Development of a Method for Weather Routing of Ships

Hege Eskild

Marine Technology
Submission date: June 2014
Supervisor: Sverre Steen, IMT

Norwegian University of Science and Technology
Department of Marine Technology

NTNU Trondheim
Norwegian University of Science and Technology
Department of Marine Technology

MASTER THESIS IN MARINE TECHNOLOGY

SPRING 2014

FOR

Hege Eskild

Development of a method for weather routing of ships

An important tool in optimizing the operation of ships on long voyages is weather routing, which means that the route is laid to minimize sailing time or fuel consumption by taking the weather into account. The objective of the master thesis is to develop a re-usable weather routing software implemented in Matlab. The software should have the following features:

- The routing should respect limitations with respect to slamming, green water on deck, cargo safety and possibly other limitations related to operation in extreme weather conditions.
- Added resistance due to irregular waves and wind should be taken into account
- The software shall be able to utilize both hindcast weather data and weather forecast data
- The software should as far as possible be applicable to different ships and different routes.
- If time allows, a user-friendly interface should be implemented.

In the thesis the candidate shall present his personal contribution to the resolution of problem within the scope of the thesis work.

Theories and conclusions should be based on mathematical derivations and/or logic reasoning identifying the various steps in the deduction.

The thesis work shall be based on the current state of knowledge in the field of study. The current state of knowledge should be established through a thorough literature study, the results of this study shall be written into the thesis. The candidate should utilize the existing possibilities for obtaining relevant literature.

The thesis should be organized in a rational manner to give a clear exposition of results, assessments, and conclusions. The text should be brief and to the point, with a clear language. Telegraphic language should be avoided.

The thesis shall contain the following elements: A text defining the scope, preface, list of contents, summary, main body of thesis, conclusions with recommendations for further work, list of symbols and acronyms, reference and (optional) appendices. All figures, tables and equations shall be numerated.

The supervisor may require that the candidate, in an early stage of the work, present a written plan for the completion of the work. The plan should include a budget for the use of computer and laboratory resources that will be charged to the department. Overruns shall be reported to the supervisor.

The original contribution of the candidate and material taken from other sources shall be clearly defined. Work from other sources shall be properly referenced using an acknowledged referencing system.

NTNU Trondheim
Norwegian University of Science and Technology
Department of Marine Technology

The thesis shall be submitted electronically (pdf) in DAIM:

- Signed by the candidate
- The text defining the scope (signed by the supervisor) included
- Computer code, input files, videos and other electronic appendages can be uploaded in a zip-file in DAIM. Any electronic appendages shall be listed in the main thesis.

The candidate will receive a printed copy of the thesis.

Supervisor	: Professor Sverre Steen
Start	: 14.01.2014
Deadline	: 10.06.2014

Trondheim, 14.01.2014

[illegible]

Sverre Steen
Supervisor

Preface

This master thesis is written as the final work of an integrated master in Marin Technology at the Norwegian University of Science and Technology, NTNU, in Trondheim the spring of 2014.

Working on this has been challenging an very rewarding. I have had challenges with both theory and the implementation of it, fortunately these problems have been solved as a result of dedication and hard work.

I would not have been able to finish this thesis without programming help from everyone on the internet. I would like to give thanks to the girls in my office at Tyholt. I have spent almost every day with these girls during this spring. I want to thank them for their support and help in understanding difficult hydrodynamic theory. An extra thanks to Ingrid, who has proofread this thesis. I would also give a great thank you to my supervisor Sverre Steen. He has guided me, and challenged me in to getting a deep understanding of my topic. He always has had time to answer my emails, and has supported me with enthusiasm. Finally, I would like to give thanks to the city of Trondheim. I have lived here for nearly five years, and I could not think of a better city to grow up and get educated in.

Hege Eskild

Hege Eskild

Trondheim, 10.06.2014

Summary

Weather routing is a method within ship routing that is used to avoid rough weather and to find the minimal fuel, time or cost route between ports. Several methods and different software exit and are available to the ship owner or vessel. Three different methods are considered to be the main methods in weather routing: The modified isochrone method, calculus of variation and dynamic programming. In this thesis a shortest path algorithm is used to find the minimal fuel route. This method is similar to the dynamic programming, but the algorithm used to find the route is different.

The vessel used for modeling is a 3500 twenty-foot equivalent unit (TEU) container vessel. The total resistance for a given sea state is defined as the sum of the calm water resistance, the wave added resistance and the wind added resistance. The safety of crew and cargo is ensured by implementing a set of seakeeping criteria. The seakeeping performance of the vessel is calculated with regard to roll, relative vertical acceleration, probability of slamming and probability of deck wetness. The mean added resistance due to waves and the vessel responses is calculated by spectral analysis, where the significant wave height and peak period are the spectral parameters. The MARINTEK program ShipX was used to calculate the motion transfer functions and the attainable speed for different sea states.

Network optimization is used to solve the routing problem. A network of nodes and arcs is constructed between the two ports. The network defines the feasible region for the vessel, meaning all possible ship positions. The nodes are geographical coordinates and the arcs are defined as traveled path between nodes. Each node contains information about the significant wave height, mean wave period, mean wave direction and the U- and V-component of wind. Each arc is assigned a cost, which is the fuel consumption between two nodes. The fuel consumption is calculated by keeping the brake power constant, making the only unknown variable time. The time it takes to travel between two nodes is calculated as $t = \frac{Distance}{Velocity}$, where the velocity is dependent on the total resistance. A shortest path algorithm called Dijkstra's algorithm is used to solve the optimization problem.

A Matlab program was developed to find the minimal fuel route. A GRIB file (GRIdded Binary) including weather data is needed to calculate the resistance and ship responses. In addition to the GRIB file, the input files needed in the program are output files from ShipX containing transfer functions and attainable speed for different sea states. The added resistance due to wind can be calculated either by using estimated wind velocity, or actual wind velocity, i.e. the wind components from the GRIB file. The program can run for all oceans and seasons. By changing the input files from ShipX the user can use this program to find optimal routes for different vessels.

The Matlab program has been run for one vessel, but for different seasons and oceans. The main result obtained in this thesis is that weather routing does not result in fuel saving. The optimal route has approximately the same fuel consumption as the great circle route, meaning the shortest route on a sphere. It can therefore be concluded that the actual distance traveled has a much higher impact on fuel consumption than the added resistance. However, the program can be used to avoid rough weather. The route over the North Atlantic in January deviates from the great circle route, and the fuel consumption is higher. This is because the the roll responses exceeds the allowable roll standard deviation for a large part of feasible region, and the route chosen avoids this. The results also show that the optimal route for estimated and actual wind are almost identical. The fuel consumption for actual wind is consequently lower than for estimated wind. This is partly due to that the estimated wind velocity is overestimated, but mainly due to differences between how the velocity is calculated in Matlab and in ShipX. The overall conclusion is that weather routing can not be used to reduce the fuel consumption, but it is applicable to ensure safety of cargo and crew.

Sammendrag

Vær-ruting er en metode innenfor skipsruting som brukes til å unngå dårlig vær og til å finne den ruten hvor skipet bruker minst brensel, tid eller hvor kostnaden er lavest mellom to havner. Flere metoder og ulike programvare eksiterer i dag, og tilgjengelig for rederi og skip. Tre ulike metoder anses for å være de viktigste metodene i vær-ruting: Den modifiserte isokron metoden, variasjon av parametere og dynamisk programmering. I denne oppgaven er det brukt en algoritme for korteste vei til å finne ruten som resulterer i minst brenselsforbrukt, heretter kalt minste brensels rute.

Fartøyet som benyttes i denne avhandlingen er et 3500 TEU containerskip. Totalmotstanden for en gitt sjøtilstand er definert som summen av stillevannsmotstanden, tilleggsmotstand på grunn av bølger og tilleggsmotstand på grunn av vind. Sikkerheten til mannskap og last tas hensyn til ved å implementere sjødyktighets kriterier i metoden. Disse kriteriene definerer skipets tiltatte responser i rull og vertikal akselerasjon, samt sannsynligheten for bunnslag og vann på dekk. Den gjennomsnittlige tilleggsmotstanden på grunn av bølger og fartøyets responser har blitt beregnet ved hjelp av spektralanalyse, hvor signifikant bølgehøyde og bølgespekterets toppperiode er de spektrale parameterne. MARINTEK programmet ShipX brukes til de hydrodynamiske analysene. Det inkluderer skipets responsfunksjoner og oppnåelig hastighet for forskjellige sjøtilstander.

Nettverkoptimalisering har blitt brukt til å løse rutingproblemet. Et nettverk definerer mulighetsområdet for skipet, det vil si alle skipets tillatte posisjoner. Nettverket er bestående av noder og buer, hvor nodene er geografiske koordinater og buene seilruter mellom nodene. Hver node inneholder informasjon om signifikant bølgehøyde, gjennomsnittlig bølgeperiode, gjennomsnittlig bølgeretning og U- og V-komponenten av vinden. Hver bue blir tilordnet en kostnad, i denne oppgaven er kostnaden definert som drivstofforbruket mellom to noder. Brenselsforbruket er beregnet ved at motorens bremsekraft holdes konstant, og den eneste ukjente variabelen er tid. Tiden det tar å seile mellom to noder blir beregnet som $t = \frac{Avstand}{Hastighet}$, hvor hastigheten er avhengig av totalmotstanden. En algoritme for korteste vei ved navn Dijkstras algoritme brukes til å løse nettverksproblemet.

Et Matlab-program har blitt utviklet for å finne minste brensels rute. Værdataen som brukes i programmet blir hentet ut ifra en GRIB-fil definert av brukeren. I tillegg til GRIB-filen, er flere inputfiler nødvendige for at programmet skal fungere. Skipet defineres av input-filer fra ShipX. Disse inneholder transferfunksjoner og oppnåelig hastighet for ulike sjøtilstander. Tilleggsmotstanden på grunn av vind kan regnes ut i programmet enten ved å bruke estimert vindhastighet, eller faktiske vindhastighet, dvs. vind komponentene fra GRIB -filen. Programmet kan kjøres for alle hav og årstider. Ved å endre input-filene fra ShipX kan brukeren også finne optimale ruter for ulike fartøy.

Matlab-programmet har blitt kjørt for ett fartøy, men for ulike årstider og hav. Hovedresultatet hentet fra denne oppgaven er at vær-ruting ikke resulterer i drivstoffbesparelse. Drivstofforbruket til den optimale ruten er tilnærmet likt som forbruket til den store sirkel ruten, altså korteste vei mellom to havner definert på en sfære. Ut ifra dette kan det forståes at den faktiske avstanden skipet seiler har en mye høyere innvirkning på drivstofforbruket enn tilleggsmotstanden fra bølger og vind. Programmet kan likevel brukes for å unngå dårlig vær. Ruten over Nord-Atlanteren i januar avviker fra den store sirkel ruten, og drivstofforbruket er høyere. Dette er fordi rull responsen overskrider tillatte verdier for store deler av mulighetsområdet, den optimale ruten unngår dette. Resultatene viser også at den optimale ruten for estimert og faktisk vind er nesten identiske. Drivstofforbruket for faktisk vind er lavere enn for estimert vind. Dette er delvis på grunn av at den estimerte vindhastigheten er overestimert, men hovedsakelig på grunn av at hastigheten beregnes litt forskjellig i Matlab og i ShipX. Hovedkonklusjonen i denne avhandlingen er at vær-ruting ikke kan brukes for å spare drivstoff, men er aktuelt for å ivareta sikkerheten til last og mannskap.

Contents

List of Figures

List of Tables

Nomenclature

α	The ship's heading
β	Wave heading
η_0	Propeller efficiency
η_1	Displacement surge
η_2	Displacement sway
η_3	Displacement heave
η_4	Displacement roll
η_5	Displacement pitch
η_6	Displacement yaw
η_D	Propulsive efficiency
η_H	Hull efficiency
η_M	Mechanical efficiency
η_R	Relative rotative efficiency
γ	Peakedness parameter
∇	Volume displacement
ν	Kinematic viscosity
ω	Wave frequency
ω_p	Peak frequency
$\overline{T}_p$	Mean peak period
ρ	Water density
ρ_A	Air density

ε	Phase angle
ζ	Wave elevation
ζ_a	Wave amplitude
A_T	Transverse projected area above water line
A_{33}	Added mass in heave
B	Breadth molded
B_{33}	Damping in heave
B_{WL}	Breadth waterline
C_A	Correlation resistance coefficient
C_B	Block-coefficient
C_D	Wind drag coefficient
C_F	Frictional resistance coefficient
C_M	Midship coefficient
C_P	Prismatic-coefficient
C_R	Residual resistance coefficient
C_T	Total resistance coefficient
C_V	Viscous resistance coefficient
C_{AA}	Air resistance coefficient
C_{AW}	Added resistance coefficient
C_{BD}	Base drag resistance coefficient
H	Wave height
$H(\omega)$	Transfer function
H_S	Significant wave height
J	Advance coefficient
J^*	Full scale propulsion point

k	Wave number
K_Q	Torque coefficient
K_T	Thrust coefficient
L_{PP}	Lenght per perpendiculars
L_{WL}	Lenght on waterline
n	Number of propeller revolutions
P_B	Brake power
P_D	Propulsion power
P_E	Effective power
Q	Propeller torque
R_n	Reynolds number
R_{AA}	Air resistance
R_{AW}	Added resistance
R_{Wind}	Added resistance due to wind
S	Wetted surface
$S(\omega)$	Wave spectrum
T	Draft
T	Propeller thrust
T	Wave period
t	Thrust deduction
t	Time in hours for sailing between two points
T_1	Mean wave period
T_p	Peak period
V	Ship's velocity
V_A	Velocity of advance of the propeller

V_W	Wind velocity
w	Effective wake
ARO	Added resistance operator
BSFC	Brake specific fuel consumption
ECMWF	European Center for Medium-Range Weather Forecasts
FP	Fore perpendicular
GCR	Great circle route
LCB	Longitudinal center of bouyancy
RAO	Response amplitude operator
RMS	Root mean square
TEU	Twenty-foot equivalent unit

1. Introduction and previous work

1.1. Background and motivation

When a vessel sails it is influenced by multiple conditions caused by the weather. Waves, wind and currents will influence the ship's speed and fuel consumption. The objective of weather routing is to find a route that will minimize time, fuel or cost. Weather routing is also used to exclude specific routes where the ship is exposed to rough sea, and avoid possible damages to the vessel, cargo and crew. Depending on the type of vessel, different criteria for acceptable routes can be used. With route planing it's important to find a balance between the minimum time/fuel and the avoidance of rough weather.

Christiansen et al. (2007) stated that commercial ship operations is normally divided into three types: Liner, tramp and industrial shipping. Liner operation means that the sailing route is fixed, and the ports that need to be visited are predefined. Tramp ships follow the available cargo, this is similar to how a taxi operates. Industrial shipping means that the operator normally owns both the vessel and the cargo. The objective of tramp shipping is to maximize the profit, but for liner and industrial shipping, the objective is to minimize the cost. Bunker fuel prices has a great impact on the marine transportation, and the ship owners cost priority changes with the fuel price. Before the oil crises in 1970's, the minimum time routes were most often recommended. However, when the fuel prices are high, minimizing the fuel consumption has a higher priority. Weather routing can be an efficient way of minimizing the fuel cost.

Weather routing of ships is increasingly recognized as an important contribution to safe, economical and reliable ship routes. Weather routing is used in deep sea shipping, typically large ships traveling in the transcontinental oceans. A ship can access the weather forecast by satellite, and update the recommended route one or two times a day. Weather routing agencies provides route recommendations. There are several large weather routing agencies, and with increasing knowledge about the weather and currents there has been a rise in the number of agencies, Thomson (2011).

Service provider	System	Weather forecast	Planning	Optimization	Warning system	Ship monitoring	Data recording	Vessel tracking
Aerospace and Marine International (USA)	Weather 3000, internet service, maps displaying fleet and weather information	X					X	X
Applied Weather Technology (USA)	BonVoyage System, on-board system	X	X					
Deutscher Wetterdienst (Germany)	MetMaster, MetFerry, ashore routing systems, advice on demand		X	X				
Euronav (UK)	seaPro, on-board system, software or fully integrated bridge system	X	X				X	
Finish Meteorological Institute (Finland)	Weather and routing advice from ashore for the Baltic sea	X	X					
Fleetweather (USA)	Meteorological consultancy from ashore	X	X					X
Force Technology (Denmark)	SeaSense, real-time on-board decision support system managing				X	X	X	
Germanischer Lloyd, Amarcon B.V. (Germany, Netherlands)	Shipboard Routing Assistance System SRAS	X	X		X	X	X	
Meteo Consult (Netherlands)	SPOS, on-board routing system	X	X	X			X	
Metworks Ltd. (UK	Meteorological consultancy from ashore	X	X					
Norwegian met office, C-Map (Norway, Italy)	C-Star, on-board system	X	X					
Oceanweather INC., Ocean Systems INC. (USA)	Vessel Optimization and Safety System, VOSS, on-board system	X	X	X	X			
Swedish met and hydrology institute (Sweden)	Seaware Routing, Seaware Routing Plus and Seaware EnRoute Live, on-board systems, and support ashore	X	X		X	X	X	
US Navy (USA)	STARS, on-board system	X	X	X		X		
Weather News International, Oceanwaves (USA, Japan)	VPS and ORION, combination of ashore and onboard routing and optimization software	X	X	X				
Weather Routing Inc. (USA)	Routing advice from ashore and Dolphin navigation program combined with a web-based interactive site	X	X					
Transas (UK)	Ship Guard SSAS, on-board system, software or integrated to bridge system	X	X			X	X	X

Table 1.1.: Routing services and support systems, Hinnenthal (2008)

Tab. 1.1 gives and overview over the main routing agencies and services. The table is obtained from Hinnenthal (2008). A typical agency will provide both onshore and onboard guidance. The agency will give a recommendation prior to the departure, and by monitoring the ship's position recalculations will be performed during the voyage. Several agencies also provide post voyage analysis, where they evaluate the ship's performance with regard to speed, fuel consumption and emissions. With this information the agency can recommend the optimal operational condition as well as the frequency of maintenance. Several nations provide meteorological data that can be used for ship routing. Weathernews Inc. is the world's largest private company providing this data, oceanroutes.com (2013).

As stated in this section weather routing is applicable for all large transoceanic vessels. With different priorities such as minimal time, fuel, cargo safety and more, vessel routing agencies can customize routing strategies for each vessel.

1.2. Objective and limitations

The main objective of this master thesis is to develop a re-usable software for weather routing. The software should be developed in Matlab and include added resistance from irregular waves and wind. The routing program should take cargo and crew safety into account by defining seakeeping criteria for the vessel. The software should be able to use both hindcasted and forecasted weather data, and it should be applicable for different ships and routes.

Limitations with this method is that added resistance from current is not included. The method does not account for land mass, so the user must choose routes that spans over water.

1.3. Methodology

- Matlab program developed to conduct route suggestions and calculate fuel consumption
- ShipX is used to perform hydrodynamic calculations
- Historic weather files are obtained from (ECMWF)
- Network optimization is used to find the minimum fuel path

1.4. Historic perspective

Use of wind, waves and currents for transportation at sea is probably as old a concept as sea transportation itself. Thor Heyerdahl showed that the Polynesian people probably arrived from Peru by using knowledge of wind and the main ocean currents.

In more modern times weather routing was applied by implementing statistical data over ocean areas in route planning. The first example of this is Matthew Fontaine Maury who in the 19th century collected data from ships' log books and made the data available for other vessels. He also generated route recommendations for the different seasons. These routes had a large influence on the transit time. Bowditch (2002) stated: *Average transit time on the New York to California via Cape Horn route was reduced from 183 days to 139 days with the use of his recommended seasonal routes.*

Several private meteorological groups started to provide information about weather to transoceanic ships in the 1950's. This made it possible for the shipping companies to avoid rough weather and choose shorter routes. With the implementation of computers and the internet, this information has been made available to everyone. Today weather forecasts can be used to predict the route, however the reliability of the forecast is crucial in order to predict an optimal route.

1.5. Optimization and previous research

Optimization is a field within applied mathematics which uses mathematical models to determined the best possible solution to a problem, Lundgren et al. (2010). In optimization the goal is to find the best solution or the best possible solution. If the best solution is not possible do to a set criterion, the best possible is considered the best. Three terms are important in optimization: Objective function, decision variable and restriction. The objective function is the function that is minimized or maximized to find the optimal solution, $max\, f(X)$. The objective function includes one or more decision variable, X, which is changed in order to find the optimal solution. The objective function is normally subjected to a set of restrictions called constrains. The constrains defines the feasible region of the function, and limits the value of X.

Optimization has many applications. This includes production planning, transport and logistics, design, and planning and scheduling. This thesis will discuss the transport and routing part of optimization. Christiansen et al. (2007) defines three areas of optimization within shipping. Strategical planning is the long term planning for ship and fleet, with a time frame of several years. This includes ship design, marked and trade, network design and fleet size and mix. Tactical planning has a shorter time horizon and usually spans over a few months. This includes fleet deployment, ship routing and scheduling, ship management and load handling. Operational planning is planning the next few days or a voyage. This includes cruising speed, loading and weather routing.

There are three methods that are mainly used in weather routing. These methods will be briefly presented here.

1.5.1. Modified isochrone method

The isochrone method is a method that is often used in weather routing software. This method was originally proposed by James (1957). Hagiwara (1989) modified the method, and the method is formally known as the modified isochrone method. He presented a method with three different applications: minimize time, fuel or cost. The objective function which is to be minimized for minimal time routing in the modified isochrone method is:

$$J = \int_{t_0}^{t_f} A(X,U,t)dt + B(t_f,t_s) \tag{1.1}$$

Where A(X,U,t) is the decision variable, it represents traveled time, X is the vessels position and U is the control vector including heading and number of propeller revolutions. $B(t_f,t_s)$ is a penalty variable, and it is larger than zero if the vessel arrives after the scheduled arrival. t_0 is the time of departure, t_s is the scheduled time of arrival and t_f is the actual time of arrival. The decision variable for minimal fuel is F, and C for minimal cost, they are dependent on the same parameters as A. The delay penalty varies for the three different objective functions. For minimal fuel routing $B(t_f,t_s) = \infty$ for $t_f > t_s$, and for minimal cost routing $B(t_f,t_s) = w(t_f - t_s)^2$ for $t_f > t_s$ where w is a chosen constant.

An isochrone is defined as a line on a map or diagram connecting places from which it takes the same time to travel. The isochrone methods works in the follow way: Isochrones are constructed with a given time limit from the departure point. The isochrones has different physical length, in nautical miles, due to the variance in resistance for the different ship headings. The engine power is kept constant and the velocity will vary with resistance. New isochrones are again constructed at the end points for the previous isochrones. The isochrones are construed in the same manner until the destination is reached. When the first isochrone meets the destination point, the optimal route has been found. An example of isochrone construction in shown in the fig. 1.1.

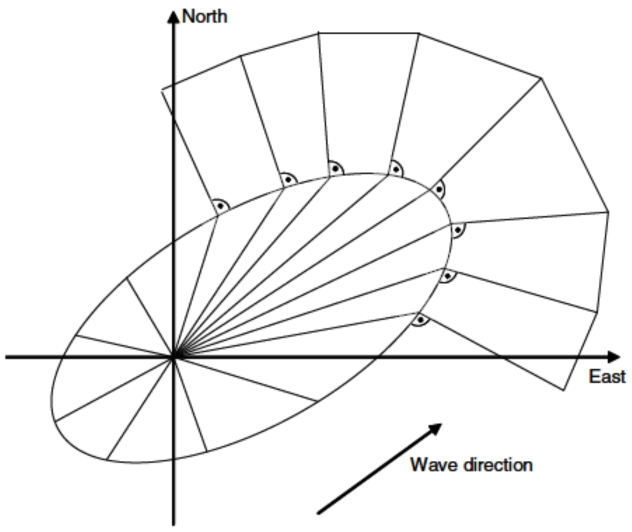


Figure 1.1.: Construction of isochrones, according to Szlapczynska and Smierzchalski (2007)

1.5.2. Calculus of variation

The calculus of variation method was originally proposed Haltiner et al. (1962). Papadakis and Perakis (1990) developed the method further, and were able to find the routes and the vessel's power setting. Perakis and Papadakis (1989) also extended the method to be valid in a time-dependent environment.

The calculus of variation is an analytical approach to weather routing. It considers the course and power output as decision variables.

The objective function in this method is formulated as

$$J = \int_{x_0}^{x_f} \frac{1}{V}(1 + y_x^2)^{1/2} dx \tag{1.2}$$

Where the starting point is defined as (x_0, y_0) and the end point is defined as (x_f, y_f), and

$$V = [P(H) + Q(H)cos(u - \Phi)]R(p) \tag{1.3}$$

Where V is the ship's speed, H denotes the significant wave height, Φ is the wave direction and u is the course. P and Q are positive functions of H and R is a positive increasing function of the power setting. Unlike the modified isochrone method, this method allows a variation in engine power. The wave direction and wave height are known functions of x and y. The objective is to find the appropriate route, y_x, that minimizes the objective function J.

Calculus of variation considers the ship routing problem as a continuous optimization problem. When solving the routing problem with this method the route is first a guess. The optimal route is found by adjusting the heading and power setting. This is done with a set of constrains so that the optimal route is realistic. There are two ways to find the solution: Either the Euler Hamiltonian equation, or with a set of linear equations.

1.5.3. Dynamic programming

Dynamic programming is an optimization strategy where the main idea is to divide the problem into several subproblems that can be solved independently. The problem is divided into stages where the optimal solution is found for a number of subproblems, Lundgren et al. (2010). A state in dynamic programming is defined as a possible solutions in a stage, i.e. the ship's position. When using dynamic programming in weather routing a grid is predefined. The grid consists of geographical points that defines the states and stages. Each point in the grid contains information about the waves and wind. The figure below shows a typical grid design, where the stages are set of points, numbered between 1 and 16, and the states are the points in each stage.

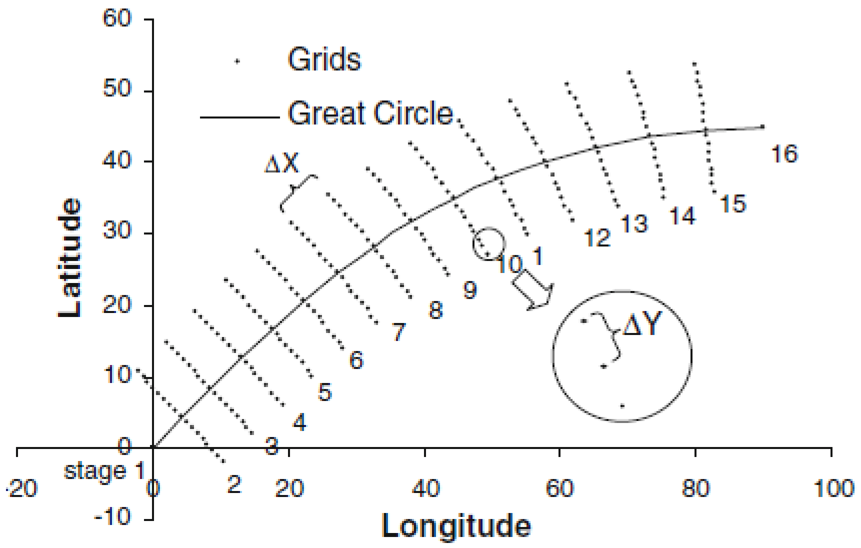


Figure 1.2.: Grid design defined by Shao et al. (2012)

Shao et al. (2012) defined the subproblem as the calculation of fuel consumption between two stages. The optimal fuel consumption is found by summing the fuel consumption between the stages for the entire route. A forward algorithm is used in order to determine the optimal solution. When using a forward algorithm, the optimal solution is defined as: *A path is optimal if and only if, for any intermediate stage, the choice of the previous path is optimum for this stage, Lundgren et al. (2010).* This means that when evaluating the cost for a state in stage 3, one needs to evaluate the total cost from stage 1 and not just from stage 2. The objective function in dynamic programming is:

$$f_t(s_t) = \underset{x_t}{min}[c_t(s_t, x_t) + \sum_{i=t-1}^{T} c_i(s_i, x_i)] = \underset{x_t}{min}[c_t(s_t, x_t) + f_{t-1}(s_{t-1})] \tag{1.4}$$

Where t defines the stage and f_t is the objective function for the given stage. s_t is the state, meaning the point we start from in t. x_t is the decision variable, the point traveled to in stage t, and $c_t(s_t, x_t)$ is the cost, or in this case, the fuel consumption between the points s_t and x_t.

The forward algorithm is complex, and will not be presented here. In this thesis a shortest path algorithm is used to find the minimal fuel path, this method is similar to the dynamic programming method, but the algorithm used is different. An in-depth description of the algorithm used in this thesis can found in chapter 3.

1.6. Structure of thesis

This thesis is composed by six chapters. In chapter 2 the hydrodynamic theory and calculations necessary in weather routing is presented. Network optimization and the routing algorithm is presented in chapter 3. Chapter 4 contains a description of the Matlab program developed in this thesis. Chapter 5 includes the results and discussion, and the final chapter, chapter 6 is the conclusion.

2. Ship hydrodynamics and wave statistics

In order to determine the optimal route between two ports knowledge about the resistance and performance of the vessel must be had. When a vessel is exposed to waves and wind the resistance will vary, and the vessel will experience motions and accelerations. The resistance is divided into calm water resistance, added resistance due to waves and added resistance due to wind. In this chapter the vessel used for modeling is presented. The theory behind the resistance, vessel responses and propulsion calculations is described, and some results of the calculations from ShipX are also presented.

ShipX is a software developed by MARINTEK. It is used in ship design and analysis. The program has several plug-ins that can perform different calculations with regard to seakeeping, resistance, added resistance, station keeping and more. ShipX is used to calculate the the added resistance and the response functions of the vessel. These results are later used in Matlab to determine the optimal route.

2.1. Presentation of ship used for modeling

The ship used for modeling is a container vessel, 3500 TEU. A model test performed at MARINTEK in 1995 is the basis of the data needed. Due to confidentiality of the model test, the details will not be presented, only the ship's general dimensions and the main results with regard to resistance an propulsion is included.

The general dimensions of the ship is presented in tab. 2.1.

Name	Symbol	Unit	Value
Length on waterline	L_{WL}	m	226.27
Length per perpendiculars	L_{PP}	m	233.00
Breath molded	B	m	32.20
Breath waterline	B_{WL}	m	32.20
Draft	T	m	11.00
Volume displacement	∇	m^3	47202.48
Prismatic coefficient	C_P	-	0.602
Block coefficient	C_B	-	0.584
Midship coefficient	C_M	-	0.971
Longitudinal center of buoyancy	LCB	m	-6.78
Wetted surface	S	m^2	8973.03

Table 2.1.: General ship dimensions

The model was tested for both 11 and 12.5 m draft. 11 m is the design draft, and all calculations are therefore performed with this value.

