageType5",)

  # for x in messagetype[:1]:
  mmsi = MMSI
  table = messagetype[4]

  # print table
  with con:
      curstring = ’select t1.* from MessageType5 t1 where t1.userid={userid} and (t1.dim_bow+t1.dim_stern)={length}’

      cur = con.execute(curstring.format(userid=mmsi, length=length))
      row = cur.fetchone()
      x = row
      ship = Ship(x[17])

      # Unixtime
      unixtime = x[0]

      # Callsign
      ship.callsign = x[1]
      callsign = x[1]

      # Destination
      destination = x[2]

      # Length
      length = x[3] + x[6]
      ship.length = length

      # Breadth
      breadth = x[4] + x[5]
      ship.breadth = breadth
      if (ship.breadth + 0.01) / (ship.length + 0.01) > 0.5:
          ship.status = "rejected"
          ship.statusDescription = "Ship breadth more than 0.5x length"
          return ship
      draught = x[7]


      curstring = ’SELECT AVG (draught) FROM ’ + table + ’ where draught>0 and userid = {userid}’
      cur = con.execute(curstring.format(userid=mmsi))
      row = cur.fetchone()

      draught = row[0]
      if row[0] == 0 or row[0] == None:
          ship.status = "rejected"
          ship.statusDescription = "Ship has no draught"
          return ship
```

```
    ship.draught = draught / 10.0

    curstring = 'SELECT MAX (draught) FROM ' + table + ' where draught>0 and userid = {userid}'
    cur = con.execute(curstring.format(userid=mmsi))
    row = cur.fetchone()
    maxdraught = row[0]
    if maxdraught == 0 or maxdraught == None:
        ship.status = "rejected"
        ship.statusDescription = "Ship has no draught"
        return ship
    ship.maxdraught = maxdraught / 10.0

    curstring = 'SELECT MIN (draught) FROM ' + table + ' where draught>0 and userid = {userid}'
    cur = con.execute(curstring.format(userid=mmsi))
    row = cur.fetchone()
    mindraught = row[0]

    if mindraught == 0 or mindraught == None:
        ship.status = "rejected"
        ship.statusDescription = "Ship has no draught"
        return ship
    ship.mindraught = mindraught / 10.0

    # Dte
    dte = x[8]

    # eta
    eta = x[9]

    # IMO
    ship.imo = x[10]
    imo = x[10]


    # msgid
    ship.msgid = x[11]
    msgid = x[11]

    # Name
    ship.name = x[12]
    name = x[12]

    # Pos_Type
    ship.postype = x[13]
    postype = x[13]

    # Repeat
    repeat = x[14]

    # Ship type
    ship.shiptype = x[15]
    shiptype = x[15]

    # Spare
    spare = x[16]

    # Userid / MMSI
    userid = x[17]

    # Version
    version = x[18]



    curstring = 'SELECT max(sog) FROM MessageType1 where userid = {userid}'
```

```
    # print curstring
    cur = con.execute(curstring.format(userid=mmsi))
    row = cur.fetchone()
    #   print row
    maxspeed = row[0]

    curstring = 'SELECT max(sog) FROM MessageType2 where userid = {userid}'
    #  print curstring
    cur = con.execute(curstring.format(userid=mmsi))
    row = cur.fetchone()
    if row[0] > maxspeed:
        maxspeed = row[0]
    curstring = 'SELECT max(sog) FROM MessageType3 where userid = {userid}'
    cur = con.execute(curstring.format(userid=mmsi))
    row = cur.fetchone()
    if row[0] > maxspeed:
        maxspeed = row[0]

    if maxspeed == None:
        ship.status = "rejected"
        ship.statusDescription = "Ship has no speed records"
        return ship
    maxspeed = maxspeed / 10.0
    ship.maxspeed = maxspeed

    curstring = 'SELECT sum(sog),count(*) FROM MessageType1 where userid = {userid}'
    cur = con.execute(curstring.format(userid=mmsi))
    m1 = cur.fetchone()
    curstring = 'SELECT sum(sog),count(*) FROM MessageType2 where userid = {userid}'
    cur = con.execute(curstring.format(userid=mmsi))
    m2 = cur.fetchone()
    curstring = 'SELECT sum(sog),count(*) FROM MessageType3 where userid = {userid}'
    cur = con.execute(curstring.format(userid=mmsi))
    m3 = cur.fetchone()
    # print m1
    # print m2
    # print m3
    if (m1[0] == None and m2[0] == None and m3[0] == None):
        ship.status = "rejected"
        ship.statusDescription = "Ship has no speed records"
        return ship
    else:
        numRecords = 0
        sumSpeed = 0
        if (m1[0] <> None):
            sumSpeed += m1[0]
            numRecords += m1[1]
        if (m2[0] <> None):
            sumSpeed += m2[0]
            numRecords += m2[1]
        if (m3[0] <> None):
            sumSpeed += m3[0]
            numRecords += m3[1]
        avgspeed = sumSpeed / numRecords
        avgspeed = avgspeed / 10.0
        ship.avgspeed = avgspeed

    # LNG Ships
    if ship.maxdraught <= 13.0 and ship.deltadraught <= 3.5 and ship.maxspeed >= 16.0 and shiptype >= 80 and \
                shiptype <= 89 and ship.breadth > 0:
        ship.cb = 0.72
        ship.sfc = 215.0
        ship.noeff = 0.65
        ship.shiptypegroup = "LNG carrier"
        # General group
```

```
        if ship.breadth >= 40 and ship.breadth <= 52 and ship.length >= 270 and ship.length <= 300:
            ship.shiptypename = "General group"
        # Q-Flex
        elif ship.breadth >= 48 and ship.breadth <= 50 and ship.length >= 314 and ship.length <= 316:
            ship.shiptypename = "Q-Flex"
        # Q-Max
        elif ship.breadth >= 46 and ship.breadth <= 54 and ship.length >= 344 and ship.length <= 345:
            ship.shiptypename = "Q-Max"
        # No match
        else:
            ship.shiptypename = "None of the above LNG ship size"

    # Container ships
    elif ship.maxspeed >= 15.9 and shiptype >= 70 and shiptype <= 79 and ship.breadth > 0:
        ship.cb = 0.60
        ship.sfc = 208.0
        ship.noeff = 0.65
        ship.shiptypegroup = "Container ship"

        # Panamax Container Vessels 1
        if ship.breadth >= 31 and ship.breadth <= 33.0 and ship.length >= 210.0 and ship.length <= 269.9 and \
                        ship.maxdraught <= 10.0 and ship.maxdraught <= 13.0 and ship.deltadraught <= 5.5:
            ship.shiptypename = "Panamax Container Vessels 1"
        # Panamax Container Vessels 2
        elif ship.breadth >= 31 and ship.breadth <= 33.0 and ship.length >= 270.0 and ship.length <= 300.0 and \
                        ship.maxdraught >= 11.0 and ship.maxdraught <= 14.0 and ship.deltadraught <= 5.5:
            ship.shiptypename = "Panamax Container Vessels 2"
        # Post Panamax Container Vessels
        elif ship.breadth >= 40 and ship.breadth <= 43 and ship.length >= 270 and ship.length <= 315 and \
                        ship.deltadraught <= 5.5:
            ship.shiptypename = "Post Panamax Container Vessels"
        # New Panamax Container Vessels
        elif ship.breadth >= 46 and ship.breadth <= 52 and ship.length >= 320 and ship.length <= 370 and \
                        ship.deltadraught <= 5.5:
            ship.shiptypename = "New Panamax Container Vessels"
        # Post New Panamax Container Vessels
        elif ship.breadth >= 54 and ship.breadth <= 58 and ship.length >= 380 and ship.length <= 397 and \
                        ship.deltadraught <= 5.5:
            ship.shiptypename = "Post New Panamax Container Vessels"
        # Trippel E Container Vessels
        elif ship.breadth >= 58 and ship.breadth <= 61 and ship.length >= 397 and ship.length <= 401 and \
                        ship.deltadraught <= 5.5:
            ship.shiptypename = "Trippel E Container Vessels"
        else:
            ship.shiptypename = "None of the above container ship size"

    # Bulk carrier
    elif ship.maxspeed <= 16.0 and shiptype >= 70 and shiptype <= 79 and ship.breadth > 0:
        ship.cb = 0.825
        ship.sfc = 197.0
        ship.noeff = 0.65
        ship.shiptypegroup = "Bulk carrier"

        if ship.breadth >= 30 and ship.breadth <= 34 and ship.length >= 180 and ship.length <= 250 and \
                        ship.mindraught >= 5.0 and ship.deltadraught >= 5.5:
            ship.shiptypename = "Panamax Bulk Carrier"
        elif ship.breadth >= 36 and ship.breadth <= 50 and ship.length >= 230 and ship.length <= 320 and \
                        ship.deltadraught >= 5.0 and ship.maxspeed <= 15.0:
            ship.shiptypename = "Capsize"
        elif ship.breadth >= 29 and ship.breadth <= 33 and ship.length >= 160 and ship.length <= 180 and \
                        ship.deltadraught >= 5.0 and ship.maxspeed <= 15.0:
            ship.shiptypename = "Handymax"
        elif ship.breadth >= 20 and ship.breadth <= 29 and ship.length >= 130 and ship.length <= 180 and \
                        ship.deltadraught >= 5.0 and ship.maxspeed <= 15.0:
            ship.shiptypename = "Handysize"
```

```
        else:
            ship.shiptypename = "None of the above bulk carrier size"

    # Oil tanker
    elif ship.maxspeed <= 18.0 and shiptype >= 80 and shiptype <= 89:
        ship.cb = 0.825
        ship.sfc = 210.0
        ship.noeff = 0.58
        ship.shiptypegroup = "Oil Tanker"