Name	Symbol	Unit	Value
Propeller diameter	D	m	7.5
Pitch ratio at 0.7R	P/D	-	0.9482
Number of blades	Z	-	5
Blade area ratio	A_E/A_0	-	0.75
Cord length	c	m	2.617
Thickness	t	m	0.0906
Mechanical efficiency	η_M	-	0,98
Brake power	P_B	kW	28710
Brake specific fuel consumption	BSFC	g/kWh	166

Table 2.2.: Principal propulsion data

The brake power is the estimated power after the model test, and the brake specific fuel consumption, BSFC, is chosen after evaluating engines with the desired power, WärtsilaEngines (2013).

The result from the model test with regard to service speed and propeller RPM is:

$$V = 23.78\, knots \quad RPM = 114.7\, [rev/min]$$

2.2. Calm water resistance

In shipbuilding the calm water resistance of the vessel is used to determined the propulsion system, even though the ship is rarely subjected to calm water. Lee et al. (1985) stated that the calm water probability in the North Atlantic is 0.7 percent, while the probability of calm water occurring in the North Pacific i 1.3 percent. Knowing this, it might seem strange that the calm water performance is the target for most hull and propulsion optimization. However, this is due to that the contract the shipowner sings with the yard that includes a clause about the desired service speed. The service speed is designed for calm water because another sea state is difficult to determine for the full scale trials.

To determine the calm water resistance of a ship, a towing test is normally done. The objective of preforming a towing test is to obtain the residual resistance coefficient from the total resistance, and use this coefficient to calculate the total resistance of the ship.

The resistance is made dimensionless using this formula:

$$C = \frac{R}{{}^{1}\!/\!{}_{2}\rho S V^2} \tag{2.1}$$

Where R is the resistance, ρ is the water density, S is the wetted surface of the hull and V is the vessel velocity.

The total resistance for the ship is the sum of different resistance components. Steen (2007) defines the total resistance as:

$$C_T = C_R + C_V + C_{AA} + C_{BD} + C_A \tag{2.2}$$

Where C_R is the residual resistance coefficient, C_V is the viscous resistance coefficient, C_{AA} is the air resistance coefficient, C_{BD} is the base drag resistance coefficient and C_A is the correlation resistance coefficient. The residual coefficient in model scale is assumed to be equal to the residual coefficient in full scale. This coefficient can be found from model test using the following formula:

$$C_{RM} = C_{TM} - (1 + k) \cdot C_F - C_{AAM} - C_{BDM} \tag{2.3}$$

Where k is the ship's form factor. The form factor factor accounts for the increase in frictional resistance due to the increase of water velocity around the hull. The total resistance is the measured resistance.

The towing test used in this analysis was performed for velocities between 20 and 26 knots. In this project it is reasonable to assume that the velocity might be lower than 20 knots when the vessel is exposed to rough sea. The resistance curve must therefore be extrapolated to lower velocities. This is done by extrapolating the residual resistance coefficient, and calculating the remaining resistance coefficients. The values for C_R is collected from the test results, and the exponential regression formula $C_R = 0.0052e^{0.2039V}$ has been used to calculate the residual resistance curve.

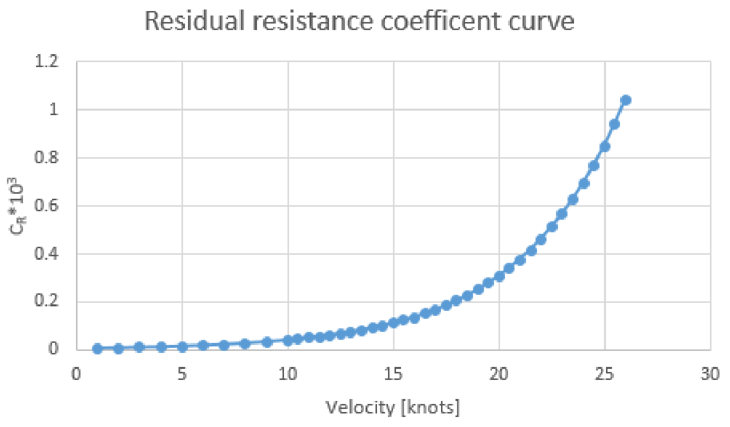


Figure 2.1.: Residual resistance coefficient curve

The base drag, air and correlation resistance coefficients can be assumed not to vary to with speed. The base drag resistance is a resistance caused by the transom stern being submerged. The correlation coefficient is an all inclusive correction factor that takes errors in the scaling procure into account. The form factor and the three coefficients mentioned above were calculated in the model test, and can be used directly in the extrapolation of the resistance curve.

The viscous resistance coefficient in full scale is calculated as:

$$C_{VS} = (1+k)(C_{FS} + \Delta C_F) \tag{2.4}$$

The ITTC standard for calculating the frictional resistance coefficient:

$$C_F = \frac{0.075}{(logR_n - 2)^2} \tag{2.5}$$

Where $\mathrm{R_n}$is the Reynolds number.

$$R_n = \frac{V \cdot L_{WL}}{\nu} \tag{2.6}$$

ν is the kinematic viscosity. ΔC_F is:

$$\Delta C_F = (110.31(H \cdot V)^{0.21} - 403.33)C_F^2 \tag{2.7}$$

Where H is the surface roughness of the hull. A typical value for this is 150μ(10^{-3}mm).

To find the total resistance all the coefficients can be inserted in equation 2.2, and recalculated to Newton using equation 2.1.

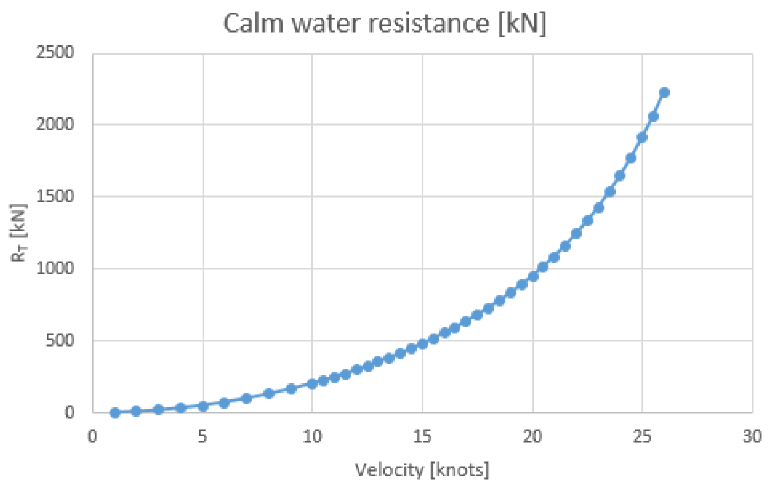


Figure 2.2.: Extrapolated resistance curve

2.3. Propulsion

To find the brake power required the propulsion factors are found. An open water test is when the propeller is tested in a cavitation tunnel or in a towing tank without influence of the vessel. The aim of performing an open water test is to determined the propeller characteristics.

During the open water test measurements are performed with regard to the torque, Q, thrust, T, rate of revolutions, n, and the velocity of advance, V_A, Steen and Minsaas (2013). The propeller efficiency is calculated as in equation 2.8.

$$\eta_0 = \frac{K_T}{K_Q} \cdot \frac{J}{2\pi} \tag{2.8}$$

Where the torque coefficient is:

$$K_Q = \frac{Q}{\rho n^2 D^5} \tag{2.9}$$

and the thrust coefficient is:

$$K_T = \frac{T}{\rho n^2 D^4} \tag{2.10}$$

D is the propeller diameter. The advance coefficient is calculated as:

$$J = \frac{V_A}{n \cdot D} \tag{2.11}$$

The results from the model test at MARINTEK is presented in fig. 2.3. For a given advance coefficient we can read the thrust and torque and determine the propeller efficiency.

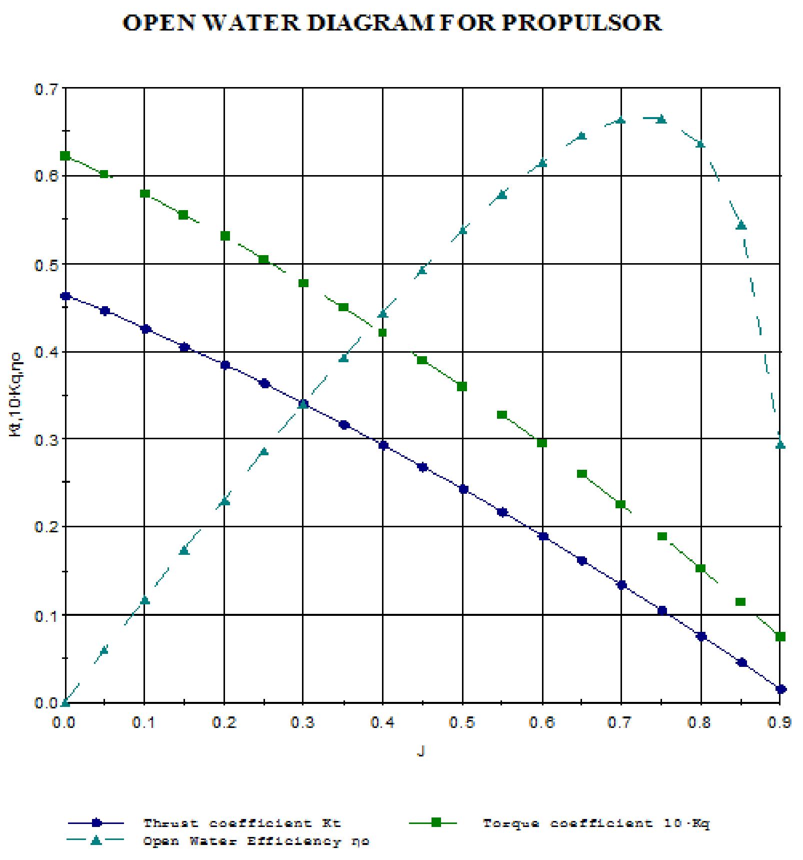


Figure 2.3.: Open water diagram

In an open water test the velocity V_A is equal to V. However when the propeller is installed behind the hull, the hull will influence the velocity of the water flowing around the propeller. The velocity of advance is influenced by wake.

$$V_A = V(1 - w) \tag{2.12}$$

Where V is the ship's speed and w is the effective wake. A propulsion test is performed in order to determined the wake, thrust deduction, t, and the full scale propulsion point, J*. The full scale propeller efficiency is found by determining the full scale propulsion point. J* is determined by finding the point where the intersection between K_T and J^2 is equal to the equation below.

$$\frac{K_T}{J^2} = \frac{R_{TS}}{\rho_S \cdot (1 - t) \cdot D^2 V_S^2 \cdot (1 - w_s)^2} \tag{2.13}$$

The propeller rate of revolution can be calculated as:

$$n = \frac{(1 - w)}{D} \cdot \frac{V_s}{J^*} \tag{2.14}$$

From the propulsion test the hull efficiency, η_H, and the relative rotative efficiency η_R can be found as:

$$\eta_H = \frac{1 - t}{1 - w} \tag{2.15}$$

$$\eta_R = \frac{Q_0}{Q} \tag{2.16}$$

Where Q is the torque of the propeller from the open water test, and Q_0 is the torque of the propeller behind the vessel. η_H and η_R are speed depended, but the variation with speed is small. When the ship has a low speed variation η_H and η_R are often considered to be constant. Nakamura and Naito (1975) stated that the coefficients are not significantly influenced by the waves, so one does not need to consider the effect of waves when calculating the coefficients.

The brake power, P_B, for the vessel can now be calculated with the following equations.

$$P_B = \frac{P_D}{\eta_M} \tag{2.17}$$

$$P_D = \frac{P_E}{\eta_D} \tag{2.18}$$

$$P_E = R_T \cdot V \tag{2.19}$$

$$\eta_D = \eta_H \cdot \eta_0 \cdot \eta_R \tag{2.20}$$

P_D is the propulsive power, η_M is the mechanical efficiency, η_D is the propulsive efficiency and P_E is the effective power.

The model test provides all the relevant information in order to calculate the brake power for both calm water and the required power for a given sea state. When combining the resistance and propulsion calculation ShipX gives the following calm water performance prediction:

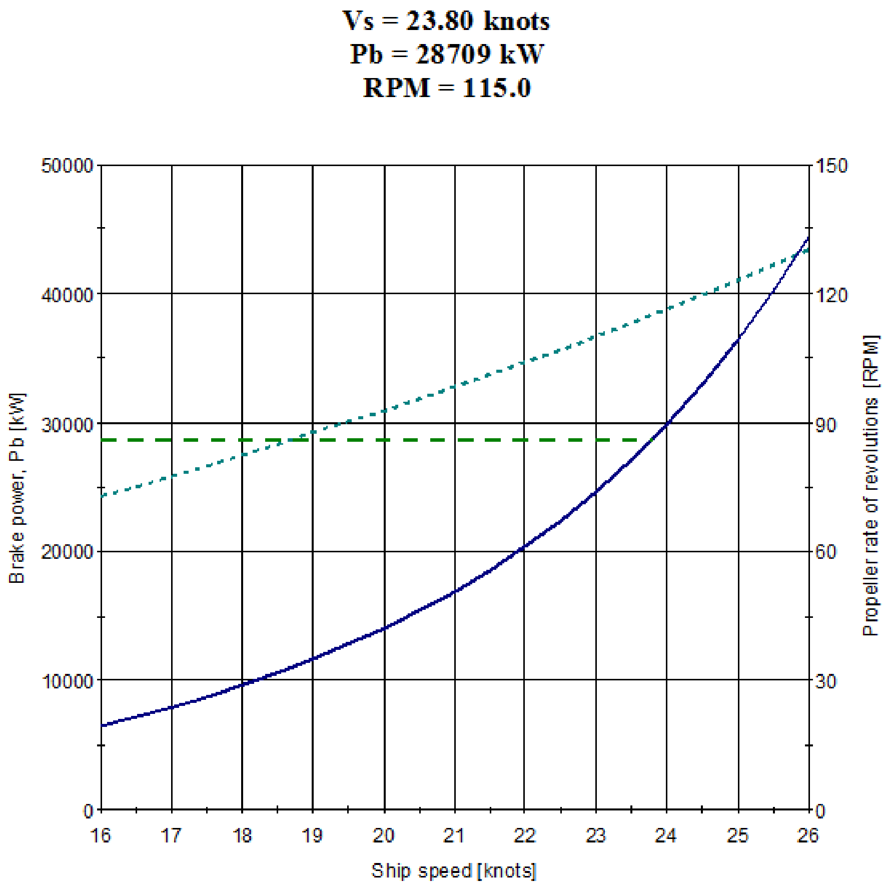


Figure 2.4.: Calm water performance prediction

We can see that with a brake power of 28 709 kW the service speed is calculated to be 23.8 knot and the propeller RPM is 115. The results from ShipX correspond with the results from the model test.

In this thesis the vessel will sail with a constant brake power, so the main objective is to find the attainable speed for a given sea state. The brake power is set to be constant for practical purposes. The engine is designed for a service power, when running the engine at this loading it will result in less damages and better economic fuel consumption. It is also easier to keep a constant power, than to adjust the power to obtain a constant speed. The efficiencies and the the total resistance, including calm water, wind and wave resistance, depend on the velocity of the vessel. Finding the velocity is an iteration process where the total resistance, velocity and efficiencies are calculated until the accurate brake power is found.

2.4. Statistic modeling of waves

knowledge about the irregular waves must be had to investigate a ship's movement and resistance. Waves are generated by steady wind blowing over an area. The time the wind has blown is referred to as duration, while the stretch or area the wind has blown is referred to as fetch. A single wave is mainly composed by the wave generated by the current wind and the swell. Swell are waves generated from wind blowing far away from the given sea area. This section will not describe the generation of waves and wind, but rather the properties and application of irregular wave theory. The theory in this section and in sec. 2.5 is mainly obtained from Lloyd (1998) and Faltinsen (1990).

A wave forecast typically includes significant wave height, H_S, mean wave period, T_1 and mean wave direction. A typical wave elevation profile of an irregular wave is shown in fig. 2.5.

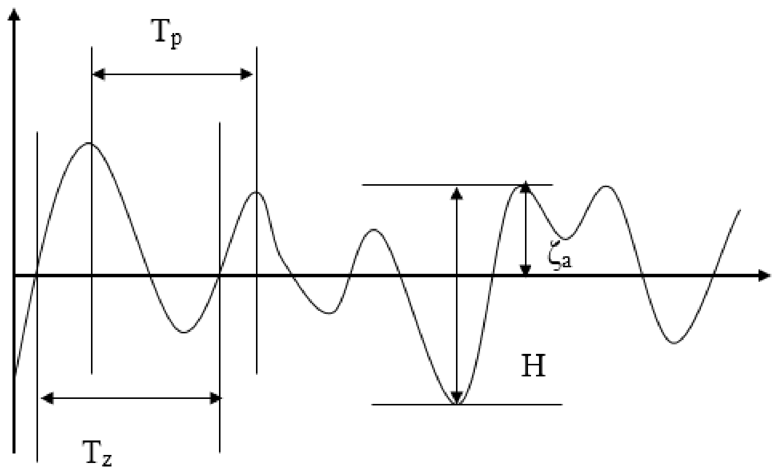


Figure 2.5.: Irregular wave profile

An explanation of different wave parameters is given in tab. 2.3.

Abbreviation	**Definition**	**Explanation**
T	Wave period	Time in seconds between to waves
T_1	Mean wave period	Mean wave period for irregular waves
$\overline{T}_p$	Mean peak period	Mean time between two successive peaks
T_p	Peak period	Peak period of wave spectrum
T_z	Zero crossing period	Period between waves at zero elevation
ω	Wave frequency	Wave frequency in rad/s $\omega = \frac{2\pi}{T}$
ω_p	Peak frequency	Peak frequency of wave spectrum
ζ	Wave elevation	Wave elevation for a given wave frequency
ζ_a	Wave amplitude	Wave amplitude
H	Wave height	Height in meters $H = 2\zeta_a$
H_s	Significant wave height	Mean height of the 1/3 highest waves
$H_{1/3}$	Significant wave height	Estimate for H_s by measured wave heights
β	Wave heading	Heading in degrees, 0° is head sea

Table 2.3.: Wave parameters

With these parameters the sea state can be modeled, and the mean added resistance as well as the responses can be calculated for the vessel.

2.4.1. Wave energy spectrum

The wave elevation in irregular sea can be defined as the sum of regular waves propagating along the positive x-axis.

$$\zeta(t) = \sum_{n=1}^{N} \zeta_{An} cos(\omega_n t + \varepsilon_n) \tag{2.21}$$

Where t represents time and ε is the phase angle for a given wave. The sea state can be described by a wave spectrum, $S(\omega)$, or by its full name wave amplitude energy density spectrum. The spectrum describes the wave frequency distribution for a given sea state. The spectral ordinate is defined as:

$$S(\omega) = \frac{\zeta_A^2}{2\Delta\omega} \tag{2.22}$$

Several different ways of formulating the wave spectrum have been developed.

The JONSWAP spectrum (Joint North Sea Wave Project), was developed as a multinational project. The sea state was observed in the southeast part of the north sea, Myrhaug (2007). The JONSWAP spectrum is a two parameter spectrum, which means that in order to formulate the spectrum the peak period and significant wave height must be known. The JONSWAP spectrum does not represent fully developed sea. Fully developed sea means a sea that has been exposed to wind that has been blowing over a significantly long stretch. The JONSWAP spectrum also includes a peakedness parameter γ, that provides information about how much of the energy is concentrated around the peak frequency.

The PM spectrum (Pierson-Moskowitz), is suitable for fully developed sea, and it is based on data from the north Atlantic. The PM spectrum is a one parameter spectrum. For the PM this parameter is the wind velocity at 19.5 m above the sea surface. This spectrum it equal to the JONSWAP spectrum when $\gamma = 1$

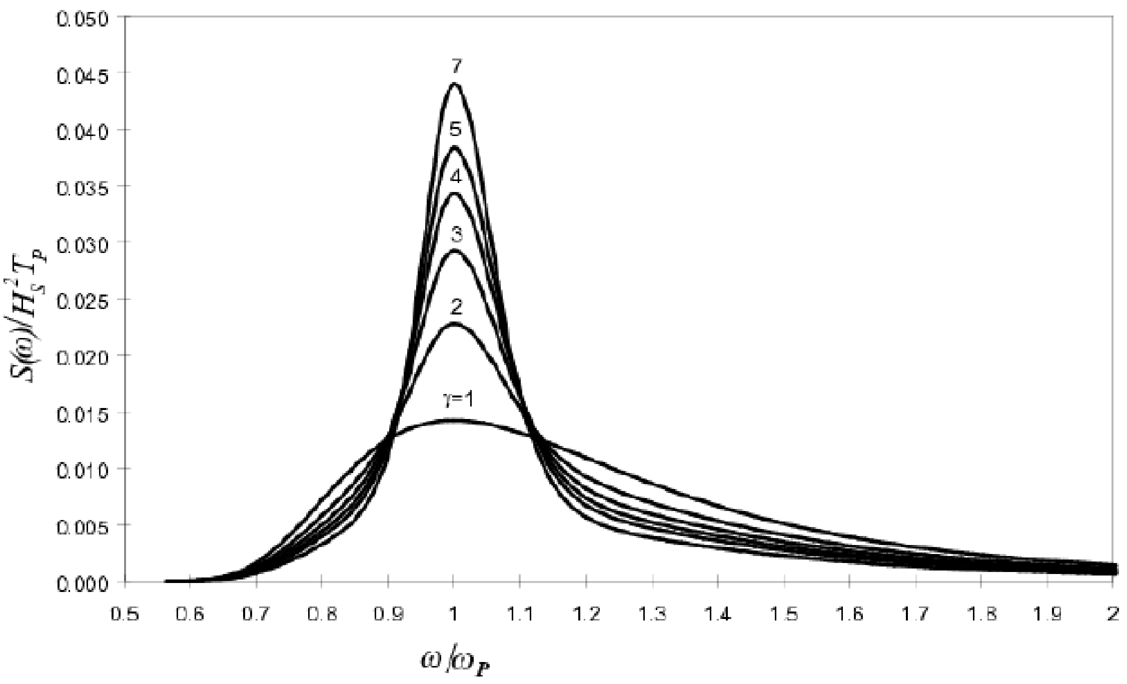


Figure 2.6.: JONSWAP spectra for γ=1-7, Fathi (2012)

The figure above shows the JONSWAP spectrum for different values of γ. Faltinsen (1990) states that the 17th ITTC (International Towing Tank Conference), recommends using the JONSWAP spectrum. Below follows a more detailed explanation of the spectrum formulation.

$$S(\omega) = \alpha g^2 \omega^{-5} e^{-\frac{5}{4}(\frac{\omega_P}{\omega})^4} \gamma^{e^{-\frac{1}{2}(\frac{\omega-\omega_P}{\sigma\omega_P})^2}} \tag{2.23}$$

Where α is the spectral parameter, g is the acceleration of gravity, ω_p is the peak frequency, γ is the peakedness parameter and σ is the spectral width parameter.

$$\alpha = 5.061 \frac{H_s^2}{T_p^4}(1 - 0.287 ln\gamma) \tag{2.24}$$

The spectral width parameter varies with ω:

$$\sigma = \begin{cases} 0.07 & for\, \omega < \omega_P \\ 0.09 & for\, \omega > \omega_P \end{cases} \tag{2.25}$$

By knowing the spectral values of different wave frequencies, the spectral moments can be found. The spectral moments are defined as the variance of the irregular sea. These can be used to estimate wave periods and significant wave height. The spectral moments can be calculated by integrating the wave spectrum:

$$m_n = \int_0^\infty \omega^n S(\omega) d\omega \tag{2.26}$$

Where n equals 0 for the variance in elevation, 2 for the velocity variance and 4 for acceleration variance of the waves. By knowing these moments, as well as m_1, the estimates for periods and significant height are as follows:

$$T_1 = 2\pi \frac{m_0}{m_1} \tag{2.27}$$

$$T_p = 2\pi \sqrt{\frac{m_2}{m_4}} \tag{2.28}$$

$$T_z = 2\pi \sqrt{\frac{m_0}{m_2}} \tag{2.29}$$

$$H_s = 4\sqrt{m_0} \tag{2.30}$$

As previously mentioned, a wave forecast typically provides T_1 and H_s, and to formulate the spectrum a value for T_p must be calculated. Lloyd (1998) provides a relationship between the spectral peak period and the mean wave period.

$$T_p = 1.296 T_1 \tag{2.31}$$

With this, the peak frequency can be found, and the spectrum can be formulated for a chosen peakedness parameter.

2.5. Hydrodynamics in waves and wind

When a ship operates in waves and wind it can be subjected to both voluntary and involuntary speed loss. Voluntary speed loss is due to the masters decision to reduce the speed because of motions and acceleration. Involuntary speed loss is due to the added resistance of wind and waves. The first section considers the motions and accelerations, while in the next section the involuntary speed loss is discussed.

2.5.1. Motions and accelerations in irregular sea

Evaluating the ship's motions in irregular sea was for many years a hard task. One has simplified the sea state as regular waves, this however does not provide a realistic representation of the ship's motions. Researchers in the past decades have provided methods to analyze the ship motions in irregular sea. According to Denis and Pierson (1953) the vessel response can be treated like an analog filter. This means that the filter input are waves with different frequencies, and the output are the ship's motions and accelerations.

Motions in regular head waves must be understood to evaluate the vessel in irregular sea. The theory in this section is based on some basic assumptions:

- Linear theory is applied. This means that there is a linear relationship between the vessel response and incident wave amplitude.
- The vessel is slender. The length of the hull is much larger than its breath and draft.
- Strip theory is is applied. The vessel can be divided in to sections and the total forces can be found by integrating over the length of the ship.
- Potential theory is applied. The water is assumed to be irrational, incompressible and non-frictional.

The ship is defined as rigid body that is free to move in six degrees.

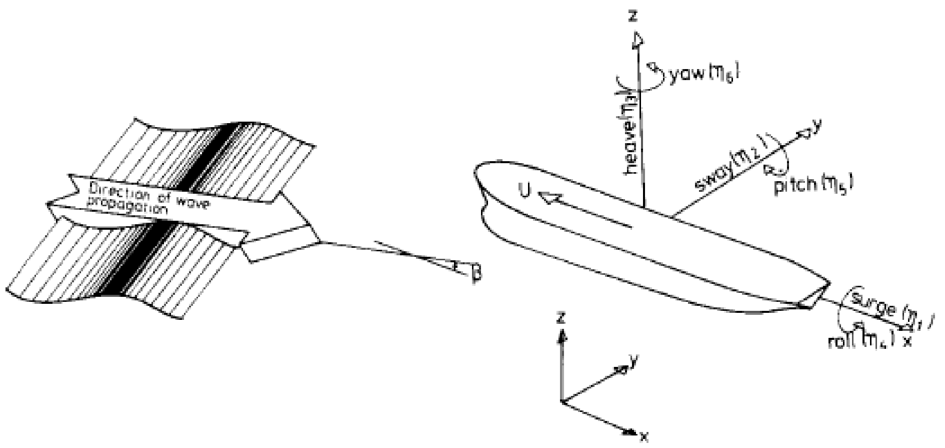


Figure 2.7.: Ship with six degrees of freedom, Faltinsen (1990)

As shown in the figure above the translatory displacements are defined as surge (η_1), sway (η_2), and heave (η_3) for the x-,y- and z direction. The angular displacements of the coordinate system are defined as roll (η_4), pitch (η_5) and yaw (η_6). Any motion of any point at the ship can be describes as

$$s = (\eta_1 + z\eta_5 - y\eta_6)\vec{i} + (\eta_2 - z\eta_4 + x\eta_6)\vec{j} + (\eta_3 + y\eta_4 - x\eta_5)\vec{k} \tag{2.32}$$

Where x, y and z are positions at the vessel and $\vec{i}$, $\vec{j}$ and $\vec{k}$ are the unit vectors along the x-, y- and z-axis. The response functions are defined as:

$$\eta_j = \eta_{ja} cos(\omega t + \varepsilon_n) \quad j = 1, 2...6 \tag{2.33}$$

Where η_{ja} is the motion amplitude and ε is the phase angle. As well as the wave frequency, the response functions are dependent on forward speed and wave heading. The responses are found by solving the equation for rigid body motions:

$$\sum_{k=1}^{6} [(M_{jk} + A_{jk})\ddot{\eta}_k + B_{jk}\dot{\eta}_k + C_{jk}\eta_k] = F_j e^{i\omega t} \tag{2.34}$$

Where M_{jk} is the generalized mass matrix, A_{jk} is the added mass matrix, B_{jk} is the linear damping matrix, C_{jk} is the stiffness matrix and F_j is the complex amplitude of the wave exciting forces and moments.

It is common to use the transfer functions to evaluate the motions of the ship. The transfer function for the respective degree of freedom is defined as the response divided by the wave amplitude.

$$H_j(\omega) = \frac{\eta_j(\omega)}{\zeta_a} \tag{2.35}$$

$H(\omega)$ is also called the response amplitude operator, RAO.

The response of the vessel in irregular sea is found by combining the transfer function with the wave spectrum, see equation 2.26. When investigating the motions for a vessel at sea it is common to look at the standard deviation of the response, $\sigma_n = \sqrt{m_n}$ where n represent the different moments. The calculation for σ is as follows.

$$\sigma_n^2 = \int_0^\infty \omega^n S(\omega)|H(\omega)|^2 d\omega \quad n = 0, 2, 4 \tag{2.36}$$

Where n=0 gives the standard deviation in displacement, n=2 gives the standard deviation in velocity and n=4 gives the standard deviation in acceleration. When evaluating seakeeping the roll movement and the relative vertical movements are of particular interest. This is because these are the critical responses of the vessel. The standard deviation for roll is:

$$\sigma_{roll}^2 = \int_0^\infty S(\omega)|H_4(\omega)|^2 d\omega \tag{2.37}$$

At sea the standard deviation for roll should not exceed a certain value to ensure the safety of crew and cargo. This criterion depends on the vessel type and the purpose of the operation. According to NORDFORSK (1987) the seakeeping criteria for merchant ships is $\sigma_{roll} < 6°$.