        # ULCC & VLCC
        if ship.breadth >= 50 and ship.breadth <= 70 and ship.length >= 320 and ship.length <= 400 and \
                        ship.maxdraught <= 25.0 and ship.mindraught >= 10.0 and ship.deltadraught >= 8.0:
            ship.shiptypename = "ULCC & VLCC"
        # Suezmax
        elif ship.breadth >= 45 and ship.breadth <= 50 and ship.length >= 265 and ship.length <= 320 and \
                        ship.deltadraught >= 5 and ship.draught <= 20.0:
            ship.shiptypename = "Suezmax"
        # Aframax
        elif ship.breadth >= 38 and ship.breadth <= 44 and ship.length >= 235 and ship.length <= 265 and \
                        ship.deltadraught >= 0.0:
            ship.shiptypename = "Aframax"
        # Panamax Oil Tanker
        elif ship.breadth >= 30 and ship.breadth <= 33.5 and ship.length >= 200 and ship.length <= 235 and \
                        ship.deltadraught >= 4.0:
            ship.shiptypename = "Panamax Oil Tanker"
        else:
            ship.shiptypename = "None of the above Oil tankers size"
    else:
        ship.shiptypegroup = "Type: Ship outside ship groups. Not matching ship contraints"
        if ship.length < 100:
            ship.status = "rejected"
            ship.statusDescription = "Ship length < 100m"
            return ship
        else:
            ship.status = "rejected"
            ship.statusDescription = "Ship outside ship groups. Not matching ship contraints. Ship reports length above 100 m"
            return ship


    # Test to reject vessels with faulty AIS information
    if ship.length == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.length == None"
        return ship

    if ship.lpp == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.lpp == None"
        return ship
    if ship.breadth == None or ship.breadth == 0:
        ship.status = "rejected"
        ship.statusDescription = "ship.breadth == " + str(ship.breadth)
        return ship
    if ship.shiptype == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.shiptype == None"
        return ship
    if ship.displacement == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.displacement == None"
        return ship
    if ship.draught == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.draught == None"
```

```
        return ship
    if ship.maxdraught == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.maxdraught == None"
        return ship

    if ship.mindraught == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.mindraught == None"
        return ship
    if ship.maxspeed == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.maxspeed == None"
        return ship
    if ship.avgspeed == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.avgspeed == None"
        return ship
    if ship.shiptypegroup == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.shiptypegroup == None"
        return ship

    if ship.cb == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.cb == None"
        return ship

    if ship.sfc == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.sfc == None"
        return ship

    if ship.longcofb == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.longcofb == None"
        return ship
    if ship.transbulbarea == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.transbulbarea == None"
        return ship
    if ship.cobkeel == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.cobkeel == None"
        return ship
    if ship.cm == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.cm == None"
        return ship
    if ship.cwp == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.cwp == None"
        return ship
    if ship.transom == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.transom == None"
        return ship
    if ship.sapp == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.sapp == None"
        return ship
    if ship.sternparameter == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.sternparameter == None"
        return ship
```

```
    if ship.propdiameter == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.propdiameter == None"
        return ship
    if ship.numberOfPropellerBlades == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.numberOfPropellerBlades == None"
        return ship
    if ship.propellerClearance == None:
        ship.status = "rejected"
        ship.statusDescription = "ship.propellerClearance == None"
        return ship

    if ship.totalconsumption > 90000:
        ship.status = "rejected"
        ship.statusDescription = "Total consumption way to high, something's wrong!"
        return ship

    curstring = 'SELECT distinct sog FROM MessageType1 where userid = {userid} and sog > 0 ORDER BY sog'
    cur = con.execute(curstring.format(userid=mmsi))
    rows = cur.fetchall()
    curstring = 'SELECT distinct sog FROM MessageType2 where userid = {userid} and sog > 0 ORDER BY sog'
    cur = con.execute(curstring.format(userid=mmsi))
    rows += cur.fetchall()
    curstring = 'SELECT distinct sog FROM MessageType3 where userid = {userid} and sog > 0 ORDER BY sog'
    cur = con.execute(curstring.format(userid=mmsi))
    rows += cur.fetchall()

    speeds = set(rows)

    hm = Holtrop(ship, speeds)
    pb = hm.getBrakePowers()
    ship.pb = pb

    curstring = 'SELECT unixtime,sog FROM MessageType1 where userid = {userid} ORDER BY unixtime'
    cur = con.execute(curstring.format(userid=mmsi))
    rows = cur.fetchall()

    curstring = 'SELECT unixtime,sog FROM MessageType2 where userid = {userid} ORDER BY unixtime'
    cur2 = con.execute(curstring.format(userid=mmsi))
    rows += cur2.fetchall()

    curstring = 'SELECT unixtime,sog FROM MessageType3 where userid = {userid} ORDER BY unixtime'
    cur3 = con.execute(curstring.format(userid=mmsi))
    rows += cur3.fetchall()
    rows.sort()

    speed = None
    time = None
    totalconsumption = 0.0
    total_days = 0
    ship.numSpeedRecords = len(rows)
    for row in rows:
        if speed != None and speed > 0:
            timespan = row[0] - time
            target = speed / 10.0
            if target > 5:  # Record sea days
                total_days += timespan

            power = pb[target]

            if timespan == 0:
                totalconsumption = totalconsumption
            elif target < 30:
                stepconsumption = (ship.sfc / 1000000) * (timespan / 3600) * power
```

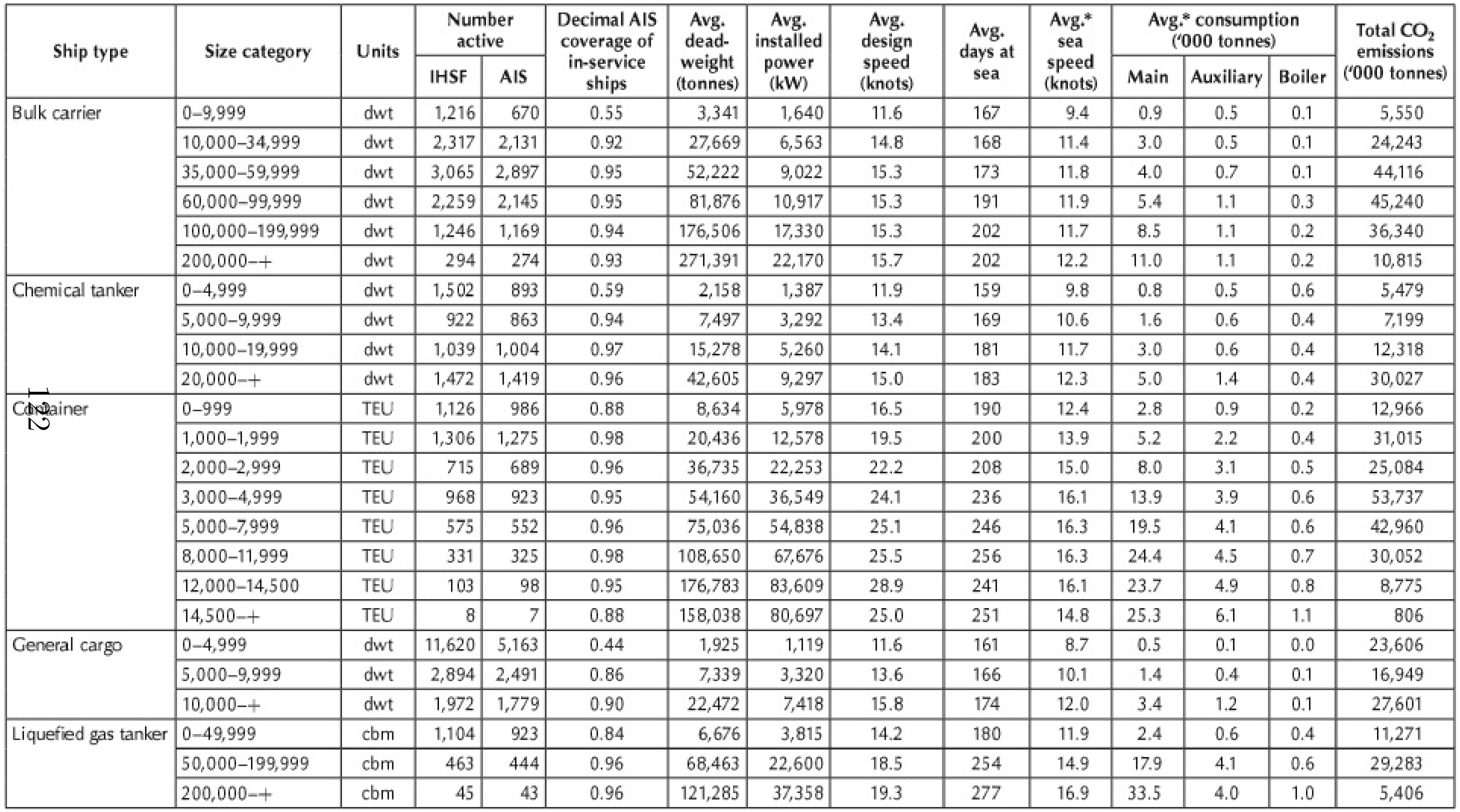
```
                totalconsumption += stepconsumption

        speed = row[1]
        time = row[0]
    ship.totalconsumption = totalconsumption
    ship.totalSailingDays = total_days / (3600.0 * 24.0)
    ship.status = "success"
    ship.statusDescription = "Ship consumption calculated without any errors"
    return ship
```

F | Main.py

```
ain
port MessageType5
port sqlite3 as lite
port time
om tqdm import tqdm
port datetime

om timeit import default_timer as timer

om Ship import Ship

obal allconsumption
lconsumption=0
2emission = 0
4emission = 0
0emission = 0
xemission = 0
vocemission = 0
emission = 0
emission = 0
2emission = 0

obal mmsi

obal databasepath
tabasepath="data/masterdatabase1.db"
atabasepath="data/masterdatabase_reduced.db"