The calculation of the transfer function for the relative vertical motions is a bit more complex. Since the movement is in z-direction, the expression corresponding to $\vec{k}$ in equation 2.32 is valid. The point evaluated is normally located at the center line of the ship, which gives y=0. The displacement function for the relative motions are as follows:

$$\eta_R = \eta_3 - x\eta_5 - e^{-ik(xcos\beta + ysin\beta)} \tag{2.38}$$

Where $e^{-ik(xcos\beta+ysin\beta)}$ describes the undisturbed wave elevation at a given position, k is the wave number and β is the wave heading. The transfer functions are complex numbers, compounded by a real and an imaginary part. Kreyszig (2006) provides a method for calculating the relative vertical transfer function. The derivation of the calculation is shown below.

$$\eta_j = r_j e_j^{i\theta_j} \tag{2.39}$$

$$r_j = \sqrt{real_j^2 + imaginary_j^2} \quad \theta_j = arctan(\frac{imaginary_j}{real_j}) \tag{2.40}$$

$$H_R(\omega) = \frac{r_3 e^{i\theta_3} - x r_5 e^{i\theta_5} - e^{-ik(xcos\beta + ysin\beta)}}{\zeta_a} \tag{2.41}$$

The relative vertical standard deviation for displacement, velocity and acceleration can now be found.

Relative vertical movement:

$$\sigma_0^2 = \int_0^\infty S(\omega)|H_R(\omega)|^2 d\omega \tag{2.42}$$

Relative vertical velocity:

$$\sigma_2^2 = \int_0^\infty \omega^2 S(\omega)|H_R(\omega)|^2 d\omega \tag{2.43}$$

Relative vertical acceleration:

$$\sigma_4^2 = \int_0^\infty \omega^4 S(\omega)|H_R(\omega)|^2 d\omega \tag{2.44}$$

NORDFORSK (1987) has different criteria for vertical acceleration.

$$RMS\,of\,vertical\,acc\,at\,FP \leq 0.275g \quad for\,L \leq 100m$$
$$RMS\,of\,vertical\,acc\,at\,FP \leq 0.050g \quad for\,L \geq 300m$$

2.5.1.1. Probability of slamming

Another phenomenon preferred to avoid at sea is slamming. Slamming is a phenomenon that describes loads with high pressure peaks. When a hull hits the water with high velocity these loads can occur. Slamming loads could have consequences for crew and cargo, and if the slamming loads are frequent, it can also cause fatigue and deformation of the hull. The bow of the ship is the part where slamming loads are most probable to occur, and therefore this is the region where the calculation is performed. For the responses the standard deviation is used as operability criterion, while when looking at slamming it is normal to calculate the probability of slamming occurring for a given wave spectrum. According to Faltinsen (1990) the probability of slamming occurring can be calculated as:

$$P(slamming) = e^{-(\frac{V_{cr}^2}{2\sigma_2^2}\frac{d^2}{2\sigma_0^2})} \tag{2.45}$$

Where d is the draft and V$_{\mathrm{cr}}$ is the threshold velocity.

$$V_{cr} = 0.093\sqrt{gL} \tag{2.46}$$

NORDFORSKs seakeeping criteria for slamming is

$$P(slamming) \leq 0.03 \quad for\,L \leq 100m$$
$$P(slamming) \leq 0.01 \quad for\,L \geq 300m$$

2.5.1.2. Probability of green water on deck

While large waves may cause the hull to exit the water and slam back into it, large waves may also cause water flows onto the deck of a ship. This is refereed to as green water or deck wetness, and may cause damage to the cargo, and in danger the crew. Water on deck might also influence the stability of the vessel, proper draining is crucial. To calculate the probability of green water the standard deviation for motions must be known.

$$P(water\ on\ deck) = e^{-\frac{d^2}{2\sigma_0^2}} \tag{2.47}$$

Where d in this this case is the freeboard. NORDFORSKs seakeeping criteria for water in deck is

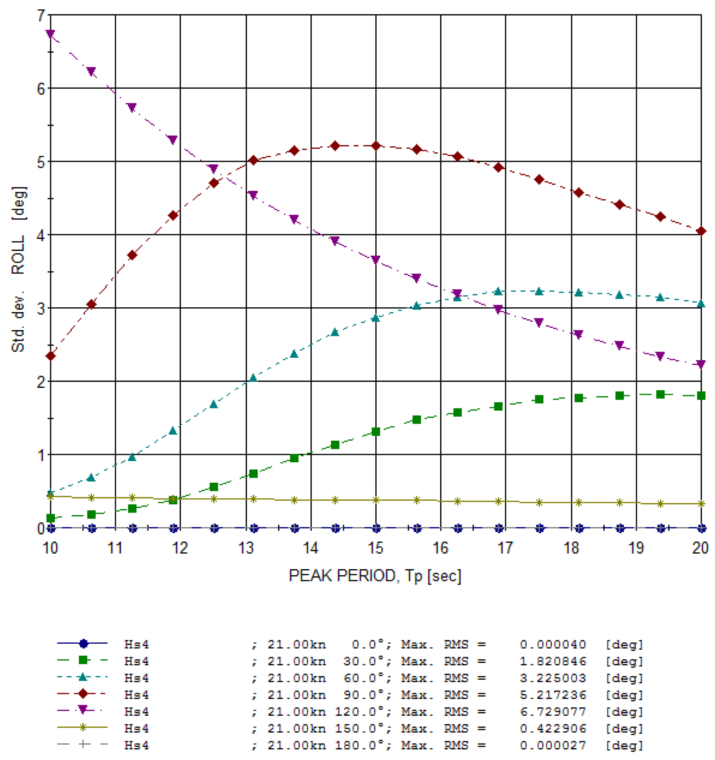
$$P(water\ on\ deck) \leq 0.05$$

2.5.2. Motions and accelerations calculations

The ShipX plug-In Veres has several applications, one is to calculate the ship motions in regular, as well as irregular sea. It can also be used to calculate the added resistance operator. Veres includes four postprocessors that provide the possibility of calculating motions in irregular waves as well as finding the operability region for the vessel. By defining a criterion, for example slamming, ShipX can provide information on the limiting wave height or period. For the purpose of this work the general transfer function file is the target of the Veres analysis. However, the postprocessors for calculating operability and irregular wave responses are useful for comparing the calculations performed in Matlab. The transfer functions are presented in a text file that includes the real and imaginary part of the transfer function for six degrees of freedom. The functions will be calculated for a number of velocities and headings.

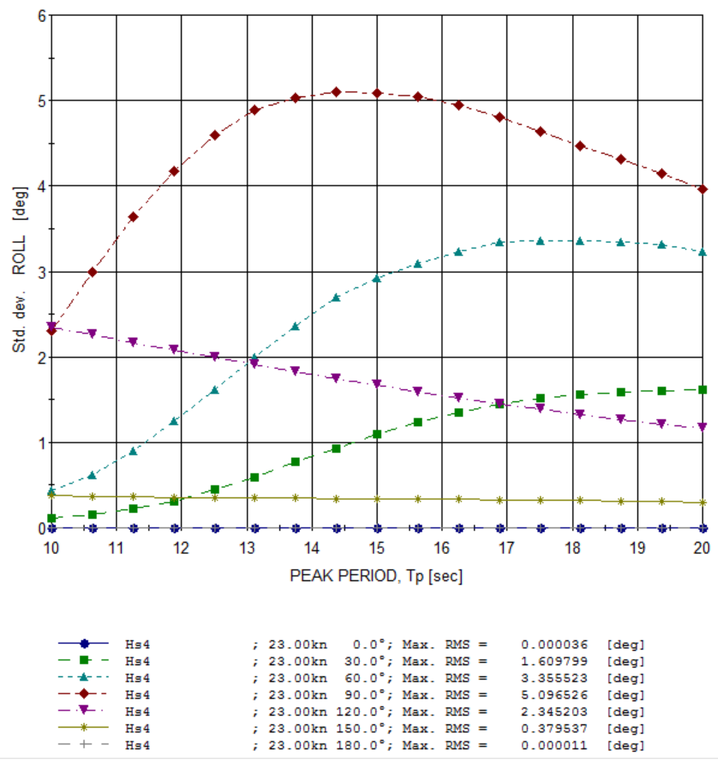
The transfer functions are calculated for V=16 knots to V=25 knots with a step of 1 knot, with T=2 s to T=27 s with a step of 1 s. The wave directions are calculated in a range of 0° to 180° with a step of 30°. In order to ensure accuracy of the response standard deviation calculations additional functions are calculated for wave periods around the most probable peak periods. For the North Sea, peak periods are normally between 7 and 13 s, therefore additional steps of 0.5 s is added in this region of the Veres calculation. The heave and pitch functions are not dependent on wave height, but the roll movement however is dependent on the wave height so several files need to be generated in order to calculate the standard deviation for roll correctly. Transfer functions for roll are calculated for H=2,4,6,8 and 10 meters.

The following figures are generated by the ShipX postprocessor and display the standard deviation for roll and relative vertical responses in irregular sea.



```
—●—   Hs4        ; 21.00kn   0.0°; Max. RMS =   0.000040  [deg]
– ■ – Hs4        ; 21.00kn  30.0°; Max. RMS =   1.820846  [deg]
-·▲·- Hs4        ; 21.00kn  60.0°; Max. RMS =   3.225003  [deg]
–◆–   Hs4        ; 21.00kn  90.0°; Max. RMS =   5.217236  [deg]
-·▼·  Hs4        ; 21.00kn 120.0°; Max. RMS =   6.729077  [deg]
—*—   Hs4        ; 21.00kn 150.0°; Max. RMS =   0.422906  [deg]
— + – Hs4        ; 21.00kn 180.0°; Max. RMS =   0.000027  [deg]
```

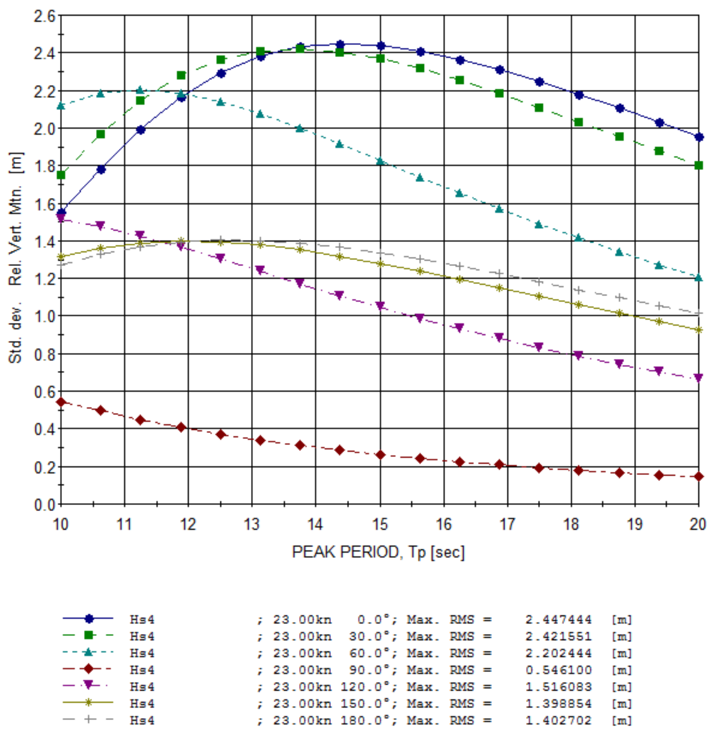
(a) V=21 knots



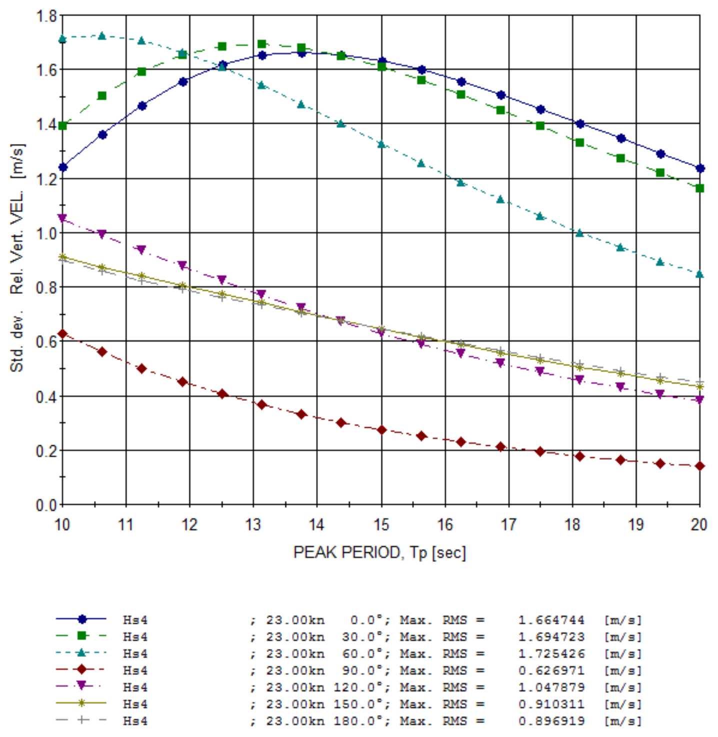
```
—●—   Hs4        ; 23.00kn   0.0°; Max. RMS =   0.000036  [deg]
– ■ – Hs4        ; 23.00kn  30.0°; Max. RMS =   1.609799  [deg]
-·▲·- Hs4        ; 23.00kn  60.0°; Max. RMS =   3.355523  [deg]
–◆–   Hs4        ; 23.00kn  90.0°; Max. RMS =   5.096526  [deg]
-·▼·  Hs4        ; 23.00kn 120.0°; Max. RMS =   2.345203  [deg]
—*—   Hs4        ; 23.00kn 150.0°; Max. RMS =   0.379537  [deg]
— + – Hs4        ; 23.00kn 180.0°; Max. RMS =   0.000011  [deg]
```

(b) V=23 knots

Figure 2.8.: RMS roll for $H_s = 4$

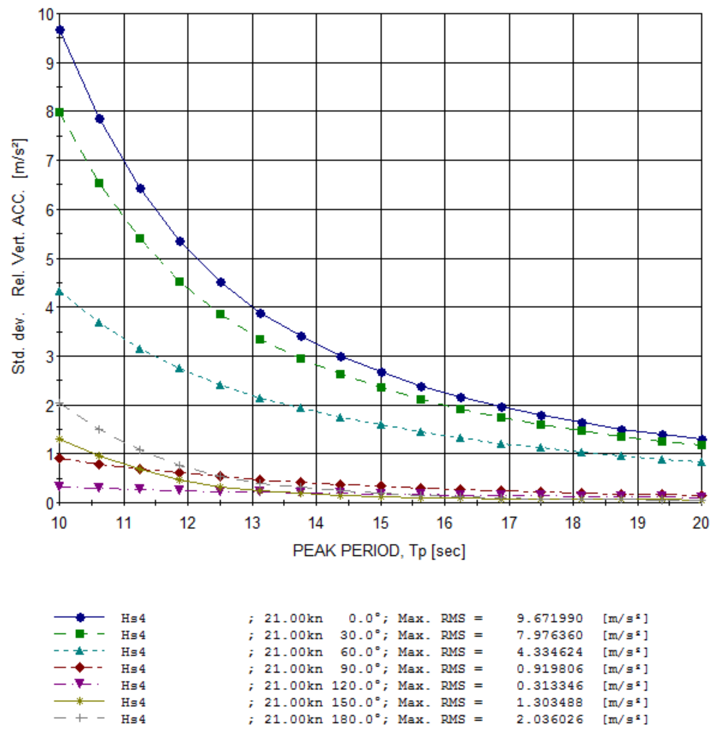


(a) Displacement



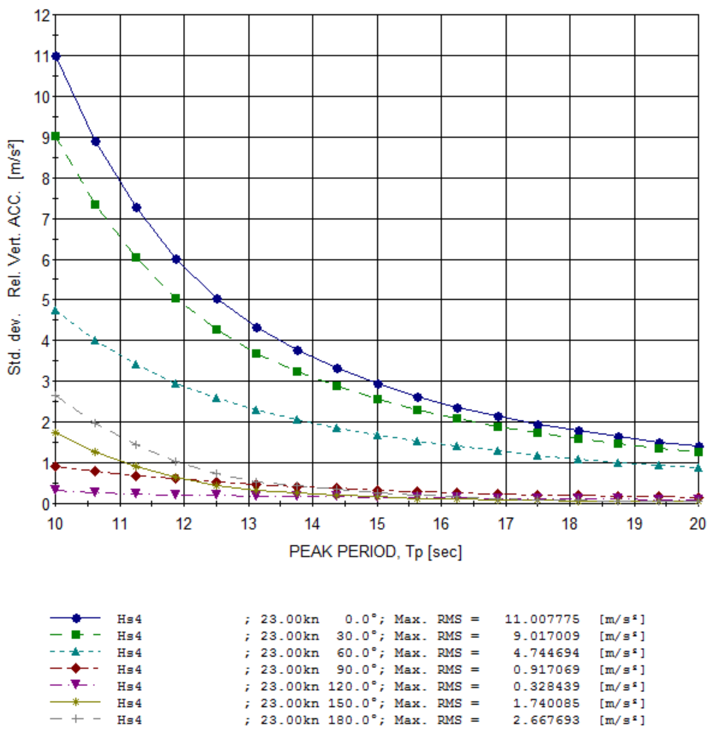
(b) Velocity

Figure 2.9.: RMS relative vertical motions $H_s = 4$



```
Hs4    ; 21.00kn   0.0°; Max. RMS =   9.671990  [m/s²]
Hs4    ; 21.00kn  30.0°; Max. RMS =   7.976360  [m/s²]
Hs4    ; 21.00kn  60.0°; Max. RMS =   4.334624  [m/s²]
Hs4    ; 21.00kn  90.0°; Max. RMS =   0.919806  [m/s²]
Hs4    ; 21.00kn 120.0°; Max. RMS =   0.313346  [m/s²]
Hs4    ; 21.00kn 150.0°; Max. RMS =   1.303488  [m/s²]
Hs4    ; 21.00kn 180.0°; Max. RMS =   2.036026  [m/s²]
```

(a) V=21 knots



```
Hs4    ; 23.00kn   0.0°; Max. RMS =  11.007775  [m/s²]
Hs4    ; 23.00kn  30.0°; Max. RMS =   9.017009  [m/s²]
Hs4    ; 23.00kn  60.0°; Max. RMS =   4.744694  [m/s²]
Hs4    ; 23.00kn  90.0°; Max. RMS =   0.917069  [m/s²]
Hs4    ; 23.00kn 120.0°; Max. RMS =   0.328439  [m/s²]
Hs4    ; 23.00kn 150.0°; Max. RMS =   1.740085  [m/s²]
Hs4    ; 23.00kn 180.0°; Max. RMS =   2.667693  [m/s²]
```

(b) V=23 knots

Figure 2.10.: RMS relative vertical acceleration $H_s = 4$

The ShipX analysis shows that there are high roll responses for beam seas. This is as expected as the forces in the y-direction are significantly higher in beam sea. It can also be observed that by reducing the speed 2 knots the roll response for $\beta = 120°$ increases to a value higher than the NORDFORSK operability criterion. This means that by reducing the speed in order to reduce the responses is not a beneficial method for roll.

In order to calculate the probability of slamming and water on deck the relative vertical transfer function is calculated. The relative vertical displacement and velocity standard deviation at FP in irregular sea is shown in fig. 2.9. Fig. 2.10 show the accelerations at FP for two different velocities.

The figures show that the motions and accelerations decrease with increasing peak period. Reducing the ship's speed will reduce the vertical responses, but the effect is not high for a reduction of only 2 knots. The accelerations exceed the NORDFORSK criterion for most of the peak periods with headings between 0 and 60 degrees. However, the relative vertical accelerations in ShipX is calculated by a different method than presented in this thesis. The relative vertical accelerations calculated in ShipX are calculated according ISO 2631/3-1985. This method is used with the objective of calculating motion sickness. The motion spectrum is divided into frequency intervals where the frequency used in the calculation is a function of the wave frequency and the exposed time, Fathi (2012). Since the method of calculating the accelerations is different in ShipX than in the rest of this thesis, the values will not be used further, but the trend can be evaluated and compared with later results.

2.5.3. Added resistance due to waves

When a ship moves in waves the resistance varies. This is due to the waves reflected by the ship, and waves generated by the ship's motion. Strip theory is used to analyze the and forces on the ship.

2.5.3.1. Added resistance in regular waves

This method was presented by Gerritsma and Beukelman (1972). It only valid for head sea waves, but Loukakis and Sclavounos (1978) generalized the method to also be valid for different wave headings.

The added resistance for head sea is calculated using the following equation.

$$R_{AW} = \frac{k}{2\omega}\int_L (B_{33}^{(2D)} + U\frac{d}{dx}A_{33}^{(2D)})V_{za}^2(x)dx \tag{2.48}$$

Where k is the wave number, ω is the wave frequency, B_{33}^{2D} and A_{33}^{2D} are the sectional damping and added mass in heave, V is the ship speed and and V_{za} is the amplitude of the motion between the ship and the waves.

The V_{za} is calculated as:

$$V_{za} = \dot{\eta}_3 - x\dot{\eta}_5 - V\eta_5 - \dot{\zeta}' \tag{2.49}$$

Where η_3 and η_5 is the displacement in heave and pitch respectively. ζ' is the effective vertical wave displacement for a cross-section. The dots indicate the time derivative.

$$\zeta' = \zeta(1 - \frac{k}{y_w}\int_{-T}^{0} ye^{kz}dz) \tag{2.50}$$

Where ζ is the wave elevation, T is the draft and y_w is the half width of the water line at a given draft. This formula neglects the wave reflection from the bow, and assumes a slender body.

To be able to calculate the resistance for waves with different headings Loukakis and Sclavounos (1978) took the wave heading into consideration:

$$R_{AW} = |R_T cos\beta| \tag{2.51}$$

Where β is the wave heading. The resistance R_T includes multiple calculations, which will not be presented here.

2.5.3.2. Added resistance in irregular waves

The mean added resistance for a given sea state is calculated using the added resistance coefficient.

$$C_{AW} = \frac{R_{AW}}{\zeta_a^2} \tag{2.52}$$

This coefficient is also called the added resistance operator, ARO. The mean added resistance due to all the encountered waves in the spectrum is:

$$\overline{R}_{AW} = 2\int_{0}^{\infty} C_{AW}S(\omega)d\omega \tag{2.53}$$

2.5.4. Added resistance due to wind

The air causes resistance to a vessel both because of the ship's velocity and the velocity of the wind. This can be separated as air and wind resistance. According to Lloyd (1998), the resistance due to air can be defined as:

$$R_{AA} = C_D \frac{1}{2} \rho_A V^2 A_T \tag{2.54}$$

Where C_D is the hull's drag coefficient above water line, ρ_A is the density of air, V is the ship's velocity and A_T is the transverse projected area.

The drag coefficient for the vessel can be found by performing a wind tunnel test of a model. Typical values for this coefficient is 0.5 to 0.8, Steen and Minsaas (2012).

By including the wind speed, V_W, the resistance for both air and wind for head wind becomes:

$$R_{AAwind} = C_D \frac{1}{2} \rho_A (V_W + V)^2 A_T \tag{2.55}$$

By extracting the wind component from equation 2.55, we find that the added resistance due to wind is:

$$R_{wind} = C_D \frac{1}{2} \rho_A (V_W^2 + 2V_W V) A_T \tag{2.56}$$

V_W is the velocity of the wind component working in the opposite direction of the vessel motion. When the vessel is exposed to wind from other directions than head wind, the drag coefficient changes.

2.5.5. Speed loss in waves and wind

The ShipX Ship Speed and Powering plug-In is mainly used to calculate the calm water resistance and propulsion. However, the plug-in can also calculate the speed loss in waves and wind. Since the model test provided information about the calm water resistance and propulsion, the speed loss calculation is of greater interest. To do this calculation a Veres calculation including the added resistance must be included.

The Matlab program should be able to operate for all oceans in all seasons. The speed loss output file from ShipX is the input file in Matlab, it defines the attainable speed for the vessel at a given sea state. It should include the probable sea states of which the vessel will be subjected. The H_s range is from 1 to 12 m with a step value of 1 m. The T_p range is from 2 to 20 s with a step value of 1 s. The wave heading is the same as for the Veres calculations.

The wind speed in the speed loss file is calculated by the significant wave height, Fathi (2012).

$$V_{wind} = \sqrt{\frac{9.81 H_s}{0.21}} \tag{2.57}$$

When the wind speed is calculated directly from the wave height, it is assumed that the wind direction is the same as the wave direction. The drag coefficient is dependent on the vessel type and heading, as shown in the figure below.

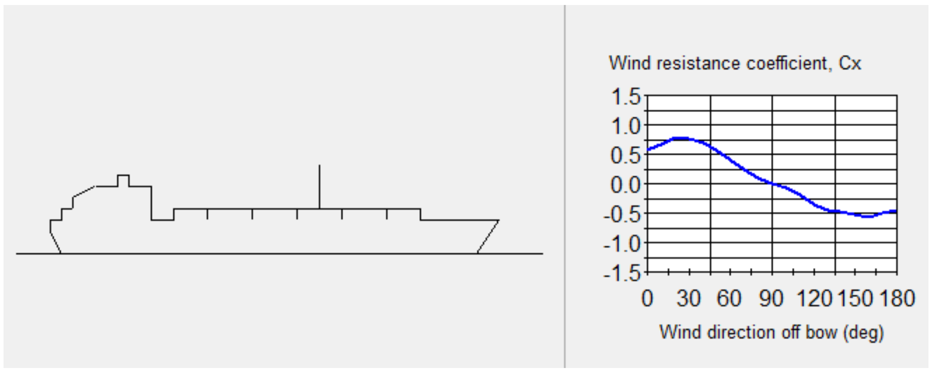


Figure 2.11.: Wind resistance coefficient vs wind direction

All information needed to calculate the speed loss is now included. The output from ShipX includes both attainable speed in waves and wind, as well as the required power for a given sea state. The attainable speed for different wave periods and headings with a significant wave height of 4 meters is shown fig. 2.12.

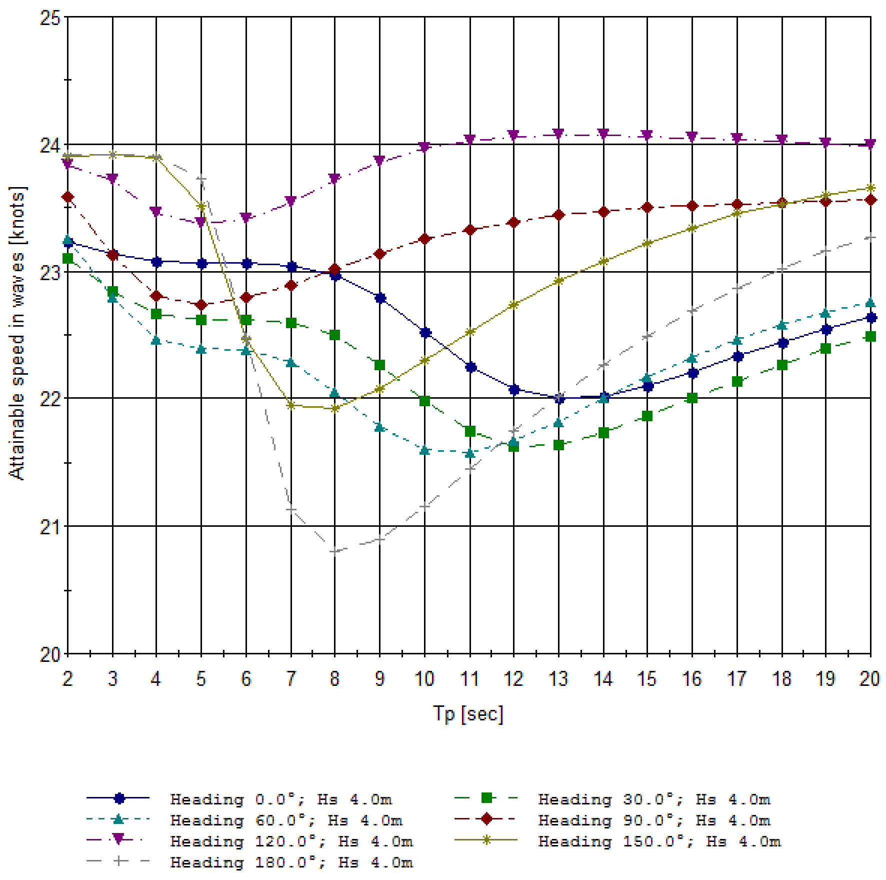


Figure 2.12.: Attainable speed in waves and wind

The figure shows that the most beneficial headings are 90° and 120°, where 0° is defined as head sea. When navigating the vessel it is desirable to obtain one of these headings for most of the usual wave periods. For $\beta = 180°$ the attainable speed for $T_P = 8$ s is very low. This is different than what would be expected, since following seas would normally give low added resistance, and therefore high attainable speed. However, when the added resistance transfer file is examined, the ARO for $\beta = 180°$ is high for all velocities over 19 knots. A physical explanation for this is not obvious, one would have go through the ARO calculations with great detail to figure this out. That however, is too time consuming, and the values for attainable speed is accepted and used further in the calculations.

V	16	17	18	19	20	21	22	23	24	25
$\beta = 180°$	-2.4	-0.9	1.6	5.3	11.7	23.4	28.5	30.5	32.5	34.0
$\beta = 0°$	1.3	1.2	1.1	1.0	1.0	0.9	0.8	0.8	0.7	0.7

Table 2.4.: ARO for T=8s

From the ShipX results the following conclusion can be made: The critical heading with regard to vertical acceleration is 0°, 30° and 60°. The critical headings for roll are 60°, 90° and 120°. While the RMS for vertical acceleration decreases slightly by reducing the speed, the values for roll increase significantly. It can be seen from fig. 2.12 that most beneficial headings with regard to the added resistance are 90° and 120°, this may again give too high responses in roll. Since there are many factors and concerns that needs to be taken into account, this is a problem where an efficient optimization tool could be very useful.

3. Network optimization and routing algorithm

In this thesis the objective is to minimize the fuel consumption for a given voyage. This is done by network optimization. A complete description of the method and underlying algorithms is presented in this chapter. As mentioned in sec. 1.5 an optimization model normally includes an objective function, one or more decision variables and a set of constrains. The objective function is defined as:

$$min\ FC(\alpha) \tag{3.1}$$

Where FC is the fuel consumption as a function of the vessel's heading α. There are several factors that influence the fuel consumption such as distance and the velocity, which is depended on the sea state. In this case the distance and sea states has been predefined and the only parameter that can be change is the route, i.e. the ship's heading. This makes α the decision variable in this optimization model. The objective function is subjected to a set of motion constrains:

Probability of slamming<Criterion slamming
Probability of deck wetness<Criterion deck wetness
RMS of roll <Criterion roll
RMS of relative vertical acceleration<Criterion acceleration

For the final route to be valid these constrains must be satisfied. When the sea is very rough it is possible that satisfying all these criteria will not be possible, and the result will be an extremely costly route, see sec. 3.2.4. This compromises the model, and limits its application. If it is not possible to find a route that satisfies the constrains, but a route suggestion is still desirable, the criteria should be adjusted.