 __name__ == '__main__':
  ships=[]
  con = lite.connect(databasepath)
  rejected_ships = 0
  count =0

  with con:
      curstring = 'select distinct t.userid, (t.dim_bow + t.dim_stern) from ' \
                  '(select t1.* from MessageType5 t1 inner join ' \
                  '(select userid, min(dim_stern+dim_bow) minlength from MessageType5 group by userid) t2 ' \
                  'on (t1.userid=t2.userid and  (t1.dim_bow+t1.dim_stern)=t2.minlength) ' \
                  'where (t1.dim_bow+t1.dim_stern)>=130 and ' \
                  '(t1.dim_bow+t1.dim_stern)<460 and ' \
                  't1.ship_type<100 and ' \
                  'LENGTH(t1.imo)=7 and ' \
                  'LENGTH(t1.userid)=9) t'

      cur = con.execute(curstring)

      rows = cur.fetchall()
      bar = tqdm(total=len(rows))

      for row in rows:

          ship=MessageType5.getShipMessage5(databasepath,row[0],row[1])
          if ship.status == "rejected":
              ships.append(ship)
              rejected_ships += 1
          else:
              ships.append(ship)
              allconsumption += ship.totalconsumption
              co2emission += ship.co2emission
              ch4emission += ship.ch4emission
```

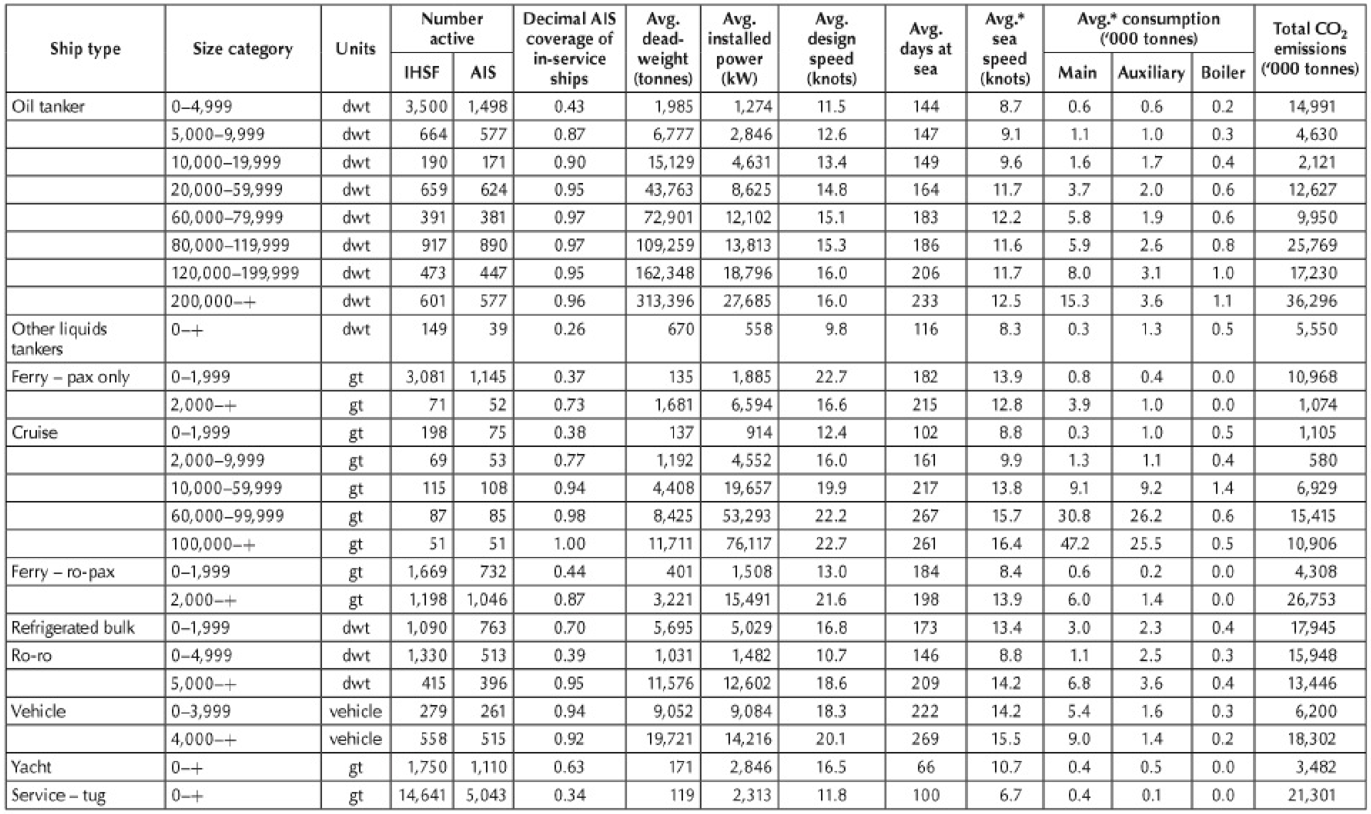
```
            n20emission += ship.n20emission
            noxemission += ship.noxemission
            nmvocemission += ship.nmvocemission
            coemission += ship.coemission
            pmemission += ship.pmemission
            so2emission += ship.so2emission
            count =count+1
        bar.update(1)