3.1. Network optimization

In weather routing it can be beneficial to look at the problem as a network optimization problem, and apply a shortest path algorithm to find the minimal fuel or time path between the start and end point. The method is similar to dynamic programming, but it is a simpler approach. Dynamic programing is mainly used when the problem is difficult to define. In this case all information regarding the route can in this case be predefined, making a network easy to construct. In optimization a network is defined as a graph consistent of nodes and arcs. In many optimization problems the nodes will be defined as geographical locations, for example production facilities, stores or bus stops. The arcs are defined as traveled paths between the nodes. The cost may be influenced by distance, transportation method and added cost for visiting the nodes in an unfavorable order.

The graph $G = (N, B)$ includes the set of nodes $N = \{n_1, n_2, n_3, ...\}$ and arcs $B = \{b_1, b_2, b_3, ...\}$. An arc can be directed, which means it has a given start and end node, or it can be undirected and allows flow in both directions. If all arcs are directed we refer to this as a directed graph. In the routing problem it is suitable to evaluate directed graphs since the vessel will be traveling in a given direction, and therefore not allow arcs going backwards. An example of a network obtained from Lundgren et al. (2010) is given in the figure below.

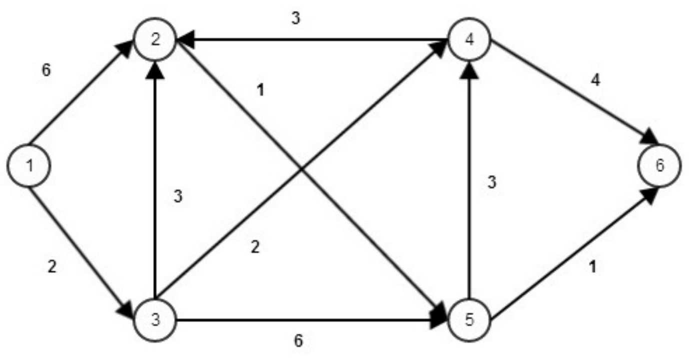


Figure 3.1.: Example of network

In this network we have six nodes and ten arcs with specified costs. The shortest path from node 1 to 6 is: 1 - 3 - 2 - 5 - 6. An algorithm used to determine this path is presented in the next section.

3.1.1. Dijkstra's algorithm

Dijkstra's algorithm was presented by Dijkstra (1959), and it is a graph search algorithm. Dijkstra's algorithm can be used to solve a shortest-path problem when the network consists of nodes and arcs with non-negative cost. This is applicable for the routing problem since there will never be a negative time or fuel consumption for traveling from one node to another. The result of this algorithm is a shortest path three, which presents the nodes visited and the total cost. In Algorithm 3.1 subscript i defines the node we are in, j defines the next node and c_{ij} is the cost of traveling from n_i to n_j.

Algorithm 3.1 Dijkstra's algorithm

Step 0: This is the initial step. The set of nodes are divided into two subsets A={searched} and D={non searched}=N, i.e. all nodes. The first node, n_s, is assigned a label (p_s, y_s), which means the previous visited node and the node prize. $(p_s, y_s) = (-, 0)$. All other nodes are given the initial node prize of ∞.
Step 1: Identify the node in subset D with the lowest cost. Choose that node as n_i.
Step 2: Search all the arcs staring from node n_i. If the cost, c_{ij}, of going to the next node plus the total cost of the traveled path of node i, y_i, is less then the current cost of node j than n_j gets a new label. $(p_j, y_j) = (i, y_i + c_{ij})$. If n_j is in subset A move node to subset D. When the node gets a new cost the arcs expanding from the nodes must be searched again.
Step 3: Move node n_i from set A to set D
Step 4: If A=N, i.e. all nodes are searched, stop. If not, go to step 1.

The algorithm used on the network shown in fig. 3.1:

Step 1: When searching from n_1 we see that n_2 gets label (1,6) and n_3 gets label (1,2). n_3 is the node with the minimal cost, and n_3 is n_i.

Step 2: Reachable nodes from n_3 is 2, 4 and 5. Total cost of going to n_4 is 4 and the node gets label gets (3,4). n_5 gets label (3,8). n_2 was labeled (1,6) in the previous step, but we see that the cost of traveling via n_3 has a lower cost, so n_2 is relabeled (3,5). None of the nodes reached are previously searched, all nodes are in subset D.

Step 3: Move node 3 from D to A.

Step 4: Go to step 1 and choose the minimal cost node: n_4

The figure below shows the network after the first search.

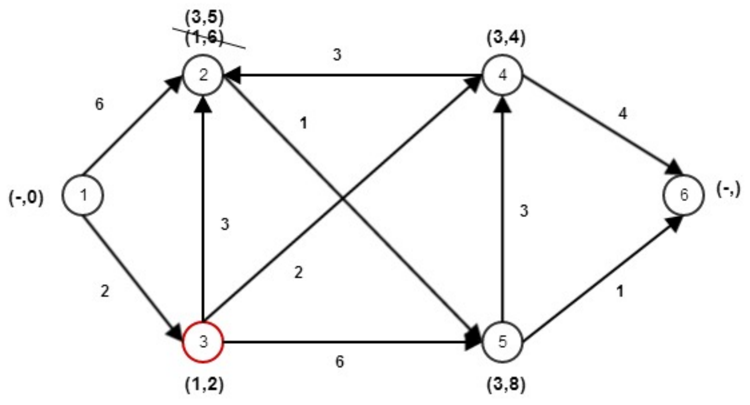


Figure 3.2.: First search on network

In the fourth search we can see that n_5 gets a new label. In the second search n_5 was labeled (3,8), but when evaluation the situation from n_2 we can see that a cheaper path to n_5 is via n_2, and the new label is (2,6). This procedure continues until all nodes are searched, and the optimal path to the end node is found. The final setup, and shortest path for the network is presented in fig. 3.3.

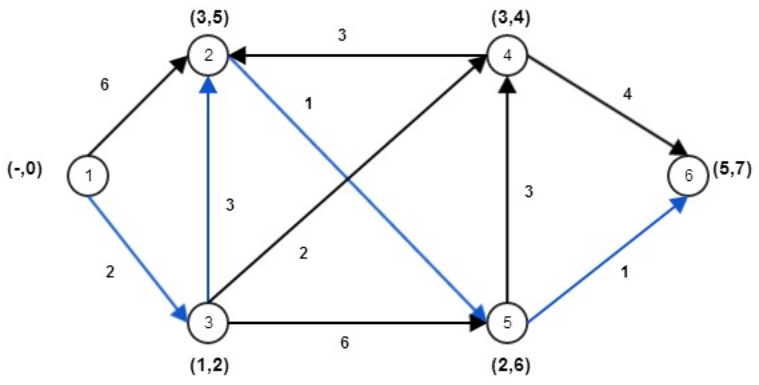


Figure 3.3.: Shortest path

3.2. Routing algorithm

In this thesis, the nodes are defined as geographical coordinates, the arcs are the possible traveled paths between the nodes and the cost of the arcs are defined as the fuel consumption. The main flow of the routing algorithm is:

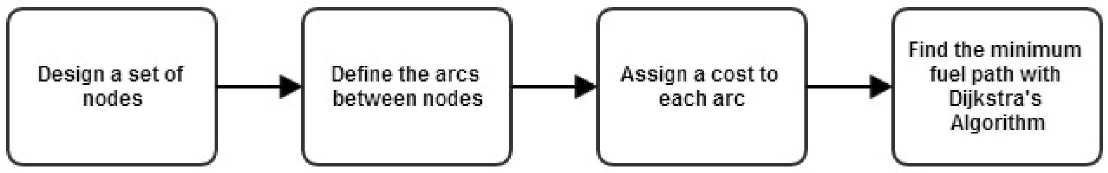

Design a set of nodes
Define the arcs between nodes
Assign a cost to each arc
Find the minimum fuel path with Dijkstra's Algorithm

Figure 3.4.: Routing algorithm

Each of these steps includes further calculations, which will be presented in detail in this section.

3.2.1. Navigation

When defining a routing procedure one needs be familiar with a few terms in navigation. When navigating on a sphere the master can chose to navigate by the rhumb line or the great circle, Bowditch (2002). A rhumb line, also called a loxodrome, is presented on a map as a straight line. If the earth was flat this line would be the shortest path between two points. By mathematical definition *a rhumb line or curve on the surface of a sphere intersecting all meridians at the same angle,* dictionary.com (2014). Benefits with using the rhumb line for navigation is that the master can keep a constant heading. The great circle is composed by orthodromes. An orthodrome is defined as the shortest line between two points on a sphere. For calm water the great circle route, GCR, will therefore be the minimum fuel route. However, when using othodromes for navigation the master will constantly need to change the heading of the vessel. A good way of navigating can be to use a combination of the rhumb line and great circle. By choosing a few point on the great circle route, the master can use the rhumb line between these points and only change the heading a few times during the voyage.

The azimuth is defined as the angel between north and the heading of the vessel. True north has an azimuth of 0° and the values increase clockwise as shown infig. 3.5. The azimuth angle is important in order to find the relative heading between the vessel and wave and wind direction. This is described in sec. 3.2.4.

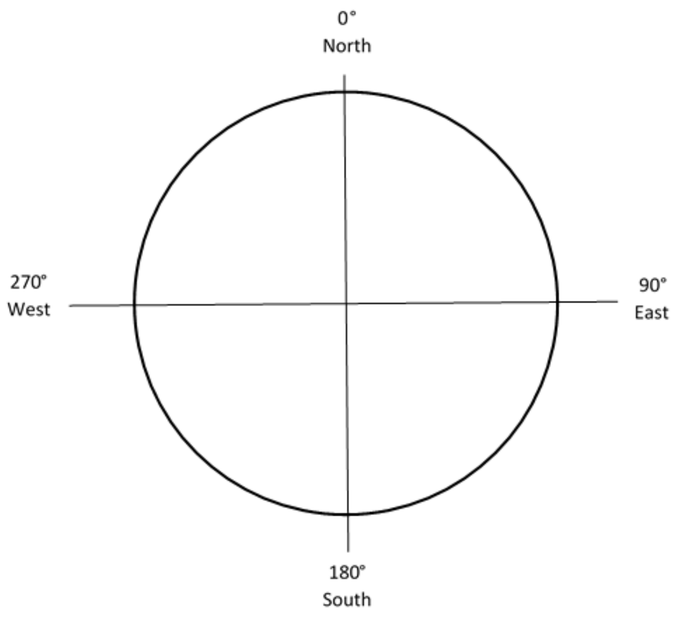


Figure 3.5.: Azimuth angle

3.2.2. Node design

In order to perform the optimization a grid with nodes must be defined. In this thesis the grid design is inspired by a method presented by Lee et al. (2002). The feasible region for the vessel will be defined around the great circle route. The following steps are performed in order to define the set of nodes:

Algorithm 3.2 Node design

```
Step 1: Find the center point along the great circle route, $X_C$
Step 2: Calculate the course, $\alpha$ between the center point and the point before the center point, $X_{C-1}$, on the GCR.
Step 3: Define a perpendicular to $\alpha$
Step 4: Draw points, $C \pm i$, along the perpendicular with a chosen distance, $\delta_i$, between the points.
Step 5: Calculate the rhumb lines between the start- and endpoint and each of the center points
Step 6: Define a set of points along each rhumb line.
```

The figure below defines the set of center points.

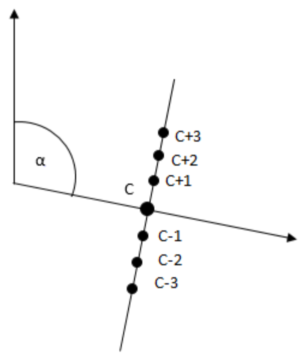


Figure 3.6.: Defining center points

An example of the grid is given in fig. 3.7.

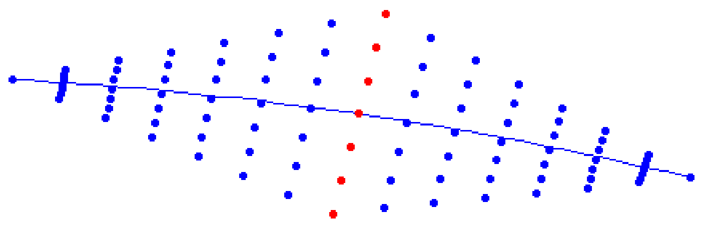


Figure 3.7.: Node design, example of grid

The line represents the great circle route, the red points is the center points and the blue points connects the end points to the center points. The chosen distance between the center points in 1°, and the points along each rhumb line is also spaced out with a distance of 1°. This corresponds to approximately 60 nautical miles. For the vessel used in this optimization the speed will normally be between 20 and 23 knots, depending on the sea state, which means that the ship master will possibly change the heading every third hour.

3.2.3. Defining arcs

In order to navigate from the departure node to the arrival node arcs need to be assigned between them. Assigning arcs between nodes gives the possibility of limiting the problem by saying that only a limited number of nodes can be reached from the previews node. An example of nodes and arcs is given in the figure below. In this example we have five center points and seven stages. A stage is defined a row of nodes which has the same time step.

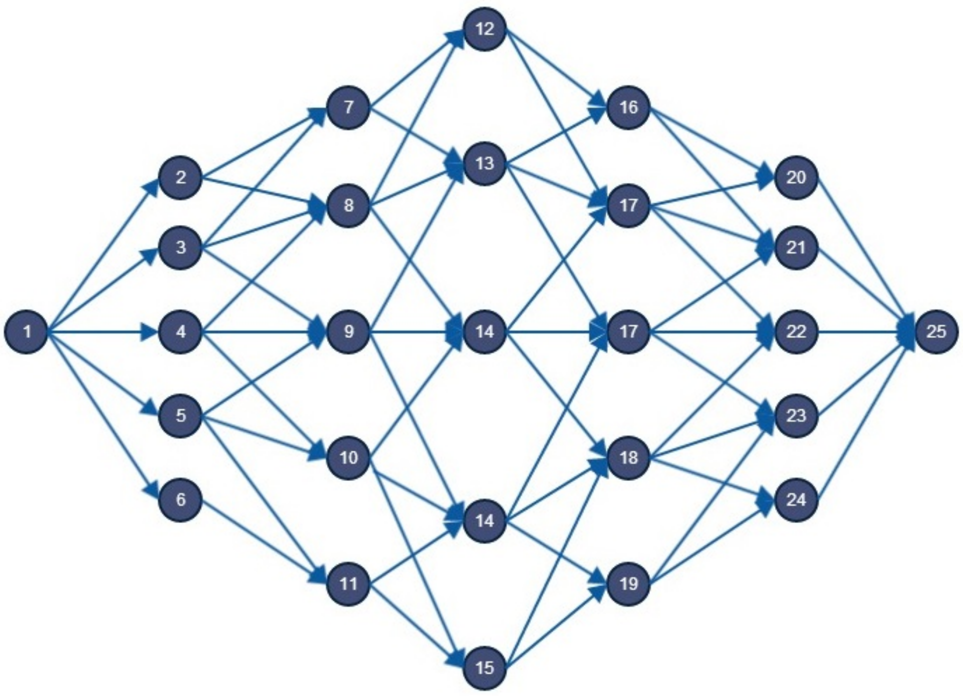


Figure 3.8.: Network for routing

In this thesis the arcs are divided into straight arcs, right arcs and left arcs. This means that for a node inside the network, i.e. excluding the start and end node, it has possibility of going to two or three different nodes. From node 4 nodes 8, 9 and 10 can be reached. When evaluating the figure above it can seem more appropriate to define them as north, south or straight arcs since we are discussing actual geographical coordinates. However, the network can be set up for voyages going east, west, north and south so the actual directions will vary for each network. To evaluate the right and left arcs the graph must be evaluated as a graph where the departure node is at the top, and the arrival node is at the bottom of the graph.

The network is set up like this to limit the problem. When the node only can reach two or three other nodes the computation time is limited. It can also be assumed that going to a more distant node would not be an optimal solution since the geographical distance is large.

3.2.4. Assigning the cost to each arc

The cost of each arc is defined as the fuel consumption of going from one node to the next. The fuel consumption for a given arc can be found from:

$$FC = bsfc \cdot P_B \cdot t \tag{3.2}$$

Where bsfc is the brake specific fuel consumption in [g/kWh], P_B is the brake power and t is the time in hours it takes to sail between two nodes. The brake power is kept constant. The bsfc is also a constant, so the only unknown in the equation is t. The length of the arc and the attainable speed of the vessel must be known in order to find the time.

$$t = \frac{d}{V} \tag{3.3}$$

Where d is the great circle length of the arc, according to Bowditch (2002) d is defined as:

$$d = cos^{-1}(sin\phi_1 sin\phi_2 + cos\phi_1 cos\phi_2 \Delta\lambda) \tag{3.4}$$

Where ϕ_1 and ϕ_2 are the latitude points of departure and arrival and $\Delta\lambda$ is the difference in longitude between the two points. Even though the nodes in the network is defined along the rhumb lines, the distance the ship will actually sail is the orthodrome distance. Therefore, when calculating the time it takes to travel it is important to consider the great circle. It should also be noticed that when the brake power i kept constant and the only parameter changing is time, so this method can also be seen as minimal time routing.

In order to find the attainable speed for the vessel we need to know the total resistance from calm water, wind and waves. The total resistance can be found by knowing the significant wave height, the mean wave period and direction, the wind velocity and direction and the vessel speed. Finding the speed is an iteration process.

The velocity is guessed until the total resistance, velocity and propulsion efficiencies gives the desired brake power.

The wind velocity is given as U- and V-components of the wind at 10 meters above the free surface. The U-component gives the wind velocity in the x-direction, i.e. east, and the V-component is velocity in the y-direction, i.e. north.

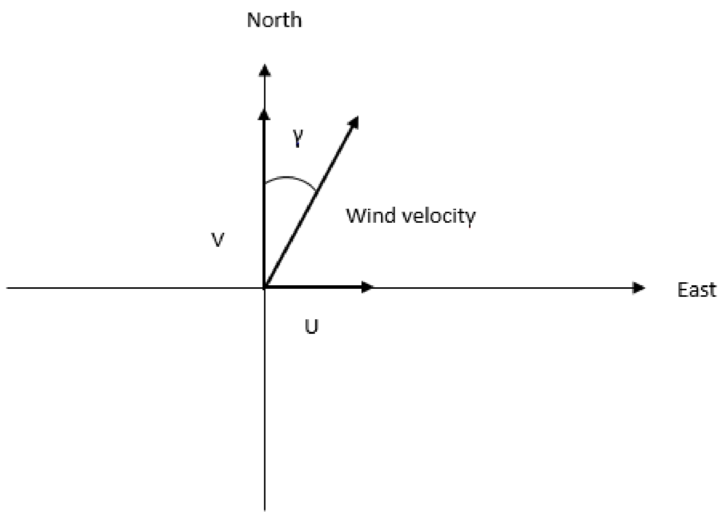


Figure 3.9.: Wind profile

The resulting wind velocity and angle is:

$$V_w = \sqrt{U^2 + V^2} \tag{3.5}$$

$$\gamma = tan^{-1}(\frac{U}{V}) \tag{3.6}$$

However, to find the relative heading between the vessel and wind the angle needs to be defined as an azimuth, blowing in a direction between 0° and 360°. This is done simply by adding 90°, 180° and 270° to γ dependent on whether U and V are positive or negative. The wind is blowing west when U has a negative value, and south when V has a negative value. The next step in finding the the total resistance is to calculate the relative angel between the vessel and wind. The azimuth angel for wind is defined as γ, the vessel's heading is defined as α and the relative angel between vessel and both wind and waves is defined as β.

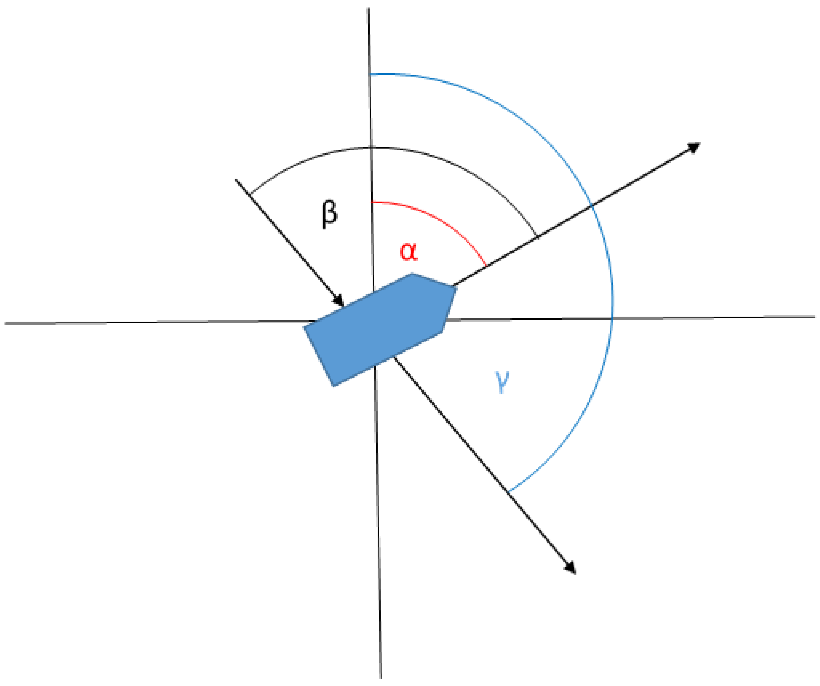


Figure 3.10.: Wind direction - β_{wind}

As shown in figure fig. 3.10, β meets the ship with $\gamma - 180°$. Since the sway force, or the yaw moment is of no interest, β_{wind} is considered to be between 0° and 180°. The way of calculating the relative wind heading is shown in the equation below.

$$\beta_{wind} = \begin{cases} |\gamma - \alpha - 180| & if \ -180 < \gamma - \alpha < 360 \\ |\gamma - \alpha + 180| & if \ \gamma - \alpha < -180 \end{cases} \tag{3.7}$$

The wind relative heading corresponds to a drag coefficient, and the added resistance due wind can be calculated using equation 2.56.

The mean wave heading in the weather forecast is defined as where the wave is coming from. This means the wave meets the vessel as in the figure below.

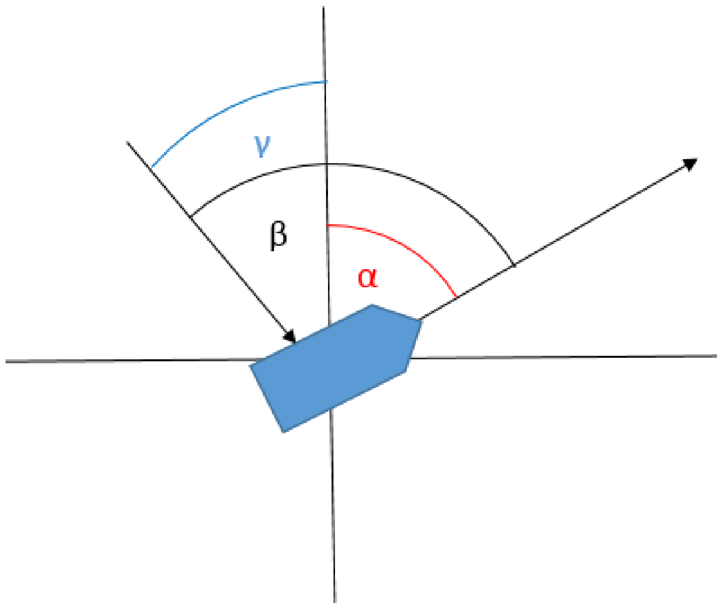


Figure 3.11.: Wave direction - β_{wave}

β_{wave} can be calculated as:

$$\beta_{wave} = \sqrt{\alpha^2 + \gamma^2 - 2\alpha\gamma} \tag{3.8}$$

With knowledge of the relative wave heading, significant wave height and mean period, the mean added resistance for the arc can be calculated using the method described in chapter 2. When all of this is performed the velocity for the given sea state can be found, and each arc assigned a cost.

In addition to the added resistance, this thesis proposes a method to avoid sea states that lead to large responses of the vessel. The seekeeping calculations are performed with regard to roll, relative vertical acceleration, deck wetness and slamming for each arc. If the arc exceeds any of the seakeeping criteria, the cost for that given arc is set to 1000 ton of fuel. This means that this arc is to expensive to travel by, and it will not be chosen as the shortest path by the optimization algorithm.

4. Matlab program

4.1. Program setup

The optimization program is written in Matlab. In order to use the program the mapping toolbox needs to be installed in order to define the geographical coordinates. The program is general, and is able to calculate the optimal route for different vessels and voyages. The program is set up so that all of the input needed is placed in the first part of the main script. The second part of the main script includes several functions that performs the calculations.

4.1.1. Input requirements

The vessel presented in sec. 2.1 is included in the program. In order to run the program the departure and arrival points must be defined, and a GRIB file describing the weather must be specified. A GRIB file (GRIdded Binary) is a binary file that is used to store forecast or historical weather data. If wind velocities are not included in the GRIB file, the wind velocity can be estimated from the significant wave height using equation 2.57. However, the estimated wind velocity might be too high, since the significant wave height from the weather file is normally a combination of wind and swell, and the equation for wind velocity assumes H_s from wind only. Four criteria with regard to seekeping are defined in compliance with NORDFORSK. The probability of slamming, deck wetness, and standard deviation of roll and vertical accelerations can be specified.

If the user wishes to calculate the route for another vessel, the input files must be changed. The input files that define the vessel are obtained from ShipX. Examples of all the input files can be found in Appendix B. To calculate the responses of the vessel several transfer functions files are specified. These files are text files, and is the .re1 files found in the Veres run. One file is defined to calculate the heave and pitch transfer function. The roll transfer function is dependent on the wave height, several input files for different wave height are included. Two input files are defined for calculating speedloss. These files are called seaway_shipx in the run folder. The SEAWAY_RES file is used when the the wind speed is calculated from the wave height, and the attainable speed can be read directly from the ShipX results. This file also includes information about for how many headings, peak periods and significant waves heights the calculation has been performed. The OUT files are used when the wind resistance is calculated in the Matlab program. It includes information about the increase in resistance due to waves as a percentage of the calm water resistance. Both speed loss files contain information at the top of the file that is not used. Therefore the files needs to be slightly edited before they can be used in the program, see example in sec. B.2.

A few more input files must be specific in order to run the program. These are not obtained from ShipX, but are only necessary if the wind speed should be calculated directly from the GRIB file. The user specified input includes a calm water resistance text file, a wind resistance coefficient text file and a open water text file. Parameters should also be defined with regard to resistance, propulsion and seakeeping. As for the input files, some of these parameters are only necessary when the wind is estimated. The list of parameters is presented in the table below.

Name	Definition	Necessary in estimated wind?
Pb	Desired brake power	YES
bsfc	Brake specific fuel consumption	YES
Xp	x-position of FP	YES
Freeboard	Freeboard	YES
A_T	Transverse projected area	NO
D	Propeller diameter	NO
Eta_M	Mechanical efficiency	NO
Eta_R	Rotational efficiency	NO
C_wake	Wake coefficient	NO
C_t	Thrust deduction coefficient	NO

Table 4.1.: User specified ship parameters

4.1.2. Description of program

The program includes one main file and several underlying functions. The functions are divided into read and calculation functions. The read functions are placed first in the script, they read the input files and give out matrices with regard to resistance, speed, weather and RAOs. There are three different functions calculating fuel consumption. The user can calculate the fuel consumption for the great circle route, if a comparison between the optimal fuel and the GCR fuel is requested. Two functions calculates the optimal fuel path, one for estimated wind, and one for actual. All the functions in the program are presented shortly in tab. 4.2.

Function name	Description	Read or calculation	Necessary in estimated wind?
speedloss	Gives out a matrix including attainable speed for a given sea state	Read	YES
increaseaddedres	Gives out matrices with added resistance due to wave, wind resistance coefficients, calm water resistance and open water data.	Read	NO
readgrib	Gives out matrices with waves and wind values	Read	YES
readinput	Gives out matrices with transfer function	Read	YES
gcfuel	Calculates the fuel consumption for the great circle route	Calculation	Optional
optfuel	Finds the optimal route, and calculates the fuel consumption for estimated wind	Calculation	YES
optfuel2	Finds the optimal route, and calculates the fuel consumption for forecasted wind	Calculation	NO
seekeeping	Calculates the responses of the vessel, and defines whether the seakeeping criteria are exceeded. Function included in optfuel and optfuel2	Calculation	YES
calculatespeed	Calculates the attaineble speed with forecasted wind. Function included in optfuel2	Calculation	NO

Table 4.2.: Functions in Matlab program

Fig. 4.1 is a flow chart of the program. It describes the flow of variables, and the order of the program. The great circle route fuel calculation is not included in the flow chart, since this function is optimal an not necessary in order to find the optimal route. The red line indicate flow of input and output.

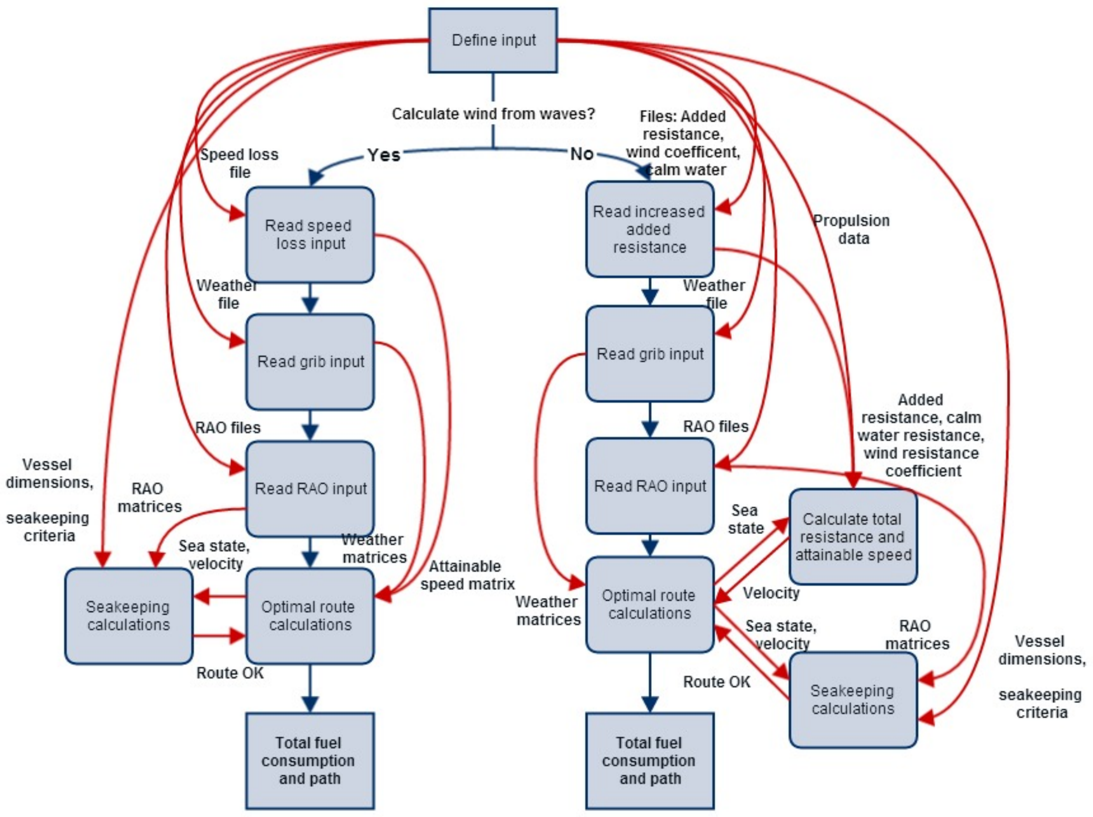

Define input
Calculate wind from waves?
Yes
No
Speed loss file
Files: Added resistance, wind coefficent, calm water
Read speed loss input
Read increased added resistance
Propulsion data
Weather file
Read grib input
RAO files
Added resistance, calm water resistance, wind resistance coefficient
Read RAO input
Vessel dimensions, seakeeping criteria
RAO matrices
Sea state, velocity
Weather matrices
Attainable speed matrix
Sea state
Calculate total resistance and attainable speed
Velocity
Seakeeping calculations
Optimal route calculations
Route OK
Total fuel consumption and path

Figure 4.1.: Flow chart of program

4.2. Function description and verification

The optimal fuel function is the final function in the script, all the previous calculations are input to this function. The output is the optimal route, and the total fuel consumption. As mentioned, there are two optimal fuel function for respectively estimated and forecasted wind. The functions are almost identical, but with one difference. When the wind is estimated the function reads the attainable speed from the ShipX speedloss file, otherwise the calculate speed function is used to find the attainable speed. The seakeeping function is an underlying function for both optimal fuel functions. In order to trust the final output it must be known that the underlying functions performs correct calculations. This section will describe the underlying calculation functions, while the final results are presented and discussed in chapter 5.

4.2.1. Seakeeping function

The seakeeping function calculates whether or not the arc satisfies the seakeeping criteria. The output of this function is a *routeOK* variable, which is equal to 1 if the traveled arc does not exceed the criteria for roll, vertical acceleration, slamming and green water on deck. The input for this function is the sea state and the transfer function matrices. The function chooses the RAOs that correspond to the given vessel speed and wave heading, and then uses the wave spectrum to calculate the seakeeping performance of the vessel.

Several checks were conducted in order to ensure that the functions read the right RAOs, and that the calculations are performed correctly. The relative vertical response were checked by calculations in a spreadsheet, the results are equal to the Matlab results. To make sure that the final values are calculated correctly, sample calculations has been performed for different wave headings and significant wave height and compared to ShipX results. The seakeeping function has been run for a ship speed of 22 knots, and a peak period of 13 s. The results are presented in tables for each of the criteria. The red values in the tables indicates that the seakeeping criteria defined by NORFORSK are exceeded. The plots from ShipX defines the operabilty region for the vessel. The wave heights above the lines of the following figures are calculated too be to high.

RMS values for roll. V=22 knots and T_p=13 s

$H_s \backslash \beta$	0	30	60	90	120	150	180
2	0	0.13	0.63	1.91	0.42	0.09	0
4	0	0.27	1.24	3.72	0.81	0.17	0
6	0	0.42	1.83	5.45	1.17	0.26	0
8	0	0.56	2.40	7.10	1.51	0.35	0
10	0	0.70	2.97	8.69	1.83	0.43	0

Table 4.3.: RMS roll for different headings and significant wave heights

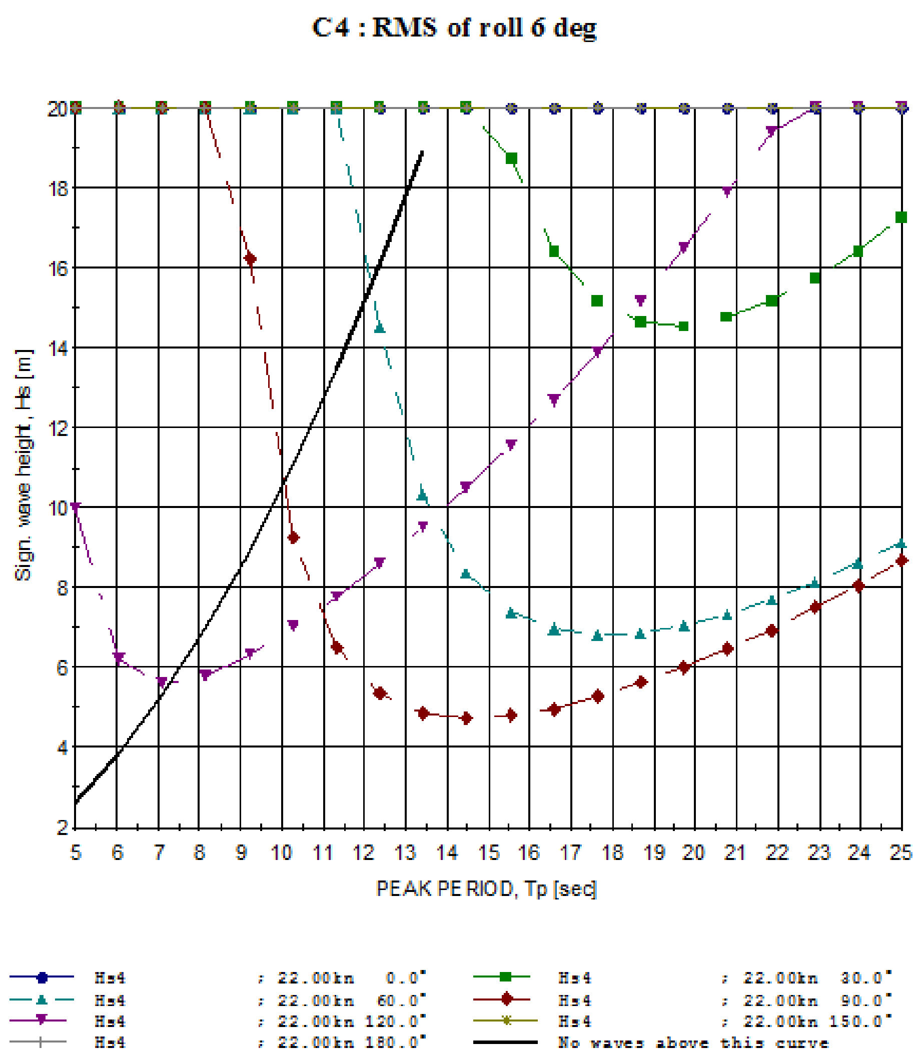


Figure 4.2.: ShipX operability results for roll

We can see from the graph and table that the seakeeping function underestimates the standard deviation for roll. The values of the σ_{roll} are highest when $\beta = 90°$, this corresponds well with the results from ShipX.

Probability of slamming. V=22 knots and T_p=13 s

$H_s \backslash \beta$	0	30	60	90	120	150	180
4	0%	0%	0%	0%	0%	0%	0%
6	0%	0%	0%	0%	0%	0%	0%
8	1%	1%	1%	0%	0%	0%	0%
10	5%	6%	3%	0%	0%	0%	0%

Table 4.4.: Probability of slamming for different headings and significant wave heights

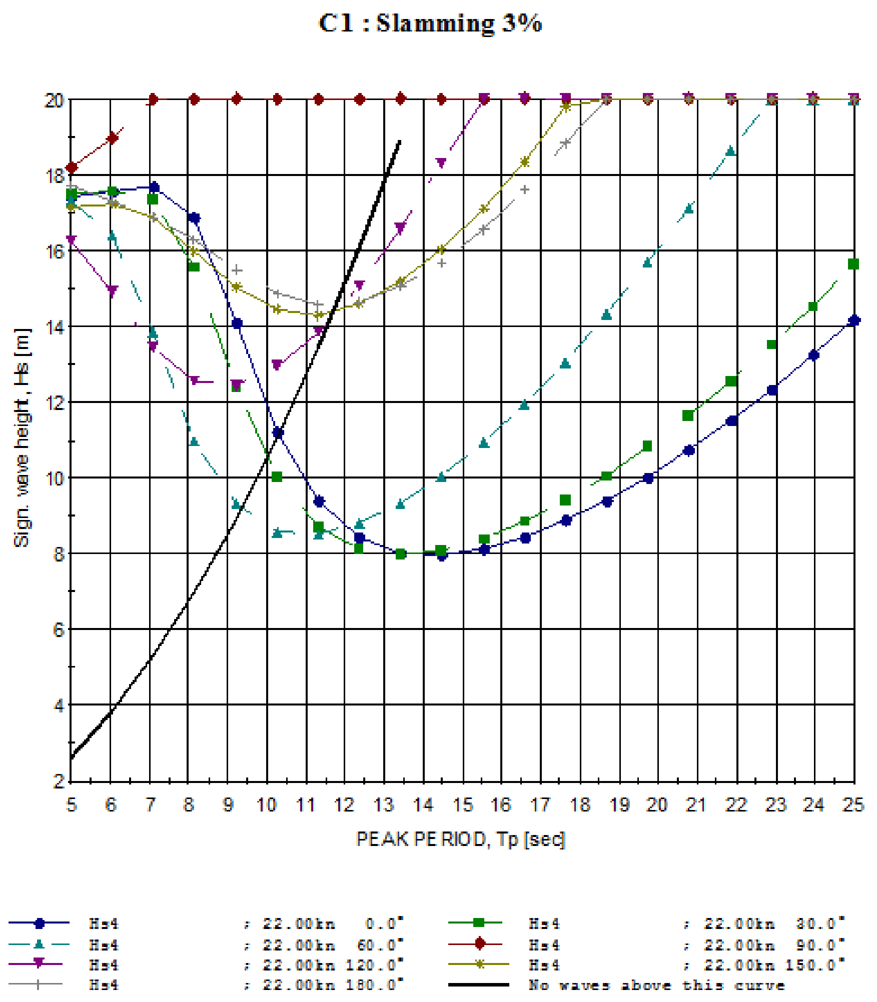


Figure 4.3.: ShipX operability results for slamming

Probability of deck wetness. V=22 knots and T_p=13 s

$H_s \backslash \beta$	0	30	60	90	120	150	180
4	0%	0%	0%	0%	0%	0%	0%
6	0%	0%	0%	0%	0%	0%	0%
8	0%	0%	0%	0%	0%	0%	0%
10	3%	3%	1%	0%	0%	0%	0%

Table 4.5.: Probability of deck wetness for different headings and significant wave heights

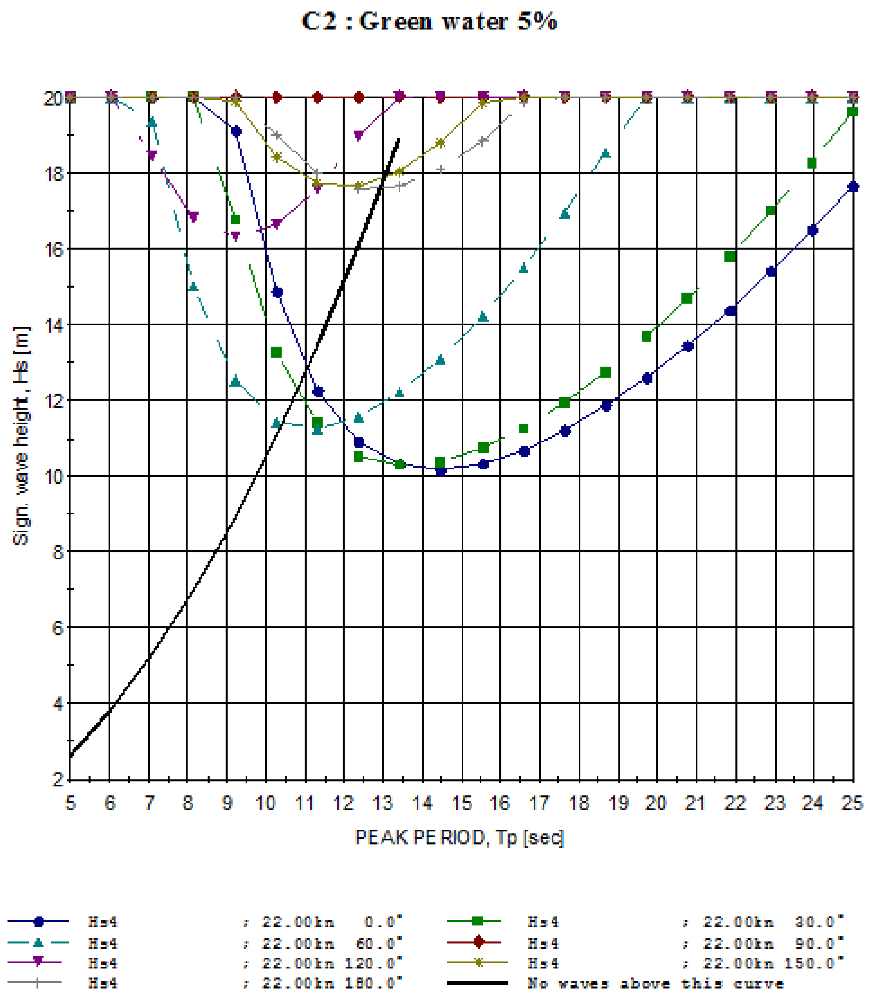


Figure 4.4.: ShipX operability results for deck wetness

The values for slamming and deck wetness are also underestimated. Slamming and deck wetness will only occur for head sea when the significant wave height is over 10 m. Table 4.6 shows the values for relative vertical acceleration, and this limits the operabilty region much more than roll, slamming and deck wetness.

$H_s\backslash\beta$	0	30	60	90	120	150	180
2	0.42	0.44	0.47	0.28	0.39	0.37	0.36
4	0.83	0.87	0.94	0.57	0.78	0.75	0.72
6	1.26	1.31	1.42	0.85	1.17	1.18	1.08
8	1.68	1.75	1.89	1.37	1.56	1.5	1.44
10	2.10	2.18	2.36	1.42	1.96	1.86	1.80

Table 4.6.: RMS relative vertical acceleration for different headings and significant wave heights

The relative vertical accelerations calculated in ShipX are calculated according ISO 2631/3-1985. The method is quite different than the method used in the Matlab program, therefore comparing the results from the seakeeping function with the ShipX results will not give any verification. The acceleration should be compared to the NORFORSK criteria.

The table above shows that the ship never exceeds the criterion for L<100 meters. If 0.05 g is set as criterion the vessel would not be able to operate in waves higher than 2 m. The vessel used in this thesis is 233 meters long. If a linear relationship can be assumed between length and acceleration criterion, a 233 m long container vessel should not exceed a RMS value for acceleration of $0.14g = 1.37m/s^2$. If that is used as a criterion for vertical acceleration, the vessel can operate in a sea state with $H_s < 6$ m for all headings, and $H_s < 8$ m for all other headings then 60°.

The vertical acceleration is highly dependent on the wave period. The figures 4.5. and 4.6. show the vertical acceleration for two different wave heights an three different headings relative to mean wave period.

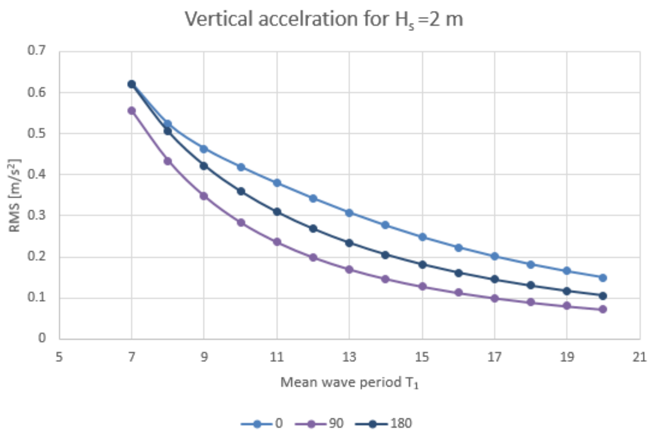


Figure 4.5.: RMS vertical acceleration for $H_s = 2$

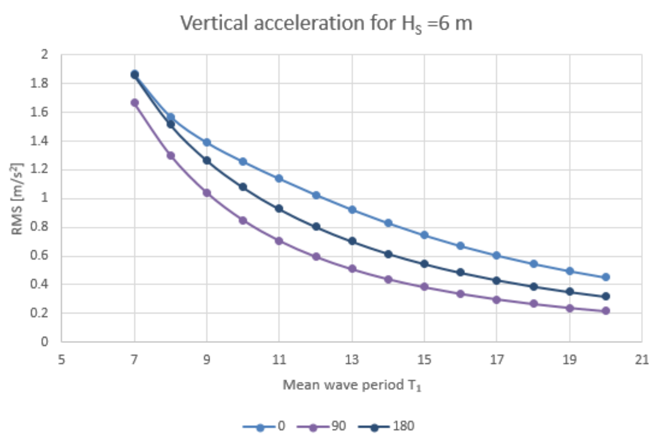


Figure 4.6.: RMS vertical acceleration for $H_s = 6$

The figures show that the accelerations decrease with increasing wave period. This trend corresponds well with the results from ShipX presented in sec. 2.5.2, even though the values are significantly lower. When the significant wave height is 2 meters the values for acceleration will not exceed the criterion of 0.14 g. However, when the significant wave height is 6 meters, the mean wave period needs to be over 10 s if the vessel should be able to operate, this corresponds to a peak period of 13 s.

When evaluating the results and compering them to ShipX it can be concluded that the values for roll, slamming and deck wetness are slightly underestimated, but the trend is correct. In theory the results should have been exactly the same since the method of calculating this is the same. However, differences can be caused by different integration methods and interpolation. In Matlab the responses are not interpolated, but integrated directly by using the trapeze method. The underlying calculations in ShipX are not known, but it is reasonable to assume that some sort of interpolation is used. To compensate for this underestimation the criteria can be defined lower. The results also show that for all other headings than $\beta = 90°$ the vertical acceleration is the limiting seakeeping criterion. This also corresponds well with what would be expected.

4.2.2. Calculate speed function

The calculate speed function is an underlying function in the optimal fuel function when the wind resistance is calculated. The sea state is defined in the optimal fuel function and the objective is to calculate the attainable speed for a given sea state. The process is described in fig. 4.7.

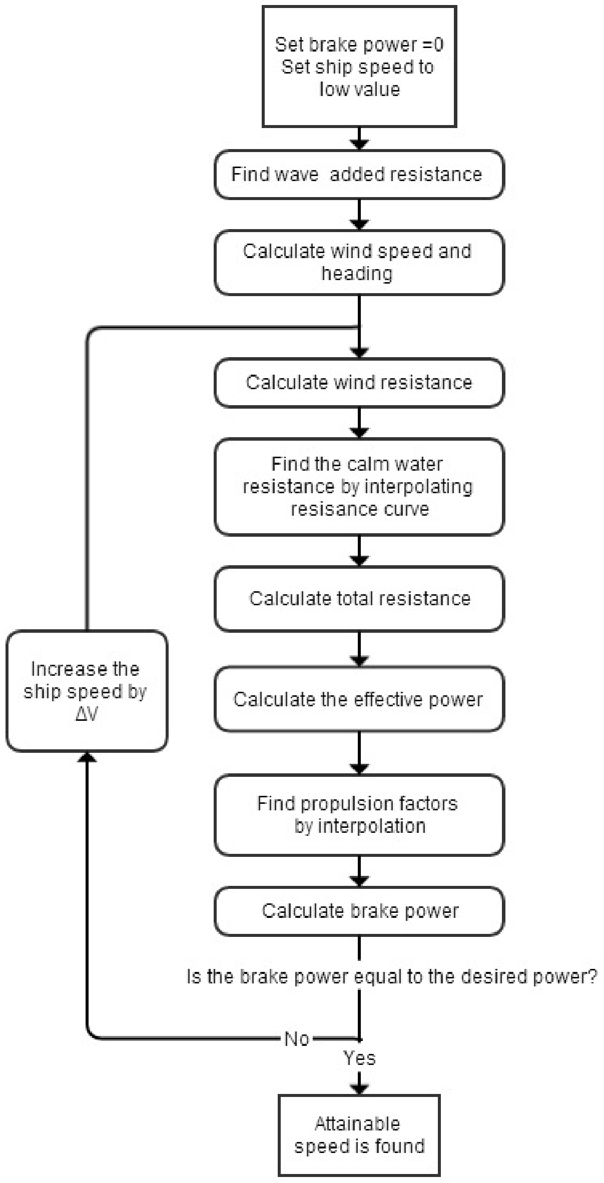

Set brake power =0
Set ship speed to low value
Find wave added resistance
Calculate wind speed and heading
Calculate wind resistance
Find the calm water resistance by interpolating resisance curve
Calculate total resistance
Calculate the effective power
Find propulsion factors by interpolation
Calculate brake power
Is the brake power equal to the desired power?
No
Yes
Increase the ship speed by ΔV
Attainable speed is found

Figure 4.7.: Speed calculation i Matlab program.

The calculate speed function is verified by comparing the calculated speed with the attainable speed from ShipX for different sea states and headings. The wind speed and heading is calculated in the same manner as in ShipX for the calculations to be comparable. The velocities are presented in tab. 4.7, where M denotes Matlab, S denotes ShipX and D is the percentage increase between Matlab and ShipX.

		$\beta = 0°$			$\beta = 90°$			$\beta = 180°$		
H_s	T_p	M	S	D	M	S	D	M	S	D
2	10	23.48	23.28	0.9%	23.76	23.55	0.9%	23.10	22.95	0.6%
	14	23.36	23.15	0.9%	23.82	23.60	0.9%	23.48	23.41	0.3%
4	10	22.76	22.53	1.0%	23.50	23.25	1.1%	21.82	21.16	3%
	14	22.58	22.02	2.5%	23.72	23.47	1.1%	22.56	22.25	1.3%
6	10	22.18	21.28	4.0%	23.10	22.77	1.4%	21.18	19.84	6.3%
	14	21.54	20.15	6.5%	23.58	23.27	1.3%	21.82	21.17	3.0%
8	10	21.16	19.22	9.2%	22.58	22.10	2.1%	20.76	19.12	7.9%
	14	20.46	17.54	14.27%	23.36	23.05	1.3%	21.36	20.34	4.8%

Table 4.7.: Speed calculation in Matlab vs ShipX

The table shows that the speed calculated in Matlab is consequently higher than the speed from ShipX. This is important to remember when evaluation the final results. The difference is significant in rough sea for head and following waves.

As for the responses the small differences can be caused by different calculation methods. The different parameters in Matlab, like calm water resistance and propulsion factors are found by using a cubic spline interpolation, which means that the resulting line used to find the final value passes through the known points. ShipX might use another type of interpolation, for example B-spline interpolation, which means that the line does not necessarily pass trough the points, but close to them. Another explanation is needed when the velocity has a higher deviation, as for rough sea. These differences can be explained by thrust loss. The ShipX calculated speeds accounts for both thrust and torque loss due to partly immersed propeller, but the Matlab calculated speed does not.

However, according to Faltinsen (1990) the probability of the waves being six meters and higher in the North Atlantic is 7.2 %, so for most sea states the calculations are fund reliable. When evaluating the function, it was found that for $H_s = 10$ m and $\beta = 120°$ the velocity was too high for the program to work properly. A break was included, and the velocity set to 25 knots. The probability of occurrence of the sea state is less and 0.05 %, so this will not compromise the functionality of the program.

4.2.3. Calculate fuel functions

The great circle route fuel consumption is calculated with estimated wind, so when evaluating the fuel savings optimal fuel consumption should also be calculated with estimated wind. In this thesis the great circle route is defined as a practical GCR. This means that instead of the route being a circle or a curve, it is constructed with three straight legs by defining two points on the GCR. The advantage of this is that instead of continuously adapting the heading, the ship master only needs to change the heading twice. The figure below show the practical great circle route.

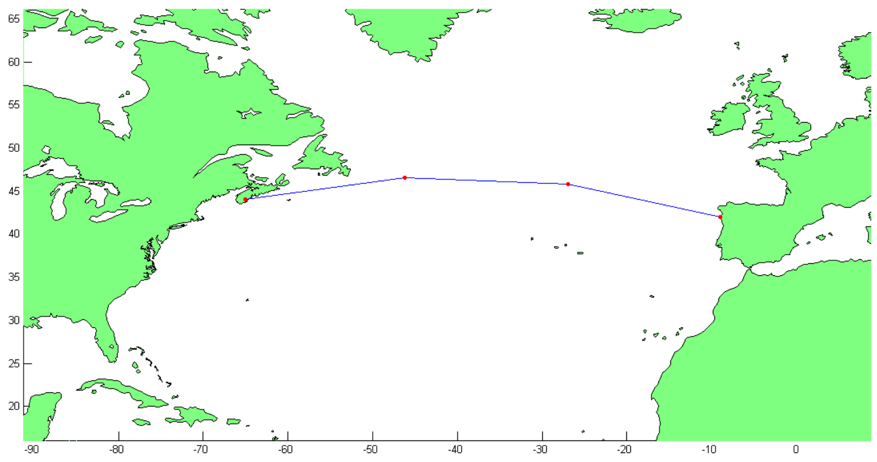


Figure 4.8.: Great circle route

In all the calculate fuel functions, the sea state that gives the velocity for the given arc is defines as the sea state at the arrival node for the arc. The accuracy of this assumption depends on the arc length. With long arc lengths this assumption would not give reasonable results, but for shorter lengths this could be a valid simplification. An evaluation of the sea states in the North Atlantic in January with an arc length equal to 60 nautical miles was performed. This evaluation says that the average difference in significant wave height between start and end node is 0.48 m, while the peak period differs with 0.43 s. The figure on the next page shows how the attainable speed varies with H_s and T_p for head sea, where the lines denote different peak periods.

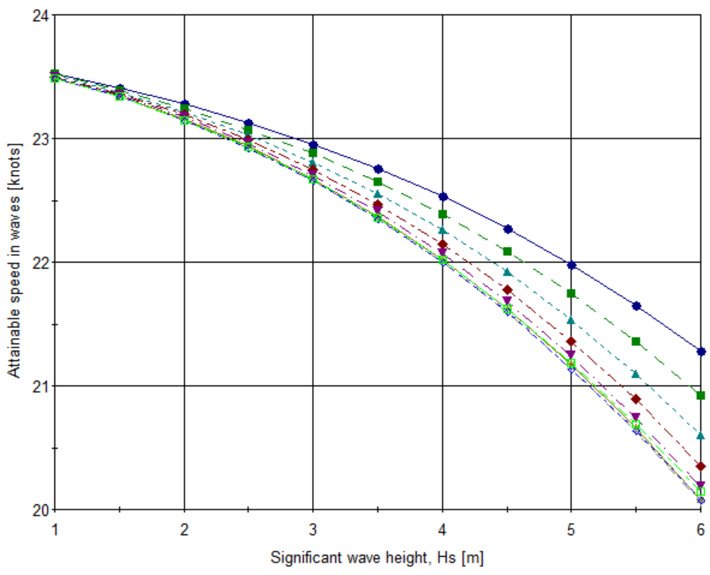


Figure 4.9.: Attainable speed difference with H_s and T_p

The velocity has low variation for calm sea, but the gradient increases with wave height. This gives the conclusion that the approximation stated would result in errors in the model and not provide accurate results. An average, or an interpolation between the two point would be more accurate.

The optimal fuel scripts operate like the method described in sec. 3.2. However, some additional comments with regard to the structure of the function must be made. All nodes are defined as matrices containing the geographical coordinates. The arc costs, i.e. fuel costs, are defined by creating three matrices where one is the cost of going straight, one right and one left. The velocity is read from ShipX file or calculated for the corresponding sea state. The weather file might include Not a Number, NaN, values when the coordinates define land mass. These NaN values are assigned calm water values in order to avoid error in the program. This assures that if the start and end coordinates are on land the program will not stop, but the optimal route might be over land mass.

An integrated Matlab function is used to find the optimal route. This function uses Dijkstra's algorithm to solve the network problem. To use this function a sparse matrix must be defined with information about departure node, arrival node, and cost of the traveled path between nodes. This sparse matrix is set up by assigning a number to each node, previously they where just geographical coordinates, and creating three vectors. The first vector includes all the departure nodes, the second all the arrival nodes and the third the cost of going from one to the other. The minimal fuel path is found, the nodes selected and the total fuel consumption calculated. A map including of the optimal route is created by reassigning geographical coordinates to the nodes.

5. Results and discussion

The program has been run for a number of different routes and seasons. The results of these different routes are mainly the same with regard to fuel savings, therefore a small selection of routes and months will be presented. The first voyage that will be presented is a voyage over the North Atlantic, while the second route is between Washington state and Japan. Both routes are tested in January and July with 15 center points and the following constrains for seakeeping:

Probability of slamming<0.03
Probability of Deck wetness<0.05
RMS of roll $<6°$
RMS of relative vertical acceleration$<1.37\ m/s^2$

The results will be presented with maps of the optimal routes compared with the great circle route, and tables including the total fuel consumption for all route options. The weather gradient over the two oceans will be evaluated and compared with the results. The weather files used are in GRIB format and collected form The European Centre for Medium-Range Weather Forecasts (ECMWF), ecmwf (2014). These are historical weather data, and include information about significant wave height of wind and swell, mean wave direction, mean wave periods and the U- and V-components of wind. Historical data is used to get the results, this because this provides the opportunity of evaluating different seasons and discuss when and where weather routing is beneficial. For future use of the program weather forecasts may also be used.