bar.write("[" + str(count) + "/" + str(rejected_ships) + "]")
print "Total rejects",rejected_ships
print "Total ships calculated " ,count
print "Consumption for all ships within limits: ", allconsumption
print "Final consumption: ", allconsumption
print "Total CO2 emission ",  co2emission
print "Total CH4 emission ",  ch4emission
print "Total N20 emission ",  n20emission
print "Total NOx emission ",  noxemission
print "Total NMVOC emission ",  nmvocemission
print "Total CO emission ",  coemission
print "Total PM emission ", pmemission
print "Total SO2 emission ",  so2emission
dateString = datetime.datetime.strftime(datetime.datetime.now(), ’%Y-%m-%d %H:%M:%S’)
f = open(’output/output[’+dateString+’].csv’, ’w’)
f.write(ships[0].getCsvHeader()+"\n")
for ship in ships:
    f.write(ship.getCSV()+"\n")
f.close()
```

G | IMO GHG study inventory



Ship type	Size category	Units	Number active		Decimal AIS coverage of in-service ships	Avg. dead-weight (tonnes)	Avg. installed power (kW)	Avg. design speed (knots)	Avg. days at sea	Avg.* sea speed (knots)	Avg.* consumption ('000 tonnes)			Total CO_2 emissions ('000 tonnes)
			IHSF	AIS							Main	Auxiliary	Boiler	
Bulk carrier	0–9,999	dwt	1,216	670	0.55	3,341	1,640	11.6	167	9.4	0.9	0.5	0.1	5,550
	10,000–34,999	dwt	2,317	2,131	0.92	27,669	6,563	14.8	168	11.4	3.0	0.5	0.1	24,243
	35,000–59,999	dwt	3,065	2,897	0.95	52,222	9,022	15.3	173	11.8	4.0	0.7	0.1	44,116
	60,000–99,999	dwt	2,259	2,145	0.95	81,876	10,917	15.3	191	11.9	5.4	1.1	0.3	45,240
	100,000–199,999	dwt	1,246	1,169	0.94	176,506	17,330	15.3	202	11.7	8.5	1.1	0.2	36,340
	200,000–+	dwt	294	274	0.93	271,391	22,170	15.7	202	12.2	11.0	1.1	0.2	10,815
Chemical tanker	0–4,999	dwt	1,502	893	0.59	2,158	1,387	11.9	159	9.8	0.8	0.5	0.6	5,479
	5,000–9,999	dwt	922	863	0.94	7,497	3,292	13.4	169	10.6	1.6	0.6	0.4	7,199
	10,000–19,999	dwt	1,039	1,004	0.97	15,278	5,260	14.1	181	11.7	3.0	0.6	0.4	12,318
	20,000–+	dwt	1,472	1,419	0.96	42,605	9,297	15.0	183	12.3	5.0	1.4	0.4	30,027
Container	0–999	TEU	1,126	986	0.88	8,634	5,978	16.5	190	12.4	2.8	0.9	0.2	12,966
	1,000–1,999	TEU	1,306	1,275	0.98	20,436	12,578	19.5	200	13.9	5.2	2.2	0.4	31,015
	2,000–2,999	TEU	715	689	0.96	36,735	22,253	22.2	208	15.0	8.0	3.1	0.5	25,084
	3,000–4,999	TEU	968	923	0.95	54,160	36,549	24.1	236	16.1	13.9	3.9	0.6	53,737
	5,000–7,999	TEU	575	552	0.96	75,036	54,838	25.1	246	16.3	19.5	4.1	0.6	42,960
	8,000–11,999	TEU	331	325	0.98	108,650	67,676	25.5	256	16.3	24.4	4.5	0.7	30,052
	12,000–14,500	TEU	103	98	0.95	176,783	83,609	28.9	241	16.1	23.7	4.9	0.8	8,775
	14,500–+	TEU	8	7	0.88	158,038	80,697	25.0	251	14.8	25.3	6.1	1.1	806