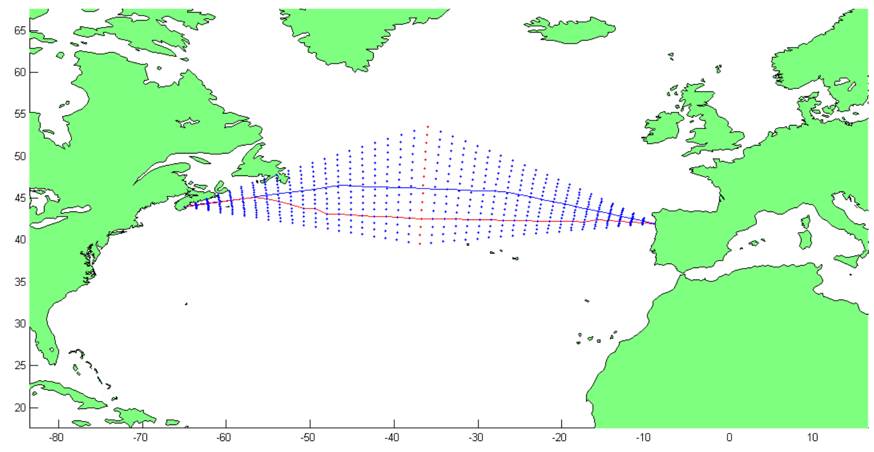
All optimal routes has been have for estimated wind and actual wind. Estimated wind means that the wind speed is estimated from the significant wave height, and that the attainable speed is read directly from the ShipX result. The actual wind is the mean wind collected from the GRIB file, and the attainable speed is calculated by the calculatespeed function in Matlab.

5.1. North Atlantic

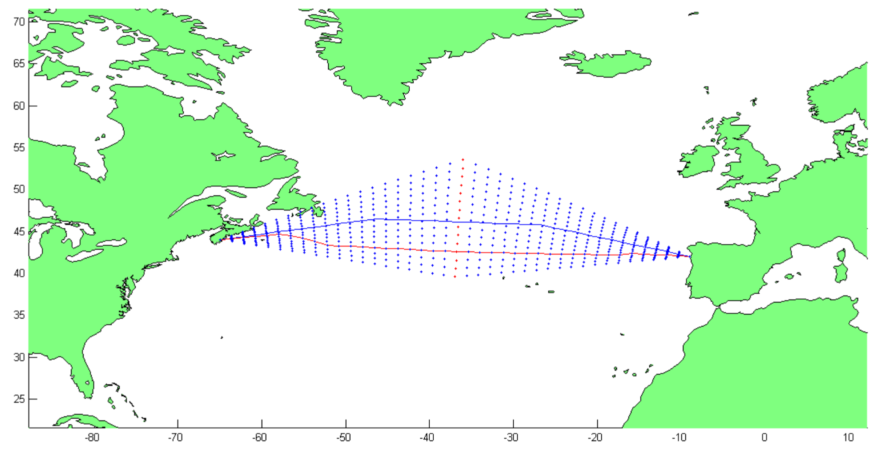
The voyage over the North Atlantic goes from Nova Scotia in Canada to Portugal. The departure coordinates are (44,-65). The arrival coordinates are (42,-9). The great circle distance between departure and arrival is 2 414 nautical miles. This route is chosen because there is little land mass between the two points. The weather in the North Atlantic can be rough and varies with season, which hopefully will have impact of the results.

5.1.1. January

The figures below show the optimal route vs the great circle route and the network of nodes over the Atlantic. The blue line denotes the great circle, the red line is the optimal route, the red dots are the center nodes and all the other nodes are blue.



(a) Estimated wind



(b) Actual wind

Figure 5.1.: Optimal route January

The figures show that both optimal routes are significantly south of the great circle route, while there is little difference between the estimated and actual wind. This means that the wind resistance either has little influence on the route suggestion or that the estimated wind resistance is accurate. Tab. 5.1 presents the calculated fuel consumption for the different routes.

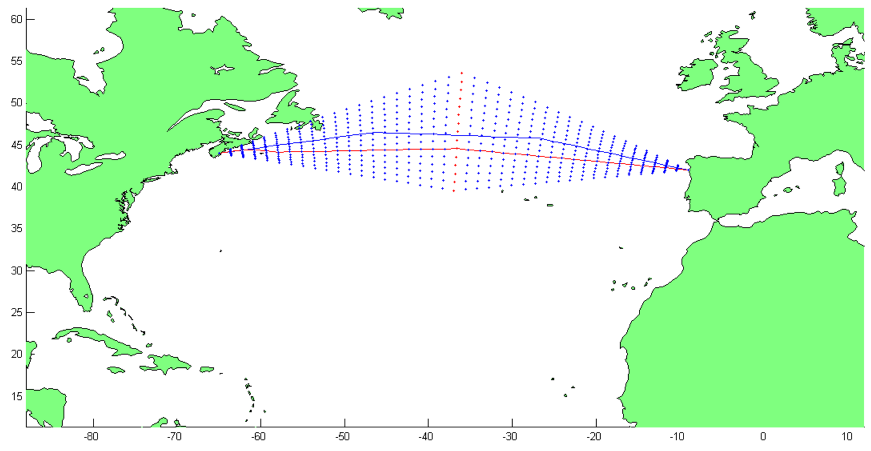
	Great circle fuel	Opt. fuel estimated wind	Opt. fuel actual wind
FC [ton]	503.2	512.6	499.9

Table 5.1.: Fuel consumption comparison for North Atlantic in January

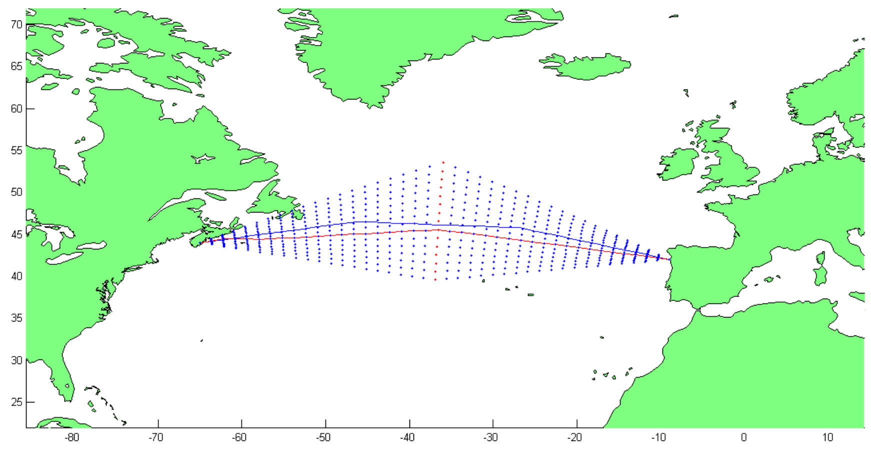
With a first view of the results above it looks as though the optimization process is not working, since the great circle route has a lower fuel consumption than the optimal route. However, as mentioned in sec. 1.5, the optimal solution can be defined at the best possible solution, and in this case we have defined a set of constrains, which limits the the feasible region for the vessel. The responses of the vessel need to be evaluated in order to determine whether the optimal solution is wrong, or the great circle route exceeds the seakeeping criteria. The Matlab program creates excel spreadsheets for each run which includes the sea state, the vessel responses and velocity for each arc. The complete spreadsheet for the North Atlantic route in January can be found in the folder together with the Matlab code and input. When evaluating these results it can be seen that the RMS values for roll are exceeded for a large number of nodes on the second half of the voyage. The program operates as expected and avoids these nodes, and therefore the optimal route has a higher fuel consumption than the great circle route. To understand why the fuel consumption for actual wind is lower than estimated wind the wind and ship's speed must be further evaluated. This is done in sec. 5.3.1.

5.1.2. July

The figures below shows the optimal route in July.



(a) Estimated wind



(b) Actual wind wind

Figure 5.2.: Optimal route July

The optimal route in July is fairly close the the great circle route. The deviation between estimated wind and actual wind is small also in July. When evaluating the sea state and the responses in July the results are, as expected, that the seakeeping criteria is never exceeded, and the vessel can choose to visit every node in the network.

The table below shows the fuel consumption for each route.

	Great circle fuel	Opt. fuel estimated wind	Opt. fuel actual wind
FC [tons]	506.9	504.3	494.5

Table 5.2.: Fuel consumption comparison for North Atlantic in July

Here it can be seen that the optimal route has a slightly lower fuel consumption the the great circle route. This means that the optimization process is working as expected. However, the saving is only 0.4 % and actually negligible. It can also be noticed that the fuel consumption for the great circle route is slightly higher in July than in January. This suggest that the rough weather in January does not give added resistance, but less resistance, and that the relative heading between the vessel and wave is beneficial and provides the vessel with extra speed. However, this difference is small, only three tons, and the results might be random.

5.1.3. Sea state

For minimal fuel weather routing to be beneficial an alternative route needs to have significantly better weather, or a more favorable wave heading. The great circle route, is as mentioned, the shortest distance on the sphere, and if a longer route should be the optimal the weather must differ a lot from one route til another.

To understand the chosen routes in the North Atlantic, and why the fuel savings are low, if any at all, the weather gradient must be evaluated. As mentioned, excel spreadsheets containing this information are generated while running the program. Optfuel contains the following: H_s, T_p, β, mean wave direction, ship's speed, RMS of roll and acceleration and probability of green water and slamming. Optfuel2 which is created while running for actual wind contains information about the ship's speed and wind speed and heading.

Tab. 5.3 shows the sea states for the center points on the North Atlantic route, i.e. the red points on the map. This is where the distance is the largest between the possible vessel position, i.e. from north to south, and where the weather gradient is expected to be largest. The wave headings in the table are calculated by the straight arcs, as presented in fig. 3.8.

	January			July		
(Latitude, Longitude)	H_S [m]	T_p [s]	β [deg]	H_S [m]	T_p [s]	β [deg]
(53.5, -36.0)	5.3	13.3	132	2.7	9.3	162
(52.6, -36.1)	5.1	13.6	130	3.1	9.9	163
(51.6, -36.1)	4.9	13.8	127	3.4	10.3	164
(50.6, -36.2)	4.8	13.8	124	3.5	10.5	164
(49.6, -36.3)	4.8	13.7	121	3.6	10.6	162
(48.6, -36.4)	4.7	13.5	120	3.6	10.5	162
(47.6, -36.4)	4.8	13.4	122	3.4	10.2	163
(46.6, -36.5)	4.9	13.3	126	3.1	9.8	165
(45.6, -36.6)	5.0	13.2	130	2.6	9.4	171
(44.6, -36.6)	5.0	13.3	133	2.3	9.1	179
(43.6, -36.7)	5.1	13.5	136	2.1	8.7	174
(42.6, -36.8)	5.1	13.6	139	1.9	8.4	170
(41.6, -36.8)	5.2	13.7	140	1.7	8.4	167
(40.6, -36.9)	5.3	13.7	139	1.5	8.5	165
(39.6, -37.0)	5.5	13.5	135	1.4	8.9	164

Table 5.3.: Sea states, North Atlantic

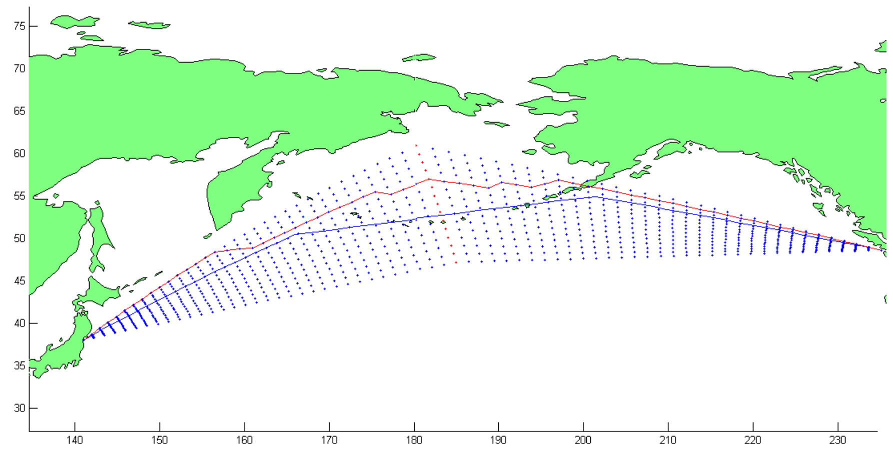
The weather in January is almost uniform over the area, and it is not much to gain by choosing a longer route. The weather actually varies more in July than in January, where the significant wave height varies from 3.6 m to 1.4 m. Even though the wave height varies, the optimal route is close to the great circle route. It can therefore be concluded that the actual distance on the sphere has an higher impact on the fuel consumption than the sea state. It also shows that the wave heading is mostly following seas, which in most cases gives low or negative added resistance.

5.2. North Pacific

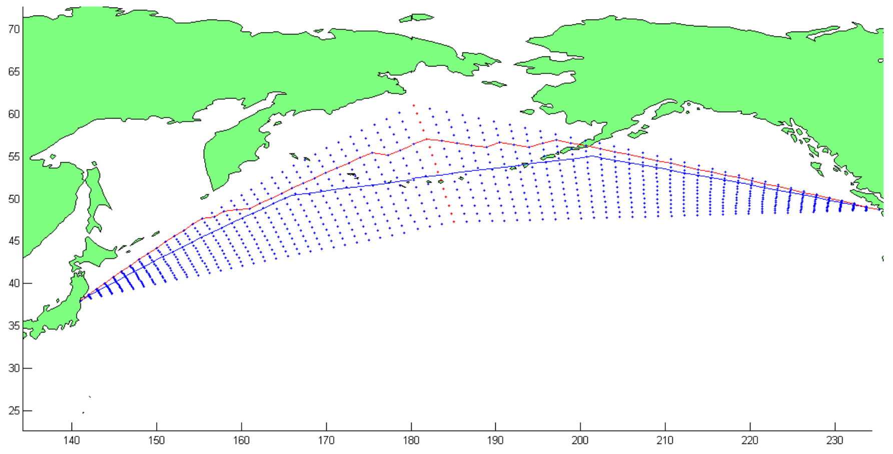
The voyage over the North Pacific goes from Washington state in USA, to Japan. The departure coordinates are (48,-125). The arrival coordinates are (38,-141). The great circle distance between departure and arrival is 3 895 nautical miles. This is a longer route than the North Atlantic route, and therefore gives more route options. There is more land mass than in the North Atlantic route, which the program does not account for, so that will not be a part of the discussion.

5.2.1. January

The figures below show the optimal routes in January.



(a) Estimated wind



(b) Actual wind

Figure 5.3.: Optimal route January

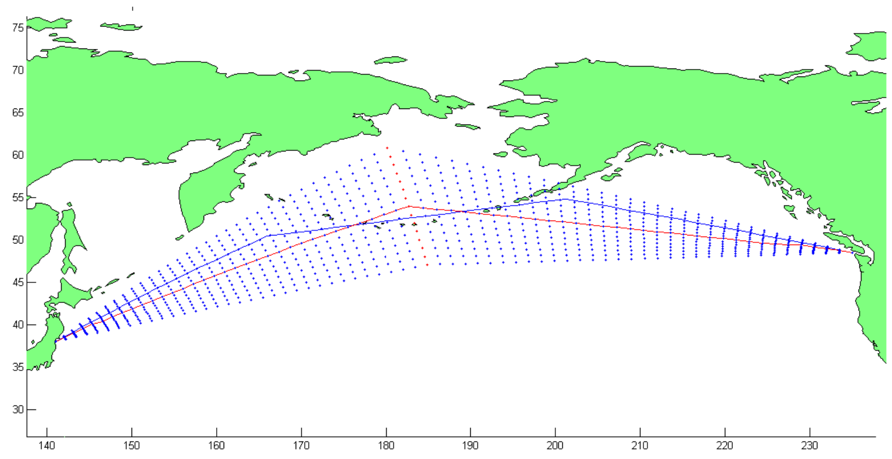
As for the North Atlantic route the optimal route with estimated and actual wind are almost identical. The optimal routes are north of the the great circle route, and the vessel's heading changes often. When evaluating the seakeeping performance of the vessel, the criteria for acceleration and roll are exceed a few times. However, the number of nodes with these sea states is small, so this would not affect the result. From the table below it can be seen that the fuel consumption for the optimal route is actually higher than for the GCR. It seems as though the optimization process for this route is not working correctly. The great circle route can not be exactly chosen with the optimization procedure, but it should be able to choose a route closer to it, and the fuel consumption might have been lower. The higher fuel consumption in this case can only be explained with calculation inaccuracy, and that the coordinates chosen in the GRC deviates slightly from the coordinates of the nodes in the network. When the sea state is not identical the ship's speed will vary and so will the fuel consumption. An example of this kind of error is described in sec. 5.3.1.

	Great circle fuel	Opt. fuel estimated wind	Opt. fuel actual wind
FC [tons]	829.1	833.6	812.5

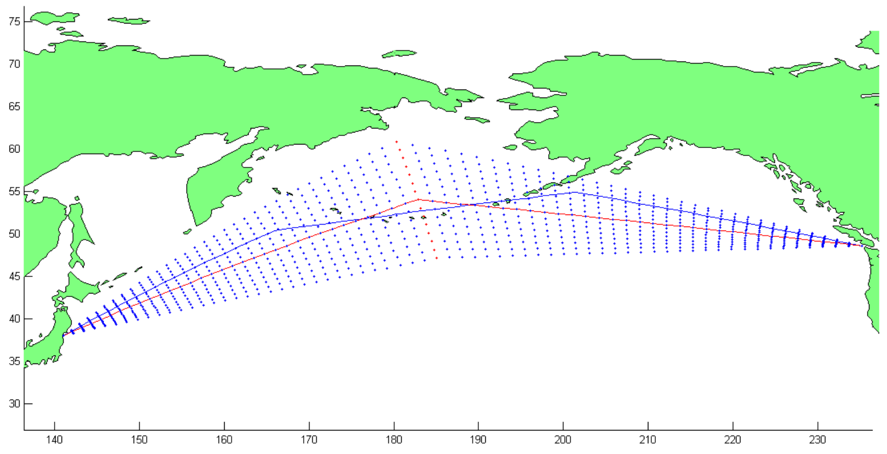
Table 5.4.: Fuel consumption comparison for North Pacific January

5.2.2. July

Below are the optimal routes for July.



(a) Estimated wind



(b) Actual wind

Figure 5.4.: Optimal route July

As for the routes in July on the North Atlantic, the routes are straight, and changes heading at the center point. Most of the route is south of the great circle route, and the route can be seen as a practically good route since the ship master only needs to change heading once. The route for estimated wind and actual wind are identical, as for all the other routes.

	Great circle fuel	Opt. fuel estimated wind	Opt. fuel actual wind
FC [tons]	812.3	819.7	803.1

Table 5.5.: Fuel consumption comparison for North Pacific in July

The table with fuel consumption shows that the optimal routes has 0.9 % higher fuel consumption than than the great circle route. This is discussed in sec. 5.3.1. The fuel consumption in the North Pacific is lower in July than in January, this suggests that the vessel is subjected to head seas and that the resistances decreases due to wave height decrease.

5.2.3. Sea states

The different sea states for each of the center nodes is presented in the table below.

	January			July		
(Latitude, Longitude)	H_S [m]	T_p [s]	β [deg]	H_S [m]	T_p [s]	β [deg]
(47.1, -175.1)	4.5	12.7	18	1.8	9.3	19
(48.1, -175.4)	4.6	12.5	21	1.8	9.3	26
(49.0, -175.7)	4.5	12.3	25	1.7	9.4	33
(50.0, -176.0)	3.8	12.1	32	1.4	10.2	45
(51.0 -176.3)	3.3	11.9	37	1.3	10.4	53
(52.0, -176.6)	2.4	10.8	14	1.0	9.4	43
(53.0 -177.0)	2.3	11.5	12	1.0	9.1	34
(53.9, -177.3)	2.4	11.5	11	1.0	9.2	38
(54.9, -177.6)	2.4	11.6	20	1.0	9.2	45
(55.9, -178.0)	2.4	12.1	31	1.0	8.2	55
(56.9, -178.3)	2.5	12.2	50	1.0	7.7	61
(57.9, -178.7)	2.5	11.9	62	1.2	7.4	62
(58.8, -179.1)	2.8	11.4	86	1.4	7.3	63
(59.8, -179.5)	3.4	10.5	105	1.4	7.3	65
(60.8, -179.8)	3.6	10.2	105	1.2	7.8	73

Table 5.6.: Sea states, at center nodes, North Pacific

The table shows that the wave heights in January are lower in the in the northern half of the network. In this half of the map the vessel is also more subjected to beam and following seas. This corresponds well with the chosen route. The weather in

July has low variation. The route chosen is straight, so this correspondences well.

5.3. Discussion

To validate the fuel calculations a comparison between the results and reported daily fuel consumption for similar vessels was done. Total distance between the departure and destination in the North Atlantic route is 2 414 nm, for a vessel sailing at 23 knots the voyage would take 4.3 days. A fuel consumption of 506 tons corresponds to a daily consumption of 117 tons. The distance in the North Pacific route is 3 895 nm, this corresponds to 7 days sailing time and a fuel consumption of 116 tons per day. The table below shows daily fuel consumption for four different vessels. This information is collected from sea web.com (2013).

Ship name	**Length (L_{PP})**	**Power [kW]**	**V [knots]**	**FC per day [tons]**
APL AMMAN	221	28,880	22.5	105
PRIMAVERA	218	28,880	23	105
ST LOUIS EXPRESS	232	29,243	21.6	125.4
XIANG SHUI WAN	229.5	27,290	22.5	100

Table 5.7.: Comparing of fuel consumption

The vessel used in this thesis is 233 m long, and with a brake power of 28,710 kW. A fuel consumption of 116 tons a day seems like a fair result.

5.3.1. Estimated wind vs actual wind

For the North Pacific it was seen that the optimal fuel consumption is higher than the great circle fuel consumption even tough the wind is estimated for both routes. This can be explained by inaccuracy in the calculation caused by rounding of the numbers. The speed loss analysis in ShipX was performed for a number of wave heights, periods and directions. However, all the values where integers in order to not overload the program. This means that when the significant wave height, peak period and heading is read from the weather file it is rounded off to the nearest integer, or for the heading, the nearest 30°. This provides differences between the two routes. For example, the significant wave height close to the GRC could be 1.4, while H_s for the optimal route is 1.6 m. When the heading is approximately 30°, and $T_p = 9$ s this will give a difference in attainable speed of 0.3 knots. This may seem insignificant, and it probably does not have a large effect on the final result, but when adding up these types of inaccuracies it can be understood that the fuel consumption calculated for the optimal route is higher than the great circle route.

For all routes the fuel consumption for calculated wind is lower than for actual wind. In order to evaluate this, we need to know whether the ship's speed is overestimated i Matlab as discussed in sec. 4.2.2, or if the wind velocity is overestimated in ShipX. This can be evaluated by comparing the significant wave height in optfuel and the wind velocity in opfuel2. The table below includes the estimated wind from H_s, the foretasted wind velocity and the corresponding ship speeds. The values are collected from the North Atlantic route in January and shows the wind and ship's speed for the center points.

(Latitude, Longitude)	Estimated wind [m/s]	Actual wind [m/s]	Dif. [%]	ShipX speed [m/s]	Matlab speed [m/s]	Dif. [%]
(53.5,-36.0)	15.7	12.4	26.9	23.8	24.9	-4.4
(52.6, -36.1)	15.3	10.6	45.5	23.8	24.7	-3.8
(51.6, -36.1)	15.2	11.0	38.3	23.8	24.7	-3.8
(50.6, -36.2)	15.0	12.7	18.3	23.8	24.6	-3.1
(49.6, -36.3)	14.9	14.0	6.4	23.8	24.4	-2.6
(48.6, -36.4)	14.9	15.0	-0.6	23.8	24.5	-2.7
(47.6, -36.4)	14.9	14.8	1.1	23.8	24.7	-3.4
(46.6, -36.5)	15.1	14.6	3.5	23.8	24.9	-4.4
(45.6, -36.6)	15.2	14.6	4.3	23.8	24.9	-4.5
(44.6, -36.6)	15.3	14.2	8.0	23.8	24.9	-4.3
(43.6, -36.7)	15.4	13.5	13.7	22.3	23.0	-3.0
(42.6, -36.8)	15.4	13.3	16.0	22.5	23.2	-2.8
(41.6, -36.8)	15.5	12.1	28.6	22.5	23.1	-2.4
(40.6, -36.9)	15.7	12.4	26.4	22.5	22.9	-1.7
(39.6, -37)	16.0	17.4	-8.3	23.8	24.5	-2.8

Table 5.8.: Comparison wind and ship's velocity, North Atlantic, January

The table shows that the wind speed in ShipX is generally overestimated. An explanation for the overestimated wind is that the significant wave height is reported as significant wave height of combined wind and swell, while the estimated wind speed formula is purely waves from wind. The table also shows that the attainable speed from ShipX is lower than the speed from Matlab, this could be explained with lower added resistance due to wind, but is is clear that the ship's speed is not strongly dependent on the wind velocity. The calm water resistance and added resistance due to waves has a larger impact on velocity calculations. An evaluation of the entire network shows that the estimated wind is at average 61 % higher than the actual wind, while the speed from ShipX is at average 2.7 % lower than the speed calculated in Matlab.

As discussed in sec. 4.2.2 the Matlab function overestimates the speed especially for rough seas because of thrust loss due to partly immersed propeller. The significant wave height corresponding to these values is approximately 5 meter, and thrust loss could occur in this sea state. This combined with the overestimated wind could explain the difference in the speed calculations. To evaluate this further the wind and velocity has been extracted for the North Pacific route in July, where the sea is calm.

(Latitude, Longitude)	Estimated wind [m/s]	Actual wind [m/s]	Dif. [%]	ShipX speed [m/s]	Matlab speed [m/s]	Dif. [%]
(47.1, -175.1)	9.1	7.4	23.3	22.9	23.3	-1.8
(48.1, -175.4)	9.1	7.6	20.2	22.9	23.3	-1.8
(49.0, -175.7)	8.8	6.9	28.0	22.9	23.3	-1.8
(50.0, -176.0)	8.2	3.5	136.0	23.2	23.7	-2.2
(51.0 -176.3)	7.9	3.1	154.1	23.2	23.7	-2.4
(52.0, -176.6)	6.8	3.6	90.5	23.2	23.7	-2.3
(53.0 -177.0)	6.8	2.0	247.6	23.2	23.7	-2.3
(53.9, -177.3)	6.8	2.1	222.1	23.2	23.6	-1.9

Table 5.9.: Comparison wind and ship's velocity, North Pacific, July

The table shows that the estimated wind varies more for high seas, the ship's speed less, but the trend is the same as for North Atlantic in January. Due to the argument presented in the beginning of this section, where the wave height, and thus the wind speed is rounded up, one would expect a difference in the velocity calculations, but the Matlab velocity should not be consistently higher. This could be a result of interpolation differences and simplified calculations. Initially it can be assumed that the Matlab calculated speed should be more accurate since the sea state used in this calculation is exact, but it can be seen that at even for calm water the speed is slightly higher than the service speed calculated in the model test.

The aim of this thesis is to provide a route suggestion, and all results indicate that the route is the same regardless of method used. It can therefore be recommended that the user should use the estimated wind calculation method since this methods requires less input and the process time is significantly lower.

5.3.2. Comparison with previous research

To ensure that the results found in thesis are adequate a comparison with previous research has also been done. Some of the research presented in the first chapter also presents some case studies. Two of these studies will be briefly presented here and compared to the results found in this thesis.

Windeck (2013) uses a shortest path algorithm similar to the one in this work to compare fuel consumption with and without wind propulsion. The computational test was perform for April between Le Havre and Miami. The resulting routes are presented in the figure below.

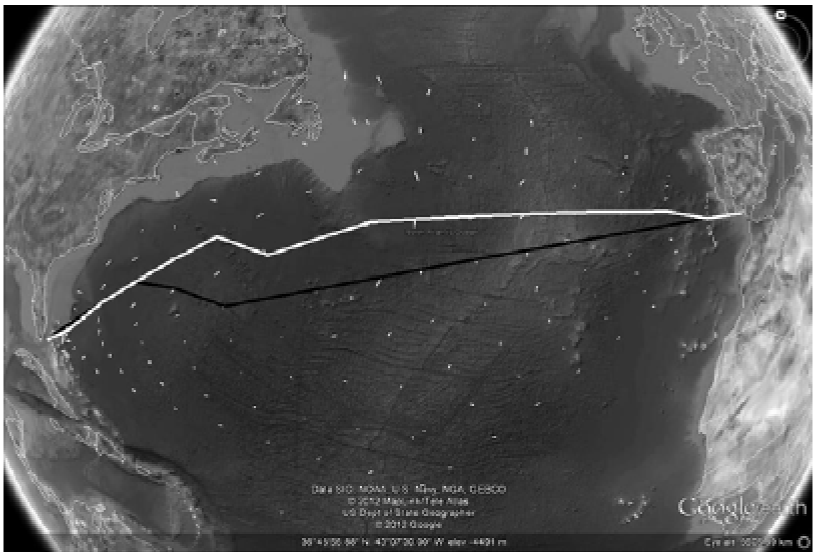


Figure 5.5.: Optimal route with and without sail assistance from Windeck (2013)

The white line denotes a vessel with sail assistance, while the black line is without. Windeck (2013) does not elaborate on the fuel savings between the optimal route and the great circle route, but the optimal route without sail corresponds well with the great circle route, so it can be assumed that the fuel savings are not high.

Shao et al. (2012) used 3D dynamic programing to find the optimal route. The route used in their work is similar to the North Atlantic route in this thesis, the computation is with weather data from January.

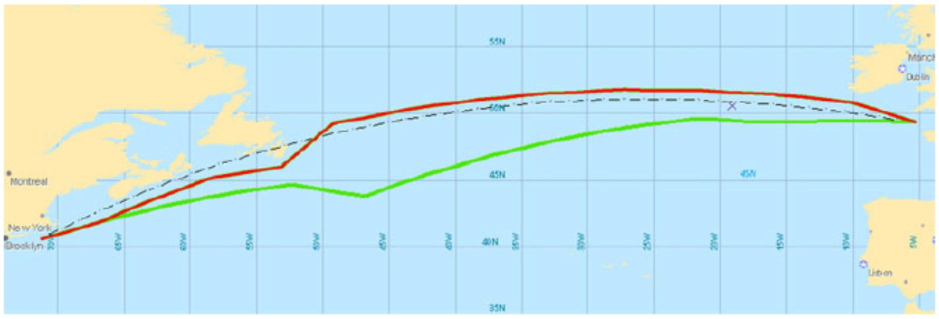


Figure 5.6.: Optimal route with 2D and 3D dynamic programing from Shao et al. (2012)

The red line denotes the 3D dynamic programing developed by Shao et al. (2012), while the green route is found by using an older optimization method called 2D dynamic programing. This article does not compare the fuel consumption between the optimal and great circle. However, when looking at the routes it is clear that the red route is fairly close to the great circle route, represented by the dotted lines, and therefore there is probably not significant fuel saving with this method either.

When evaluating the results from this thesis, and the previous research it can be concluded that there is not much to gain in fuel savings by using weather routing.

6. Conclusion

6.1. Conclusion

The results in chapter 5 showed that weather routing has no impact on fuel savings. The overall conclusion is therefore that weather routing can be used for ships to avoid rough weather, but is not beneficial for fuel savings. There are two main reasons for this: The first is that the actual distance on the sphere has larger impact on the fuel consumption than the added resistance from waves and wind. The second is that over an ocean for a given month the weather gradient is low, meaning that there is little change in weather between the northernmost route and the southernmost route. If a local storm lies within the feasible region the storm will be avoided due to seakeeping criteria.

6.2. Recommendations

Weather routing is only necessary for avoiding rough weather, this means that the accuracy in fuel consumption calculations is unimportant. The seakeeping performance of the vessel is equal for both wind resistance calculations. The program should be used for estimated wind. This version runs much faster, and the routes for estimated wind and actual wind are identical.

6.3. Future work

Since the conclusion is that weather routing does not result in fuel savings there is no further work needed with this method. However, if wind propulsion becomes common for commercial shipping, weather routing could again be an important aspect. For sail boats weather routing is an established method to find the fastest track over an ocean, because the lift of the sail is highly dependent on the wind direction. If wind propulsion becomes a significant part of the ship's propulsion is can be assumed that weather routing would have higher impact of the optimal route.

Bibliography

Bowditch, N., 2002. The American practical navigator - AN EPITOME OF NAVIGATION. National Imagery and Mapping Agency.

Christiansen, M., Fagerholt, K., Nygreen, B., 2007. Maritime Transportation. Elsevier.

Denis, M. S., Pierson, W. J., 1953. On the motions of ships in confused seas. In: SNAME.

dictionary.com, 2014. dictionary.com.
URL `http://dictionary.reference.com/browse/isochrone`

Dijkstra, E. W., 1959. A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271.

ecmwf, 2014.
URL `http://apps.ecmwf.int/datasets/data/interim_full_daily`

Faltinsen, O. M., 1990. Sea loads on ships and offshore structures. Cambridge University Press.

Fathi, D., 6 2012. SHIPX Vessel Responses (VERES) Ship Motions and Global Loads. MARINTEK.

Gerritsma, J., Beukelman, W., 1972. Analysis of the resistance increase in waves of a fast cargo ship. International Shipbuilding Progress 19, 285–293.

Hagiwara, H., 1989. Weather routing of (sail-assisted) motor vessels. Ph.D. thesis, TU Delft.

Haltiner, G. J., Hamilton, H. D., Arnason, G., 1962. Minimal-time ship routing. Journal of Applied Meteorology 1, 1–7.

Hinnenthal, J., 2008. Robust pareto, optimum routing of ships utilizing deterministic and ensemble weather forecasts. Ph.D. thesis, Technischen Universität Berlin.

James, R. W., 1957. Application of wave forecasts to marine navigation. Ph.D. thesis, New York University.

Kreyszig, E., 2006. Advanced Engineering Mathematics. John Wiley & Sons, INC.

Lee, H., Kong, G., Kim, S., Kim, C., Lee, J., 2002. Optimum ship routing and its implementation on the web, 125–137.

Lee, W. T., Bales, W. L., Sowby, S. E., 1985. Standardized wind and wave environments for north pacific ocean areas. Tech. rep., David W. Taylor navel ship research and development center.

Lloyd, A. R. J. M., 1998. Seakeeping: Ship Behaviour in Rough Weather. A R J M Lloyd.

Loukakis, T., Sclavounos, P., 1978. Some extensions of the classical approach to strip theory of ship motions, including the calculation of mean added forces and moments. Journal of Ship Research 22, 1–19.

Lundgren, J., Rönnqvist, M., Värbrand, P., 2010. Optimization. Studentlitteratur AB, Lund.

Myrhaug, D., 2007. Lecture notes, TMR4180 Marine Dynamics: Irregular sea (Marin Dynamikk: Uregelmessig sjø). NTNU.

Nakamura, S., Naito, S., 1975. Propulsive performance of a container ship in waves. J.S.N.A. Kansai Japan 158.

NORDFORSK, 1987. Assessment of a ship performance in a seaway.

oceanroutes.com, 2013.
URL `http://www.oceanroutes.com/about.html`

Papadakis, N. A., Perakis, A. N., 1990. Deterministic minimal time vessel. Routing, Operations Research 28, 426–438.

Perakis, A. N., Papadakis, N. A., 1989. Minimal time vessel routing in a time-dependent environment. Transportation Science 23, 266–276.

sea web.com, 2013.
URL `http://www.sea-web.com/seaweb_welcome.aspx`

Shao, W., Zhou, P., Thong, S. K., 2012. Development of a novel forward dynamic programming method for weather routing. Journal of Marine Science and Technology 17, 239–251.

Steen, S., 2007. Lecture Notes, TMR4247 Marine Technology 3: Resistance and Propulsion (Marin teknikk 3: Motstand og propulsjon). NTNU.

Steen, S., Minsaas, K., 2012. Lecture notes, TMR4220 Naval Hydrodynamics: Ship Resistance. NTNU.

Steen, S., Minsaas, K., 2013. Lecture Notes, TMR4220 Naval Hydrodynamics: Propeller Theory. NTNU.

Szlapczynska, J., Smierzchalski, R., 2007. Adopted isochrone method improving ship safety in weather routing with evolutionary approach. International Journal of Reliability, Quality & Safety Engineering 14, 635–645.

Thomson, K., May 2011. Weather routing. Meteotological Technology International, 6–10.

Windeck, V., 2013. A Liner Shipping Network Design : Routing and Scheduling Considering Environmental Influences. Produktion und Logistik. Springer Fachmedien Wiesbaden.

WärtsilaEngines, 2013.
URL `http://www.wartsila.com/en/engines/low-speed-engines/RT-flex82T`

A. Matlab program

A.1. main.m

```
%------------------------------------------------------------
%               main.m
%               Hege Eskild
%               Spring 2014
%
% This is the main script in the Matlab program including
% several underlying functions. The first part of the script
% in user input. User input is divided into route input,
% including start and end points and weather files, and
% Vessel input if the user wishes to change vessel.
% The second part is the functions, divided into read and
% Calculating functions. The read functions reads the ShipX
% Input and the GRIB files. The second parts calculates the
% fuel consumption for GCR and optimal route.
%
% All functions will include a short description and an
% Explanation of the variables. All input and output
% is defined and some of the internal variables. Not
% all internal variables are described in detail because
% many of them are "help" variables and not significantly
% important in order to understand the program.
%
%------------------------------------------------------------
%   Variable name                   Definition
%----------------------Input------------------------------
%   startLat                        Latitude coordinate departure
%   startLon                        Longitude coordinate departure
%   endLat                          Latitude coordinate arrival
%   endLon                          Longitude coordinate arrival
%   CalculateWindFromWave           Defines how to calculate wind
%                                   resistance
%   GRIBinput                       Weather file. Can be forecast or
%                                   hindcast in GRIB format
%   ProbSlamming                    Seakeeping criterion for slamming
%   ProbGreenWater                  Seakeeping criterion for deck
%                                   wetness
%   VAccFP                          Seakeeping criterion for vertical
%                                   acceleration
%   RMSRoll                         Seakeeping criterion for roll
%   NumCenPts                       Number of center points for opt
%                              route
%   DeltaNM                    Distance in NM between center
%                              points
%   GClat                      Four latitude values for the
%                              practical GCR
%   GClon                      Four longitude values for the
%                              practical GCR
%   TRANSinput(..)             Transfer files from ShipX (.re1)
%   SLinput                    Speed loss input file from ShipX
%                              (seaway_ship)
%   ADDEDRESinput              Speed loss input file from ShipX
%                              (seaway_ship.out)
%   Cwindinput                 Wind resistance coefficient input
%                              file
%   CalmResinput               Calm water resistance input file
%   Pb                         Desired brake power
%   bsfc                       Specific fuel consumption (g/kWh)
%   Xp                         X coordinate of FP (negative from
%                              CG)
%   freeboard                  freeboard at FP
%   A_T                        Transverse projected area
%   D                          Propeller diameter
%   Eta_M                      Mechanical efficiency
%   Eta_R                      Relative rotational efficiency
%   OpenWaterinput             Open water diagram, text file
%   C_wake                     Wake coefficient
%   C_t                        Thrust deduction coefficient
%
%--------------------Output------------------------------
%   AVspeed                    Matrix with attainable speed for
%                              different sea states
%   NOHEAD                     Number of different headings in
%                              Avspeed
%   NOTpHs                     Number of Tp and Hs in Avspeed
%   AddedRes                   Matrix with increased resistance
%                              due to waves
%   C_wind                     Matrix with wind drag coefficient
%                              for different headings
%   R_calm                     Calm water resistance matrix
%   OpenWater                  Matrix with open water values
%   lon                        Vector with longitude values from
%                              GRIB
%   lat                        Vector with latitude values from
%                              GRIB
%   time                       Vector with time values from GRIB
%   Uwind                      Matrix with U component of wind
%                              from GRIB
%   Vwind                      Matrix with V component of wind
%                              from GRIB
%   SWH                        Matrix with Hs from GRIB
%   MWD                        Matrix with mean wave direction
%                              from GRIB
%    MWP                             Matrix with mean wave period
%                                    from GRIB
%    TRANS(..)                       Matrices including transfer
%                                    functions for roll and heave/
%                                    pitch for different frequencies
%                                    heading and velocity.
%    NOHEAD                          Number of heading in transfer
%                                    file
%    NOFREQ                          Number of frequencies in
%                                    transfer files
%    Lpp                             Length per perpendiculars
%    GCFC                            Fuel consumption for GCR
%    GClats                          Latitude vector for GCR
%    GClons                          Longitude vector for GCR
%    FC                              Fuel consumption for opt. route

clear all
close all

%---------------------------------------------------------------
% The first part of the main file is for user input, both files
% and variables

%Define start and end point of voyage
startLat=48.5;
startLon=-125;
endLat=38;
endLon=141;

%Do you wish to calculate wind velocity from significant wave
%height? Write 1 for calculating wind from waves, 0 otherwise
CalculateWindFromWave=1;

%Input GRIB file
GRIBinput='janNP.grib';

%Define operability criteria; Probability of slamming and
%green water, as well ass vertical Acc at bow and rms of roll
ProbSlamming=0.03;
ProbGreenWater=0.05;
VAccFP=1.37; %[m/s^2]
RMSRoll=6;   %[deg]

%Define number of center points, odd number. Defines the range
%in which the vessel can travel. 11 center point gives 600
%nautical miles range at %center of the route
NumCenPts=15;

[GClat,GClon]=gcwaypts(startLat,startLon,endLat,endLon,3);


%Define arclength for calculating fuel consumption
```

```
DeltaNM=60;          %Deltalengtt in NM

%Define transferinput from ShipX name input.re1
%The functions are calculated with different wave amplitude
%Only include files with the same values for heading and period.
TRANSinputHP=fopen('TRANSHP.txt','r');
TRANSinputH2=fopen('TRANSH2.txt','r');
TRANSinputH4=fopen('TRANSH4.txt','r');
TRANSinputH6=fopen('TRANSH6.txt','r');
TRANSinputH8=fopen('TRANSH8.txt','r');
TRANSinputH10=fopen('TRANSH10.txt','r');

%Define speed loss input file from ShipX called seaway_ship
%(found in therun folder). File is modified. Copy only
%available speed in waves,including number of headings
%and number of Hs and Tp
SLinput=fopen('SpeedLoss_ShipX.txt','r');

%Define files for calculating the speed loss by separate wind
%velocity. The speed loss file from ShipX is know the
%seaway_shipX.out. Remove everything until the defining of
%the first heading. Define also a calm water resistance .txt
%file and a wind resistance coefficient file.
ADDEDRESinput='SpeedLoss2_ShipX.txt';
Cwindinput='WindResCoefficient.txt';
CalmResinput='CalmWaterRes.txt';


%Define constant brake power, arclength for calculating fuel
%consumption and ship motions and specific fuel consumption
Pb=28710.00;         %Brake power
bsfc=166;            %Brake specific fuel consumption in g/kWh

%Define some position of FP Xp, freeboard height and
%transverse projected area
Xp=-109.77;
freeboard=15.27;
A_T=850;

%Input propulsion settings
D=7.5;            %Propeller diameter
Eta_M=0.980;      %Mechanical efficiency
Eta_R=1.01;       %Rotational efficiency
C_wake=0.24;
C_t=0.18;
OpenWaterinput='OpenWater.txt';    %Open water diagram

setup_nctoolbox    %if needed

%-------------------------------------------------------------
%The second part of the main file is calculations
```

```
%Read ShipX Speed loss files
[AVspeed,NOHEAD,NOTpHs]=speedloss(SLinput);

%Reading increase in resistance due to waves, the wind
%resistance coefficient and calm water resistance
[AddedRes,C_wind,R_calm,OpenWater]=...
increaseaddedres(NOHEAD,NOTpHs,...
ADDEDRESinput,Cwindinput,CalmResinput,OpenWaterinput);

%Read GRIB file
[lon,lat,time,Uwind,Vwind,SWH,MWD,MWP]=readgrib(GRIBinput);

%Read ShipX transfer files
[TRANSR2,TRANSR4,TRANSR6,TRANSR8,TRANSR10,TRANSHP,NOFREQ,...
NOHEAD,Lpp]=readinput(TRANSinputHP,TRANSinputH2,TRANSinputH4...
,TRANSinputH6,TRANSinputH8,TRANSinputH10);

%Calculating GCfuel
 [GCFC,GClats,GClons]=gcfuel(GClat,GClon,Pb,bsfc,...
 AVspeed,lon,lat,SWH,MWD,MWP);

  if CalculateWindFromWave==1
 %Calculating Optimal route
 [FC]=optfuel(startLat,startLon,endLat,endLon,NumCenPts,Pb,...
 DeltaNM,bsfc,AVspeed,lon,lat,time,Uwind,Vwind,...
 SWH,MWD,MWP,ProbSlamming,ProbGreenWater,VAccFP,...
 RMSRoll,TRANSR2,TRANSR4,TRANSR6,TRANSR8,TRANSR10,TRANSHP,...
 NOFREQ,NOHEAD,Lpp,Xp,freeboard,GClats,GClons)

  elseif CalculateWindFromWave==0
 [FC]=optfuel2(startLat,startLon,endLat,endLon,NumCenPts,Pb,...
 DeltaNM,bsfc,AVspeed,lon,lat,time,Uwind,Vwind,...
 SWH,MWD,MWP,ProbSlamming,ProbGreenWater,VAccFP,...
 RMSRoll,TRANSR2,TRANSR4,TRANSR6,TRANSR8,TRANSR10,TRANSHP,...
 NOFREQ,NOHEAD,Lpp,Xp,freeboard,AddedRes,C_wind,R_calm,...
 OpenWater,A_T,Eta_M,C_wake,C_t,Eta_R,D,GClats,GClons)
    end
```

A.2. speedloss.m

```
%------------------------------------------------------------
%               speedloss.m
%               Hege Eskild
%               Spring 2014
%
% This functions reads the attainable speed from the ShipX
% speed loss file (seaway_ship). Writing out a matrix with the
% attainable speed for different sea states. Also gives out
% number of headings in the input file, and total number of
% Hs and Tp, to be used in increaseaddedres.m
%
%------------------------------------------------------------
```

```
%    Variable name                    Definition
%-------------------- Input --------------------------------------
%    SLinput                          Speed loss input file from ShipX
%                                     deduction
%--------------------Output--------------------------------------
%    AVspeed                          Matrix with attainable speed for
%                                     different sea states
%    NOHEAD                           Number of different headings in
%                                     Avspeed
%    NOTpHs                           Number of Tp and Hs in Avspeed
%-------------------Internal--------------------------------
%    HEAD                             Heading from input file
%    M                                Matrix including all columns in
%                                     the input file
%    j                                Counting integer
%    i                                Counting integer


function [AVspeed,NOHEAD,NOTpHs]=speedloss(SLinput)

%reading number of headings, as well as number of Hs And Tp
NOHEAD=fscanf(SLinput,'%f',1);
NOTpHs=fscanf(SLinput,'%f',1);

%Creating an attainable speed matrix
AVspeed=zeros(NOTpHs*NOHEAD,4);

for j=1:NOHEAD

HEAD=fscanf(SLinput,'%f',1);

for i=1:NOTpHs
    M=fscanf(SLinput, '%f %f %f %f %f %f %f %f %f %f %f %f',
      [1,12]);
    %Writing heading into the first column in the attainable
    %speed matrix
    AVspeed((j-1)*NOTpHs+i,1)=HEAD;
    %Writing Hs into the second column in the attainable
    %speed matrix
    AVspeed((j-1)*NOTpHs+i,2)=M(:,1);
    %Writing Tp into the third column in the attainable
    %speed matrix
    AVspeed((j-1)*NOTpHs+i,3)=M(:,2);
    %Writing attainable speed into the fourth column of matrix
    AVspeed((j-1)*NOTpHs+i,4)=M(:,7);
end

end

AVspeed;
```

```
end
```

A.3. increaseaddedres.m

```
%-------------------------------------------------------------
%                increaseaddedres.m
%                Hege Eskild
%                Spring 2014
%
% This read function reads several input files that is needed
% for actual wind. The output from this function is the
% the increase added resistance from waves as a percentage of
% the calm water resistance, a matrix including the wind
% resistance coefficient depended on wind heading, a matrix
% including the calm water resistance for a given velocity
% and the open water diagram used in prolusion calculations.
%
%-------------------------------------------------------------
%   Variable name                 Definition
%----------------------Input----------------------------------
%   NOHEAD                        Number of different headings in
%                                 Avspeed
%   NOTpHs                        Number of Tp and Hs in Avspeed
%   ADDEDRESinput                 Speed loss input file from ShipX
%                                 (seaway_ship.out)
%   Cwindinput                    Wind resistance coefficient input
%                                 file
%   CalmResinput                  Calm water resistance input file
%   OpenWaterinput                Open water diagram, text file
%-----------------------Output-----------------------------
%   AddedRes                      Matrix with increased resistance
%                                 due to waves
%   C_wind                        Matrix with wind drag coefficient
%                                 for different headings
%   R_calm                        Calm water resistance matrix
%   OpenWater                     Matrix with open water values
%-----------------------Internal-----------------------------
%   LengthCwind                   Number of rows in windres
%                                 coefficient file
%   LengthCalm                    Number of rows in calm water
%                                 resistance file
%   M                             Variable including all columns in
%                                 input file
%   j                             Counting integer
%   i                             Counting integer


function [AddedRes,C_wind,R_calm,OpenWater]=increaseaddedres...
    (NOHEAD,NOTpHs,ADDEDRESinput,Cwindinput,CalmResinput,...
     OpenWaterinput)
```

```
Cwind=fopen(Cwindinput,'r');
Calm=fopen(CalmResinput,'r');
Added=fopen(ADDEDRESinput,'r');
OpenW=fopen(OpenWaterinput,'r');

LengthCwind = numel(textread(Cwindinput,'%1c%*[^\n]'));
LengthCalm = numel(textread(CalmResinput,'%1c%*[^\n]'));


%Writing matrix for wind resistance coefficient
for i=1:LengthCwind
    %enters heading into first column of wind coefficient
    %matrix
    C_wind(i,1)=fscanf(Cwind,'%f',1);
    %enters coefficient into second column of wind
    %coefficient matrix
    C_wind(i,2)=fscanf(Cwind,'%f',1);
end

%Writing matrix for calm water resistance
for i=1:LengthCalm
    %enters velocity into first column of the calm
    %resistance matrix
    R_calm(i,1)=fscanf(Calm,'%f',1);
    %enters resistance into second column of the
    %calm resistance matrix
    R_calm(i,2)=fscanf(Calm,'%f',1);
end


%Writing matrix for added resistance in waves
for i=1:NOHEAD
    char=fscanf(Added,'%c',[1,35]);
    head=fscanf(Added,'%f',1);
    char2=fscanf(Added,'%c',[1,300]);

    for j=1:NOTpHs
        M=fscanf(Added,'%f %f %f %f %f %f %f %f %f %f %f %f'
          ,[1,12]);
        %Defining heading as the first column
        AddedRes((i-1)*NOTpHs+j,1)=head;
        %Defining Hs as the second column
        AddedRes((i-1)*NOTpHs+j,2)=M(1);
        %Defining peak period as the third column
        AddedRes((i-1)*NOTpHs+j,3)=M(2);
        %Defining percentage resistance increase due to waves
        %as the fourth column
        AddedRes((i-1)*NOTpHs+j,4)=M(6);
    end
end
lengthOpenW=fscanf(OpenW,'%f',1);
for i=1:lengthOpenW
```

```
    M=fscanf(OpenW,'%f %f %f %f %f',[1,5]);
    %Writing J values into the first column
    OpenWater(i,1)=M(1);
    %Writing K_Q into the second column
    OpenWater(i,2)=M(2)/10;
    %Writing K_T into the third column
    OpenWater(i,3)=M(3);
    %Writing eta_0 into the fourth column
    OpenWater(i,4)=M(3)/(M(2)/10)*M(1)/(2*pi());
    %Writing Kt/j^2 into the fifth column
    OpenWater(i,5)=M(3)/M(1)^2;
end

end
```

A.4. readgrib.m

```
%------------------------------------------------------------
%                 readgrib.m
%                 Hege Eskild
%                 Spring 2014
%
% This function reads the GRIB weather file. Gives out
% vectors with latitude, longitude and time, and matrices
% with significant wave height, mean wave period, mean wave
% direction and U- and V-components of wind.
%
% If another GRIB files with different variable names is used
% The name inside the parentheses must be changed.
%
%------------------------------------------------------------
%   Variable name                Definition
%----------------------Input---------------------------------
%   GRIBinput                    Weather file. Can be forecast or
%                                hindcast in GRIB format
%---------------------Output--------------------------------
%   lon                          Vector with longitude values from
%                                GRIB
%   lat                          Vector with latitude values from
%                                GRIB
%   time                         Vector with time values from GRIB
%   Uwind                        Matrix with U component of wind
%                                from GRIB
%   Vwind                        Matrix with V component of wind
%                                from GRIB
%   SWH                          Matrix with Hs from GRIB
%   MWD                          Matrix with mean wave direction
%                                from GRIB
%   MWP                          Matrix with mean wave period
%                                from GRIB
```

```
function [lon,lat,time,Uwind,Vwind,SWH,MWD,MWP]=readgrib(GRIBinput)

%Reading the GRIB file
wave=ncgeodataset(GRIBinput);
%Checking contends
wave.variables;

lon=wave{'lon'}(:);      %Defining longitude vector
lengthlon=length(lon);   %Defining length of lon vector
%If the lon values are defined as 0-360 degrees, the values are
%are change to be between -180 to 180 so that the corresponds
%to the Matlab values.
for i=1:lengthlon
    if lon(i)>180
        lon(i)=lon(i)-360;
    end
end
lat=wave{'lat'}(:);      %Defining latitude vector
lengthlat=length(lat);
time=wave{'time'}(:);    %Defining time vector
lengthtime=length(time);

%Creating matrices of size time*lat*lon including U and V
%component of wind, significant wave height, mean wave period
%and mean wave direction
Uwind=wave{'10_metre_U_wind_component_surface'}(:,:,:);
Vwind=wave{'10_metre_V_wind_component_surface'}(:,:,:);
SWH=wave{'Significant_wave_height_msl'}(:,:,:);
MWD=wave{'Mean_wave_direction_msl'}(:,:,:);
MWP=wave{'Mean_wave_period_msl'}(:,:,:);

S=size(MWP);
end
```