General cargo	0–4,999	dwt	11,620	5,163	0.44	1,925	1,119	11.6	161	8.7	0.5	0.1	0.0	23,606
	5,000–9,999	dwt	2,894	2,491	0.86	7,339	3,320	13.6	166	10.1	1.4	0.4	0.1	16,949
	10,000–+	dwt	1,972	1,779	0.90	22,472	7,418	15.8	174	12.0	3.4	1.2	0.1	27,601
Liquefied gas tanker	0–49,999	cbm	1,104	923	0.84	6,676	3,815	14.2	180	11.9	2.4	0.6	0.4	11,271
	50,000–199,999	cbm	463	444	0.96	68,463	22,600	18.5	254	14.9	17.9	4.1	0.6	29,283
	200,000–+	cbm	45	43	0.96	121,285	37,358	19.3	277	16.9	33.5	4.0	1.0	5,406

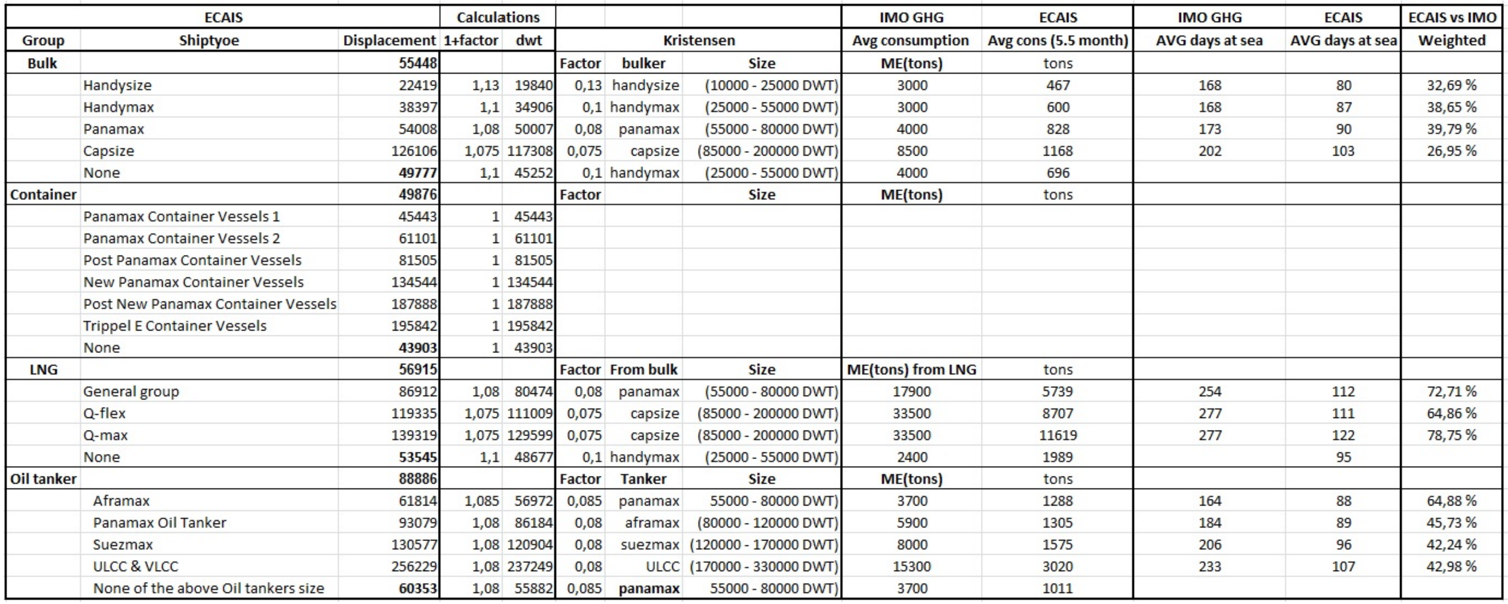
Figure G.1: Fleet description (Smith et al., 2014).



Ship type	Size category	Units	Number active		Decimal AIS coverage of in-service ships	Avg. dead-weight (tonnes)	Avg. installed power (kW)	Avg. design speed (knots)	Avg. days at sea	Avg.* sea speed (knots)	Avg.* consumption ('000 tonnes)			Total CO_2 emissions ('000 tonnes)
			IHSF	AIS							Main	Auxiliary	Boiler	
Oil tanker	0–4,999	dwt	3,500	1,498	0.43	1,985	1,274	11.5	144	8.7	0.6	0.6	0.2	14,991
	5,000–9,999	dwt	664	577	0.87	6,777	2,846	12.6	147	9.1	1.1	1.0	0.3	4,630
	10,000–19,999	dwt	190	171	0.90	15,129	4,631	13.4	149	9.6	1.6	1.7	0.4	2,121
	20,000–59,999	dwt	659	624	0.95	43,763	8,625	14.8	164	11.7	3.7	2.0	0.6	12,627
	60,000–79,999	dwt	391	381	0.97	72,901	12,102	15.1	183	12.2	5.8	1.9	0.6	9,950
	80,000–119,999	dwt	917	890	0.97	109,259	13,813	15.3	186	11.6	5.9	2.6	0.8	25,769
	120,000–199,999	dwt	473	447	0.95	162,348	18,796	16.0	206	11.7	8.0	3.1	1.0	17,230
	200,000–+	dwt	601	577	0.96	313,396	27,685	16.0	233	12.5	15.3	3.6	1.1	36,296
Other liquids tankers	0–+	dwt	149	39	0.26	670	558	9.8	116	8.3	0.3	1.3	0.5	5,550
Ferry – pax only	0–1,999	gt	3,081	1,145	0.37	135	1,885	22.7	182	13.9	0.8	0.4	0.0	10,968
	2,000–+	gt	71	52	0.73	1,681	6,594	16.6	215	12.8	3.9	1.0	0.0	1,074
Cruise	0–1,999	gt	198	75	0.38	137	914	12.4	102	8.8	0.3	1.0	0.5	1,105
	2,000–9,999	gt	69	53	0.77	1,192	4,552	16.0	161	9.9	1.3	1.1	0.4	580
	10,000–59,999	gt	115	108	0.94	4,408	19,657	19.9	217	13.8	9.1	9.2	1.4	6,929
	60,000–99,999	gt	87	85	0.98	8,425	53,293	22.2	267	15.7	30.8	26.2	0.6	15,415
	100,000–+	gt	51	51	1.00	11,711	76,117	22.7	261	16.4	47.2	25.5	0.5	10,906
Ferry – ro-pax	0–1,999	gt	1,669	732	0.44	401	1,508	13.0	184	8.4	0.6	0.2	0.0	4,308
	2,000–+	gt	1,198	1,046	0.87	3,221	15,491	21.6	198	13.9	6.0	1.4	0.0	26,753
Refrigerated bulk	0–1,999	dwt	1,090	763	0.70	5,695	5,029	16.8	173	13.4	3.0	2.3	0.4	17,945
Ro-ro	0–4,999	dwt	1,330	513	0.39	1,031	1,482	10.7	146	8.8	1.1	2.5	0.3	15,948
	5,000–+	dwt	415	396	0.95	11,576	12,602	18.6	209	14.2	6.8	3.6	0.4	13,446
Vehicle	0–3,999	vehicle	279	261	0.94	9,052	9,084	18.3	222	14.2	5.4	1.6	0.3	6,200
	4,000–+	vehicle	558	515	0.92	19,721	14,216	20.1	269	15.5	9.0	1.4	0.2	18,302
Yacht	0–+	gt	1,750	1,110	0.63	171	2,846	16.5	66	10.7	0.4	0.5	0.0	3,482
Service – tug	0–+	gt	14,641	5,043	0.34	119	2,313	11.8	100	6.7	0.4	0.1	0.0	21,301

Figure G.2: Fleet description part 2 (Smith et al., 2014).

H | IMO GHG study inventory



ECAIS			Calculations					IMO GHG	ECAIS	IMO GHG	ECAIS	ECAIS vs IMO
Group	**Shiptyoe**	**Displacement**	**1+factor**	**dwt**	**Kristensen**			**Avg consumption**	**Avg cons (5.5 month)**	**AVG days at sea**	**AVG days at sea**	**Weighted**
Bulk		**55448**			**Factor**	**bulker**	**Size**	**ME(tons)**	tons			
	Handysize	22419	1,13	19840	0,13	handysize	(10000 - 25000 DWT)	3000	467	168	80	32,69 %
	Handymax	38397	1,1	34906	0,1	handymax	(25000 - 55000 DWT)	3000	600	168	87	38,65 %
	Panamax	54008	1,08	50007	0,08	panamax	(55000 - 80000 DWT)	4000	828	173	90	39,79 %
	Capsize	126106	1,075	117308	0,075	capsize	(85000 - 200000 DWT)	8500	1168	202	103	26,95 %
	None	**49777**	1,1	45252	0,1	handymax	(25000 - 55000 DWT)	4000	696			
Container		**49876**			**Factor**		**Size**	**ME(tons)**	tons			
	Panamax Container Vessels 1	45443	1	45443								
	Panamax Container Vessels 2	61101	1	61101								
	Post Panamax Container Vessels	81505	1	81505								
	New Panamax Container Vessels	134544	1	134544								
	Post New Panamax Container Vessels	187888	1	187888								
	Trippel E Container Vessels	195842	1	195842								
	None	**43903**	1	43903								
LNG		**56915**			**Factor**	**From bulk**	**Size**	**ME(tons) from LNG**	tons			
	General group	86912	1,08	80474	0,08	panamax	(55000 - 80000 DWT)	17900	5739	254	112	72,71 %
	Q-flex	119335	1,075	111009	0,075	capsize	(85000 - 200000 DWT)	33500	8707	277	111	64,86 %
	Q-max	139319	1,075	129599	0,075	capsize	(85000 - 200000 DWT)	33500	11619	277	122	78,75 %
	None	**53545**	1,1	48677	0,1	handymax	(25000 - 55000 DWT)	2400	1989		95	
Oil tanker		**88886**			**Factor**	**Tanker**	**Size**	**ME(tons)**	tons			
	Aframax	61814	1,085	56972	0,085	panamax	55000 - 80000 DWT)	3700	1288	164	88	64,88 %
	Panamax Oil Tanker	93079	1,08	86184	0,08	aframax	(80000 - 120000 DWT)	5900	1305	184	89	45,73 %
	Suezmax	130577	1,08	120904	0,08	suezmax	(120000 - 170000 DWT)	8000	1575	206	96	42,24 %
	ULCC & VLCC	256229	1,08	237249	0,08	ULCC	(170000 - 330000 DWT)	15300	3020	233	107	42,98 %
	None of the above Oil tankers size	**60353**	1,08	55882	0,085	**panamax**	55000 - 80000 DWT)	3700	1011			

Figure H.1: Consumption comparison between AIS and third IMO GHG study