A.5. readinput.m

```
%-------------------------------------------------------------
%               readinput.m
%               Hege Eskild
%               Spring 2014
%
% This function reads the input transfer files from Shipx.
% Output is matrices including transfer function for different
% wave headings, frequencies and velocities. One transfer
% matrix is calculated for heave and pitch, and several for
% roll. The functions includes both the real and imaginary
% part of the transfer function. For the program to operate
% properly all transfer input files must be of the same size.
% i.e. calculated for the same velocities, headings and
% frequencies. Several for-loops are used to assign the right
% value to the right frequency, heading and velocity.
%
```

```
%------------------------------------------------------------
%   Variable name                   Definition
%----------------------Input-----------------------------
%   TRANSinput(..)                  Transfer files from ShipX (.re1)
%---------------------Output--------------------------------
%   TRANS(..)                       Matrices including transfer
%                                   functions for roll and heave/
%                                   pitch for different frequencies
%                                   heading and velocity.
%   NOHEAD                          Number of heading in transfer
%                                   file
%   NOFREQ                          Number of frequencies in
%                                   transfer files
%   Lpp                             Length per perpendiculars
%---------------------Internal--------------------------------
%   j                               Counting integer
%   i                               Counting integer
%   k                               Counting integer
%   NOVEL                           Number of velocities
%   NOHEAD                          Number of headings
%   NOFREQ                          Number of frequencies
%   NDOF                            Number of degrees of freedom


function [TRANSR2,TRANSR4,TRANSR6,TRANSR8,TRANSR10,TRANSHP,...
         NOFREQ,NOHEAD,Lpp]=readinput(TRANSinputHP,...
         TRANSinputH2,TRANSinputH4,TRANSinputH6,...
         TRANSinputH8,TRANSinputH10);

%--------------------------------------------------------
%Reading ShipX file for heave and pitch transfer function
%--------------------------------------------------------

info=fscanf(TRANSinputHP,'%c',[1,500]);

%Defining vessel constants
rho=fscanf(TRANSinputHP,'%f',1);
g=fscanf(TRANSinputHP,'%f',1);
Lpp=fscanf(TRANSinputHP,'%f',1);
B=fscanf(TRANSinputHP,'%f',1);
T=fscanf(TRANSinputHP,'%f',1);
LCG=fscanf(TRANSinputHP,'%f',1);
VCG=fscanf(TRANSinputHP,'%f',1);

%Defining number of velocities, headings, frequencies
%and degrees of freedom.
NOVEL=fscanf(TRANSinputHP,'%f',1);
NOHEAD=fscanf(TRANSinputHP,'%f',1);
NOFREQ=fscanf(TRANSinputHP,'%f',1);
NDOF=fscanf(TRANSinputHP,'%f',1);

%Defining size of matrix
```

```
TRANSRHP=zeros(NOFREQ*NOHEAD*NOVEL,7);

for k=1:NOVEL
VEL=fscanf(TRANSinputHP,'%f',1);
SINK=fscanf(TRANSinputHP,'%f',1);
TRIM=fscanf(TRANSinputHP,'%f',1);
XMTN=fscanf(TRANSinputHP,'%f',1);
ZMTN=fscanf(TRANSinputHP,'%f',1);

for j=1:NOHEAD
HEAD=fscanf(TRANSinputHP,'%f',1);

for i=1:NOFREQ
FREQ=fscanf(TRANSinputHP,'%f',1);
trans=fscanf(TRANSinputHP,'%f %f %f',[3,NDOF]);

%Assigning velocity, heading, frequency, and real and
%imaginary part of heave and pith to correct row and
%column in heave/pitch transfer matrix.
TRANSHP((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,1)=VEL/(1852/3600);
TRANSHP((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,2)=HEAD;
TRANSHP((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,3)=FREQ;
TRANSHP((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,4)=trans(2,3);
TRANSHP((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,5)=trans(3,3);
TRANSHP((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,6)=trans(2,5);
TRANSHP((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,7)=trans(3,5);

end

end

end
TRANSHP;

%---------------------------------------------------------
%Reading ShipX file for roll transfer function H=2
%---------------------------------------------------------
info=fscanf(TRANSinputH2,'%c',[1,500]);

%Defining vessel constants
rho=fscanf(TRANSinputH2,'%f',1);
g=fscanf(TRANSinputH2,'%f',1);
Lpp=fscanf(TRANSinputH2,'%f',1);
B=fscanf(TRANSinputH2,'%f',1);
T=fscanf(TRANSinputH2,'%f',1);
LCG=fscanf(TRANSinputH2,'%f',1);
VCG=fscanf(TRANSinputH2,'%f',1);

%Defining number of velocities, headings, frequencies
%and degrees of freedom.
NOVEL=fscanf(TRANSinputH2,'%f',1);
NOHEAD=fscanf(TRANSinputH2,'%f',1);
```

```
NOFREQ=fscanf(TRANSinputH2,'%f',1);
NDOF=fscanf(TRANSinputH2,'%f',1);

%Defining size of matrix
TRANSR2=zeros(NOFREQ*NOHEAD*NOVEL,4);

for k=1:NOVEL
VEL=fscanf(TRANSinputH2,'%f',1);
SINK=fscanf(TRANSinputH2,'%f',1);
TRIM=fscanf(TRANSinputH2,'%f',1);
XMTN=fscanf(TRANSinputH2,'%f',1);
ZMTN=fscanf(TRANSinputH2,'%f',1);

for j=1:NOHEAD
HEAD=fscanf(TRANSinputH2,'%f',1);

for i=1:NOFREQ
FREQ=fscanf(TRANSinputH2,'%f',1);
trans=fscanf(TRANSinputH2,'%f %f %f',[3,NDOF]);

%Assigning velocity, heading, frequency, and real and
%part of roll to correct row and
%column in roll H=2 transfer matrix.
TRANSR2((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,1)=VEL/(1852/3600);
TRANSR2((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,2)=HEAD;
TRANSR2((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,3)=FREQ;
TRANSR2((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,4)=trans(2,4);

end

end

end
TRANSR2;

%-------------------------------------------------------
%Reading ShipX file for roll transfer function H=4
%-------------------------------------------------------
info=fscanf(TRANSinputH4,'%c',[1,500]);

%Defining vessel constants
rho=fscanf(TRANSinputH4,'%f',1);
g=fscanf(TRANSinputH4,'%f',1);
Lpp=fscanf(TRANSinputH4,'%f',1);
B=fscanf(TRANSinputH4,'%f',1);
T=fscanf(TRANSinputH4,'%f',1);
LCG=fscanf(TRANSinputH4,'%f',1);
VCG=fscanf(TRANSinputH4,'%f',1);

%Defining number of velocities, headings, frequencies
%and degrees of freedom.
NOVEL=fscanf(TRANSinputH4,'%f',1);
```

```
NOHEAD=fscanf(TRANSinputH4,'%f',1);
NOFREQ=fscanf(TRANSinputH4,'%f',1);
NDOF=fscanf(TRANSinputH4,'%f',1);

%Defining size of matrix
TRANSR4=zeros(NOFREQ*NOHEAD*NOVEL,4);

for k=1:NOVEL
VEL=fscanf(TRANSinputH4,'%f',1);
SINK=fscanf(TRANSinputH4,'%f',1);
TRIM=fscanf(TRANSinputH4,'%f',1);
XMTN=fscanf(TRANSinputH4,'%f',1);
ZMTN=fscanf(TRANSinputH4,'%f',1);

for j=1:NOHEAD
HEAD=fscanf(TRANSinputH4,'%f',1);

for i=1:NOFREQ
FREQ=fscanf(TRANSinputH4,'%f',1);
trans=fscanf(TRANSinputH4,'%f %f %f',[3,NDOF]);

%Assigning velocity, heading, frequency, and real and
%part of roll to correct row and
%column in roll H=4 transfer matrix.
TRANSR4((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,1)=VEL/(1852/3600);
TRANSR4((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,2)=HEAD;
TRANSR4((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,3)=FREQ;
TRANSR4((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,4)=trans(2,4);

end

end

end
TRANSR4;

%---------------------------------------------------------
%Reading ShipX file for roll transfer function H=6
%---------------------------------------------------------
info=fscanf(TRANSinputH6,'%c',[1,500]);

%Defining vessel constants
rho=fscanf(TRANSinputH6,'%f',1);
g=fscanf(TRANSinputH6,'%f',1);
Lpp=fscanf(TRANSinputH6,'%f',1);
B=fscanf(TRANSinputH6,'%f',1);
T=fscanf(TRANSinputH6,'%f',1);
LCG=fscanf(TRANSinputH6,'%f',1);
VCG=fscanf(TRANSinputH6,'%f',1);

%Defining number of velocities, headings, frequencies
%and degrees of freedom.
```

```
NOVEL=fscanf(TRANSinputH6,'%f',1);
NOHEAD=fscanf(TRANSinputH6,'%f',1);
NOFREQ=fscanf(TRANSinputH6,'%f',1);
NDOF=fscanf(TRANSinputH6,'%f',1);

%Defining size of matrix
TRANSR6=zeros(NOFREQ*NOHEAD*NOVEL,4);

for k=1:NOVEL
VEL=fscanf(TRANSinputH6,'%f',1);
SINK=fscanf(TRANSinputH6,'%f',1);
TRIM=fscanf(TRANSinputH6,'%f',1);
XMTN=fscanf(TRANSinputH6,'%f',1);
ZMTN=fscanf(TRANSinputH6,'%f',1);

for j=1:NOHEAD
HEAD=fscanf(TRANSinputH6,'%f',1);

for i=1:NOFREQ
FREQ=fscanf(TRANSinputH6,'%f',1);
trans=fscanf(TRANSinputH6,'%f %f %f',[3,NDOF]);

%Assigning velocity, heading, frequency, and real and
%part of roll to correct row and
%column in roll H=6 transfer matrix.
TRANSR6((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,1)=VEL/(1852/3600);
TRANSR6((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,2)=HEAD;
TRANSR6((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,3)=FREQ;
TRANSR6((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,4)=trans(2,4);

end

end

end
TRANSR6;

%---------------------------------------------------------
%Reading ShipX file for roll transfer function H=8
%---------------------------------------------------------
info=fscanf(TRANSinputH8,'%c',[1,500]);

%Defining vessel constants
rho=fscanf(TRANSinputH8,'%f',1);
g=fscanf(TRANSinputH8,'%f',1);
Lpp=fscanf(TRANSinputH8,'%f',1);
B=fscanf(TRANSinputH8,'%f',1);
T=fscanf(TRANSinputH8,'%f',1);
LCG=fscanf(TRANSinputH8,'%f',1);
VCG=fscanf(TRANSinputH8,'%f',1);

%Defining number of velocities, headings, frequencies
```

```
%and degrees of freedom.
NOVEL=fscanf(TRANSinputH8,'%f',1);
NOHEAD=fscanf(TRANSinputH8,'%f',1);
NOFREQ=fscanf(TRANSinputH8,'%f',1);
NDOF=fscanf(TRANSinputH8,'%f',1);

%Defining size of matrix
TRANSR8=zeros(NOFREQ*NOHEAD*NOVEL,4);

for k=1:NOVEL
VEL=fscanf(TRANSinputH8,'%f',1);
SINK=fscanf(TRANSinputH8,'%f',1);
TRIM=fscanf(TRANSinputH8,'%f',1);
XMTN=fscanf(TRANSinputH8,'%f',1);
ZMTN=fscanf(TRANSinputH8,'%f',1);

for j=1:NOHEAD
HEAD=fscanf(TRANSinputH8,'%f',1);

for i=1:NOFREQ
FREQ=fscanf(TRANSinputH8,'%f',1);
trans=fscanf(TRANSinputH8,'%f %f %f',[3,NDOF]);

%Assigning velocity, heading, frequency, and real and
%part of roll to correct row and
%column in roll H=8 transfer matrix.
TRANSR8((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,1)=VEL/(1852/3600);
TRANSR8((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,2)=HEAD;
TRANSR8((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,3)=FREQ;
TRANSR8((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,4)=trans(2,4);

end

end

end
TRANSR8;

%-------------------------------------------------------
%Reading ShipX file for roll transfer function H=10
%-------------------------------------------------------
info=fscanf(TRANSinputH10,'%c',[1,500]);

%Defining vessel constants
rho=fscanf(TRANSinputH10,'%f',1);
g=fscanf(TRANSinputH10,'%f',1);
Lpp=fscanf(TRANSinputH10,'%f',1);
B=fscanf(TRANSinputH10,'%f',1);
T=fscanf(TRANSinputH10,'%f',1);
LCG=fscanf(TRANSinputH10,'%f',1);
VCG=fscanf(TRANSinputH10,'%f',1);
```

```
%Defining number of velocities, headings, frequencies
%and degrees of freedom.
NOVEL=fscanf(TRANSinputH10,'%f',1);
NOHEAD=fscanf(TRANSinputH10,'%f',1);
NOFREQ=fscanf(TRANSinputH10,'%f',1);
NDOF=fscanf(TRANSinputH10,'%f',1);

%Defining size of matrix
TRANSR10=zeros(NOFREQ*NOHEAD*NOVEL,4);

for k=1:NOVEL
VEL=fscanf(TRANSinputH10,'%f',1);
SINK=fscanf(TRANSinputH10,'%f',1);
TRIM=fscanf(TRANSinputH10,'%f',1);
XMTN=fscanf(TRANSinputH10,'%f',1);
ZMTN=fscanf(TRANSinputH10,'%f',1);

for j=1:NOHEAD
HEAD=fscanf(TRANSinputH10,'%f',1);

for i=1:NOFREQ
FREQ=fscanf(TRANSinputH10,'%f',1);
trans=fscanf(TRANSinputH10,'%f %f %f',[3,NDOF]);

%Assigning velocity, heading, frequency, and real and
%part of roll to correct row and
%column in roll H=10 transfer matrix.
TRANSR10((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,1)=VEL/(1852/3600);
TRANSR10((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,2)=HEAD;
TRANSR10((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,3)=FREQ;
TRANSR10((k-1)*NOHEAD*NOFREQ+(j-1)*NOFREQ+i,4)=trans(2,4);

end

end

end
TRANSR10;


end
```

A.6. gcfuel

```
%-------------------------------------------------------------
%               gcfuel.m
%               Hege Eskild
%               Spring 2014
%
% In this function the fuel consumption for the great circle
% route (GCR) is calculated. The GRC in this program is
% is defined as a practical GCR, meaning that the route
% is constructed of three straight legs and not a circle.
% The output of this function is the fuel consumption and
% two vectors with lat and lon later used to construct the
% feasible region for the optimal route.
%
%------------------------------------------------------------
%   Variable name                  Definition
%----------------------Input-----------------------------
%   GClat                          Four latitude values for the
%                                  practical GCR
%   GClon                          Four longitude values for the
%                                  practical GCR
%   Pb                             Desired brake power
%   bsfc                           Specific fuel consumption (g/kWh)
%   AVspeed                        Matrix with attainable speed for
%                                  different sea states
%                                  for different headings
%   lon                            Vector with longitude values from
%                                  GRIB
%   lat                            Vector with latitude values from
%                                  GRIB
%   SWH                            Matrix with Hs from GRIB
%   MWD                            Matrix with mean wave direction
%                                  from GRIB
%   MWP                            Matrix with mean wave period
%                                  from GRIB
%---------------------Output--------------------------------
%   GCFC                           Fuel consumption for GCR
%   GClats                         Latitude vector for GCR
%   GClons                         Longitude vector for GCR
%---------------------Internal-------------------------------
%   i                              Counting integer
%   k                              Counting integer
%   Hs                             Significant wave height for
%                                  the given arc
%   T1                             Mean wave period for the
%                                  given arc
%   Tp                             Peak wave period for the
%                                  given arc
%   head                           Relative heading between wave
%                                  and vessel or given arc
%   arcTimeS                       Vector with traveled time for
%                                  all arc
%   arcFuelS                       Vector with fuel consumption for
%                                  all arcs

function [GCFC,GClats,GClons]=gcfuel(GClat,GClon,Pb,bsfc,...
          AVspeed,lon,lat,SWH,MWD,MWP);

%There are three legs in the practical GCR. The for-loop
%assigns lat and lon values to the points in-between.
for i=1:3
```

```
    [arclen1(i,1),az1(i,1)] = distance('rh',GClat(i),...
                              GClon(i),GClat(i+1),GClon(i+1));
    [lat1,lon1] = track2('rh',GClat(i),GClon(i),GClat(i+1),...
                  GClon(i+1),referenceSphere('unit sphere'),...
                  'degrees',round(arclen1(1,1)));
    lats1(:,i)=lat1;
    lons1(:,i)=lon1;
end

%Creating on vector for lon and one for lat for the entire route
for i=1:3*length(lats1)-2
    if i<length(lats1)
    lats(i,1)=lats1(i,1);
    lons(i,1)=lons1(i,1);
    elseif i<2*length(lats1)
        lats(i,1)=lats1(i-length(lats1)+1,2);
        lons(i,1)=lons1(i-length(lats1)+1,2);
    else
        lats(i,1)=lats1(i-2*length(lats1)+2,3);
        lons(i,1)=lons1(i-2*length(lats1)+2,3);
    end
end
lats;
lons;

%Calculating the fuel consumption for each arc
for i=1:length(lats)-1
    %Defining arc lenghts for going straight
    [arcStraight(i,1),azS(i,1)]=distance...
    ('gc',lats(i,1),lons(i,1),lats(i+1,1),lons(i+1,1));
    %Finding the index to the latitude and longitude weather vector
    [~, idxlat] = min(abs(lat - lats(i+1,1)));
    [~, idxlon] = min(abs(lon - lons(i+1,1)));
    az = azimuth(lats(i,1),lons(i,1),lats(i+1,1),lons(i+1,1));
    %Finding Hs, T1 and direction for the given arc
    idxtime=round(i/2);
    Hs=SWH(idxtime,idxlat,idxlon);
    if Hs<1 || isnan(Hs)
        Hs=1;
    end
    T1=MWP(idxtime,idxlat,idxlon);
    if T1<1 || isnan(T1)
        T1=5.4;
    end
    Tp=T1*1.3;
    dir=MWD(idxtime,idxlat,idxlon);
    if dir<1 || isnan(dir)
        dir=90;
    end
    %Finding relative heading between wave and vessel
    head=sqrt(az^2+dir^2-2*az*dir);
```

```
        if head>180
            head=360-head;
        end
    headS(i,1)=head;
    TpS(i,1)=Tp;
    HsS(i,1)=Hs;

    %Assigning the speed to the arc for the given weather
    %condition
    for k=1:length(AVspeed)
        if AVspeed(k,1)>(head-15) && AVspeed(k,1)<(head+15)...
           && AVspeed(k,2)>(Hs-0.5) && AVspeed(k,2)<(Hs+0.5)...
           && AVspeed(k,3)>(Tp-0.5) && AVspeed(k,3)<(Tp+0.5)
              V=AVspeed(k,4);
        end
    end
    VS(i,1)=V;
    Vcalm=23.8;
    %Assiging the fuel cost for the given arc
    arcTimeS(i,1)=deg2nm(arcStraight(i,1))/V;
    arcFuelS(i,1)=Pb*arcTimeS(i,1)*bsfc/1000000;
    arcTimeCalm(i,1)=deg2nm(arcStraight(i,1))/Vcalm;
    arcFuelCalm(i,1)=Pb*arcTimeCalm(i,1)*bsfc/1000000;

end
GCFC=sum(arcFuelS)
tottime=sum(arcTimeS);
CalmWaterFC=sum(arcFuelCalm);

GClats=lats;
GClons=lons;

end
```

A.7. optfuel.m

```
%------------------------------------------------------------
%                optfuel.m
%                Hege Eskild
%                Spring 2014
%
% This is the optimal fuel function for estimated wind. The
% setup in the code is described in chapter 3 in the thesis.
%
%------------------------------------------------------------
%   Variable name                Definition
%----------------------Input-----------------------------
%   startLat                     Latitude coordinate departure
%   startLon                     Longitude coordinate departure
%   endLat                       Latitude coordinate arrival
%   endLon                       Longitude coordinate arrival
%   ProbSlamming                 Seakeeping criterion for slamming
%   ProbGreenWater                 Seakeeping criterion for deck
%                                  wetness
%   VAccFP                         Seakeeping criterion for vertical
%                                  acceleration
%   RMSRoll                        Seakeeping criterion for roll
%   NumCenPts                      Number of center points for opt
%                                  route
%   DeltaNM                        Distance in NM between center
%                                  points
%                                  practical GCR
%   Pb                             Desired brake power
%   bsfc                           Specific fuel consumption (g/kWh)
%   Xp                             X coordinate of FP (negative from
%                                  CG)
%   freeboard                      freeboard at FP
%   AVspeed                        Matrix with attainable speed for
%                                  different sea states
%   NOHEAD                         Number of different headings in
%                                  Avspeed
%   lon                            Vector with longitude values from
%                                  GRIB
%   lat                            Vector with latitude values from
%                                  GRIB
%   time                           Vector with time values from GRIB
%   SWH                            Matrix with Hs from GRIB
%   MWD                            Matrix with mean wave direction
%                                  from GRIB
%   MWP                            Matrix with mean wave period
%                                  from GRIB
%   TRANS(..)                      Matrices including transfer
%                                  functions for roll and heave/
%                                  pitch for different frequencies
%                                  heading and velocity.
%   NOHEAD                         Number of heading in transfer
%                                  file
%   NOFREQ                         Number of frequencies in
%                                  transfer files
%   Lpp                            Length per perpendiculars
%   GCFC                           Fuel consumption for GCR
%   GClats                         Latitude vector for GCR
%   GClons                         Longitude vector for GCR
%---------------------Output--------------------------------
%   FC                             Fuel consumption for opt. route
%---------------------Internal--------------------------------
%   i                              Counting integer
%   j                              Counting integer
%   k                              Counting integer
%   latC                           Vector for latitude values of
%                                  center points
%   lonC                           Vector for longitude values of
%                                  center points
%   lats                           Matrix with all latitude values in
%                                   the feasible region
%    lons                           Matrix with all longitude values in
%                                   the feasible region
%    Hs                             Significant wave height for
%                                   the given arc
%    T1                             Mean wave period for the
%                                   given arc
%    Tp                             Peak wave period for the
%                                   given arc
%    head                           Relative heading between wave
%                                   and vessel or given arc
%    V                              Ship's speed
%    arcTime(S/R/L)                 Matrix with traveled time for
%                                   all arcs. One for straight, left
  and
%                                   right
%    arcFuel(S/R/L)                 Matrix with fuel consumption for
%                                   all arcs One for straight, left and
%                                   right
%    from(S/R/L)                    Departure nodes in sparse matrix
%    to(S/R/L)                      Arrival nodes in sparse matrix
%    fuel(S/R/L)                    Fuel cost between from and to


function [FC]=optfuel(startLat,startLon,endLat,endLon,NumCenPts,...
             Pb,DeltaNM,bsfc,AVspeed,lon,lat,time,Uwind,Vwind,...
             SWH,MWD,MWP,ProbSlamming,ProbGreenWater,VAccFP,...
             RMSRoll,TRANSR2,TRANSR4,TRANSR6,TRANSR8,TRANSR10,...
             TRANSHP,NOFREQ,NOHEAD,Lpp,Xp,freeboard,GClats,GClons)
               ;



%
  --------------------------------------------------------------------

%Create the network of nodes. Ref step 1 in fig. 3.4 in thesis
%------------------------------------------------------------------
[GClat,GClon]=gcwaypts(startLat,startLon,endLat,endLon,50);
[RHlat,RHlon]=track2('rh',startLat,startLon,endLat,endLon,...
             referenceSphere('unit sphere'),'degrees',51);
%Finding the center point in the route, step 1 in algorithm 3.2 in
%the thesis
CenterLat=GClat(26);
CenterLon=GClon(26);
%Finding the course between center point, and one before
%center point, step 2 in algorithm 3.2 in thesis
Course=azimuth('rh',GClat(25),GClon(25),CenterLat,CenterLon);

%Calculating point along perpendicular to center point, step 4
%in algorithm 3.2 in thesis
%Points above center line
```

```
for i=1:(NumCenPts-1)/2
    [latCenterN(i,1),lonCenterN(i,1)]=reckon('rh',CenterLat,...
    CenterLon,i*nm2deg(DeltaNM),(Course-90));
end
%Points below centerline
for i=1:(NumCenPts-1)/2
    [latCenterS(i,1),lonCenterS(i,1)]=reckon('rh',CenterLat,...
    CenterLon,i*nm2deg(DeltaNM),(Course+90));
end

%Creating one vector with center points for both above
%and under the line
for i=1:NumCenPts
    if i<(NumCenPts+1)/2
        latC(i,1)=latCenterN((NumCenPts+1)/2-i,1);
        lonC(i,1)=lonCenterN((NumCenPts+1)/2-i,1);
    elseif i==(NumCenPts+1)/2
        latC(i,1)=CenterLat;
        lonC(i,1)=CenterLon;
    else
        latC(i,1)=latCenterS(i-(NumCenPts+1)/2,1);
        lonC(i,1)=lonCenterS(i-(NumCenPts+1)/2,1);
    end
end

latC;
lonC;

%Defining point along the rhumb lines between start and center
%points and center points and end. Step 6 in algorithm 3.2
%in thesis.
for i=1:NumCenPts
    %points for first half
    [arclen1(i,1),az1(i,1)] = distance('rh',startLat,startLon,...
    latC(i),lonC(i));
    [lat1,lon1] = track2('rh',startLat,startLon,latC(i),lonC(i),...
    referenceSphere('unit sphere'),'degrees',round(arclen1(1,1)));
    lats1(:,i)=lat1;
    lons1(:,i)=lon1;

    %points for second half
    [arclen2(i,1),az2(i,1)] = distance('rh',latC(i),lonC(i),...
    endLat,endLon);
    [lat2,lon2] = track2('rh',latC(i),lonC(i),endLat,endLon,...
    referenceSphere('unit sphere'),'degrees',round(arclen1(1,1)));
    lats2(:,i)=lat2;
    lons2(:,i)=lon2;
end

%Creating two matrices with all nodes in the feasible region.
%Where the column represents the number of center points
%and the rows are the stages in the voyage, i.e. the same time
```

```
%step. Ref. fig 3.8 in the thesis.
s=length(lats1(:,1));

for i=1:(2*s-1)
    if i<s
        lats(i,:)=lats1(i,:);
        lons(i,:)=lons1(i,:);
    else
        lats(i,:)=lats2(i-s+1,:);
        lons(i,:)=lons2(i-s+1,:);
    end
end
lats;
lons;

%----------------------------------------------------------------
%Defining arcs between nodes. Three matrices created, one
%for choosing the node to the left, one for straight, and
%for for choosing the node to the right. The definitions
%of right and left is based on how the lats and lons vectors
%are set up. The actual direction is dependent on the sailing
%rute.

%Second part is to add a cost to the given arc. This cost is
%the fuel consumption which is calculated by the weather and
%available speed. If the arc does not satisfy the seakeeping
%criteria the fuel consumption is set to 1000 so that this arc
%will not be chosen.

%This part of optfuel refers to the second and third step
%in fig. 3.4 in the thesis

for i=1:length(lats)-1
    for j=1:NumCenPts
        %------------------------------------------------------
        %Defining arcs and cost for going straight
        %------------------------------------------------------
        [arcStraight(i,j),azS(i,j)]=distance('gc',lats(i,j),...
        lons(i,j),lats(i+1,j),lons(i+1,j));
        %Finding the index to the latitude and longitude
        %weather vector
        [~, idxlat] = min(abs(lat - lats(i+1,j)));
        [~, idxlon] = min(abs(lon - lons(i+1,j)));
        az = azimuth(lats(i,j),lons(i,j),lats(i+1,j),lons(i+1,j));
        idxtime=round(i/2);
        %Finding Hs, T1 and direction for the given arc
        %Hs, T1 and dir is assigned value if NaN
        Hs=SWH(idxtime,idxlat,idxlon);
        if Hs<1 || isnan(Hs)
            Hs=1;
        end
        T1=MWP(idxtime,idxlat,idxlon);
```

```
if isnan(T1)
    T1=5.4;
end
Tp=T1*1.3;
dir=MWD(idxtime,idxlat,idxlon);
if isnan(dir)
    dir=90;
end
%Finding relative heading between wave and vessel
head=sqrt(az^2+dir^2-2*az*dir);
    if head>180
        head=360-head;
    end
headS(i,j)=head;
TpS(i,j)=Tp;
HsS(i,j)=Hs;
dirS(i,j)=dir;

%Assigning the speed to the arc for the given weather
%condition
for k=1:length(AVspeed)
    if AVspeed(k,1)>(head-15) && AVspeed(k,1)<(head+15)...
       && AVspeed(k,2)>(Hs-0.5) && AVspeed(k,2)<(Hs+0.5)...
       && AVspeed(k,3)>(Tp-0.5) && AVspeed(k,3)<(Tp+0.5)
          V=AVspeed(k,4);
    end
end
VS(i,j)=V;

%Assiging the fuel cost for the given arc
arcTimeS(i,j)=deg2nm(arcStraight(i,j))/V;
arcFuelS(i,j)=Pb*arcTimeS(i,j)*bsfc/1000000;

%Seakeeping function, checking criteria
[RouteOK,ProbSlam,ProbGW,RMSRelVertAcc,RMSroll]=...
seakeeping(T1,Hs,V,head,ProbSlamming,ProbGreenWater,...
VAccFP,RMSRoll,TRANSR2,TRANSR4,TRANSR6,TRANSR8,...
TRANSR10,TRANSHP,NOFREQ,NOHEAD,Lpp,Xp,freeboard);
%Giving the arcFuel a value of 1000 if the routeOK
%is not satisfied
if RouteOK==0
    arcFuelS(i,j)=1000;
end
ProbSlamS(i,j)=ProbSlam;
ProbGWS(i,j)=ProbGW;
RMSRelVertAccS(i,j)=RMSRelVertAcc;
RMSrollS(i,j)=RMSroll;


%--------------------------------------------------------
%Defining arcs and cost for going right
%-------------------------------------------------------
```

```
if j<NumCenPts
[arcRight(i,j),azR(i,j)]=distance('gc',lats(i,j),lons(i,j)
  ,...
lats(i+1,j+1),lons(i+1,j+1));
%Finding the index to the latitude and longitude weather
%vector
[~, idxlat] = min(abs(lat - lats(i+1,j+1)));
[~, idxlon] = min(abs(lon - lons(i+1,j+1)));
az = azimuth(lats(i,j),lons(i,j),lats(i+1,j+1),lons(i+1,j
  +1));
idxtime=round(i/2);
%Finding Hs, T1 and direction for the given arc
Hs=SWH(idxtime,idxlat,idxlon);
if Hs<1 || isnan(Hs)
    Hs=1;
end
T1=MWP(idxtime,idxlat,idxlon);
if isnan(T1)
    T1=5.4;
end
Tp=T1*1.3;
dir=MWD(idxtime,idxlat,idxlon);
if isnan(dir)
    dir=90;
end
%Finding relative heading between wave and vessel
head=sqrt(az^2+dir^2-2*az*dir);
    if head>180
        head=360-head;
    end
headR(i,j)=head;
TpR(i,j)=Tp;
HsR(i,j)=Hs;

%Assigning the speed to the arc for the given weather
%condition
for k=1:length(AVspeed)
    if AVspeed(k,1)>(head-15) && AVspeed(k,1)<(head+15)...
       && AVspeed(k,2)>(Hs-0.5) && AVspeed(k,2)<(Hs+0.5)...
       && AVspeed(k,3)>(Tp-0.5) && AVspeed(k,3)<(Tp+0.5)
          V=AVspeed(k,4);
    end
end

%Assiging the fuel cost for the given arc
arcTimeR(i,j)=deg2nm(arcRight(i,j))/V;
arcFuelR(i,j)=Pb*arcTimeR(i,j)*bsfc/1000000;

%Seakeeping function, checking criteria
[RouteOK,ProbSlam,ProbGW,RMSRelVertAcc,RMSroll]=...
seakeeping(T1,Hs,V,head,ProbSlamming,ProbGreenWater,...
VAccFP,RMSRoll,TRANSR2,TRANSR4,TRANSR6,TRANSR8,...
TRANSR10,TRANSHP,NOFREQ,NOHEAD,Lpp,Xp,freeboard);
%Giving the arcFuel a value of 1000 if the routeOK is
%not satisfied
if RouteOK==0
    arcFuelR(i,j)=1000;
end
ProbSlamR(i,j)=ProbSlam;
ProbGWR(i,j)=ProbGW;
RMSRelVertAccR(i,j)=RMSRelVertAcc;
RMSrollR(i,j)=RMSroll;
end

if j>1
%-------------------------------------------------------
%Defining arcs and costs for going right
%------------------------------------------------------
[arcLeft(i,j),azL(i,j)]=distance('gc',lats(i,j),lons(i,j)
  ,...
lats(i+1,j-1),lons(i+1,j-1));
%Finding the index to the latitude and longitude weather
  vector
[~, idxlat] = min(abs(lat - lats(i+1,j-1)));
[~, idxlon] = min(abs(lon - lons(i+1,j-1)));
az = azimuth(lats(i,j),lons(i,j),lats(i+1,j-1),lons(i+1,j
  -1));
idxtime=round(i/2);
%Finding Hs, T1 and direction for the given arc
Hs=SWH(idxtime,idxlat,idxlon);
if Hs<1 || isnan(Hs)
    Hs=1;
end
T1=MWP(idxtime,idxlat,idxlon);
if isnan(T1)
    T1=5.4;
end
Tp=T1*1.3;
dir=MWD(idxtime,idxlat,idxlon);
if isnan(dir)
    dir=90;
end
%Finding relative heading between wave and vessel
head=sqrt(az^2+dir^2-2*az*dir);
    if head>180
        head=360-head;
    end
headL(i,j)=head;
TpL(i,j)=Tp;
HsL(i,j)=Hs;

%Assigning the speed to the arc for the given weather
%condition
for k=1:length(AVspeed)
```

```
                if AVspeed(k,1)>(head-15) && AVspeed(k,1)<(head+15)...
                   && AVspeed(k,2)>(Hs-0.5) && AVspeed(k,2)<(Hs+0.5)...
                   && AVspeed(k,3)>(Tp-0.5) && AVspeed(k,3)<(Tp+0.5)
                      V=AVspeed(k,4);
                end
            end

            %Assiging the fuel cost for the given arc
            arcTimeL(i,j)=deg2nm(arcLeft(i,j))/V;
            arcFuelL(i,j)=Pb*arcTimeL(i,j)*bsfc/1000000;

            %Seakeeping function, checking criteria
            [RouteOK,ProbSlam,ProbGW,RMSRelVertAcc,RMSroll]=...
            seakeeping(T1,Hs,V,head,ProbSlamming,ProbGreenWater,...
            VAccFP,RMSRoll,TRANSR2,TRANSR4,TRANSR6,TRANSR8,...
            TRANSR10,TRANSHP,NOFREQ,NOHEAD,Lpp,Xp,freeboard);
            %Giving the arcFuel a value of 1000 if the routeOK
            %is not satisfied
            if RouteOK==0
                arcFuelL(i,j)=1000;
            end
            ProbSlamL(i,j)=ProbSlam;
            ProbGWL(i,j)=ProbGW;
            RMSRelVertAccL(i,j)=RMSRelVertAcc;
            RMSrollL(i,j)=RMSroll;
            end
        end
end

arcFuelS;
arcFuelR;
arcFuelL(:,1)=[];

%Next section is used to verify the different arcs and their
%calculated values
%Matrices corresponding to the straight arcs:
arcStraight;
headS ;
HsS;
TpS;
dirS;
VS;

%Optinal to write values to excel spreadsheet
%xlswrite('optfuel.xls',HsS,'HS')
%xlswrite('optfuel.xls',TpS,'Tp')
%xlswrite('optfuel.xls',dirS,'dir')
%xlswrite('optfuel.xls',headS,'head')
%xlswrite('optfuel.xls',VS,'V')

%Responses
ProbSlamS;
```

```
ProbGWS;
RMSRelVertAccS;
RMSrollS;

% xlswrite('optfuel.xls',ProbSlamS,'Slam')
% xlswrite('optfuel.xls',ProbGWS,'GW')
% xlswrite('optfuel.xls',RMSRelVertAccS,'ACC')
% xlswrite('optfuel.xls',RMSrollS,'roll')

%Matrices coresponding to the right arcs:
arcRight;
headR;
HsR;
TpR;
%responses
ProbSlamR;
ProbGWR;
RMSRelVertAccR;
RMSrollR;

%Matrices corresponding to left arcs
arcLeft;
headL;
HsL;
TpL;
%Responses
ProbSlamL;
ProbGWL;
RMSRelVertAccL;
RMSrollL;

%--------------------------------------------------------------
%Optimization. The last part of the optfuel function refers
%to the fourth step in fig. 3.4. To use Dijstras algorithm
%in Matlab a sparse matrix must be defined. This sparse matrix
%includes a vector with nodes that are departed (from) and nodes
%that are arrival nodes (to). The sparse matrix also includes a
%vector of the cost of going from to. This part of the function is
%difficult to understand, mostly trixing with matrices and vectors
%See fig 3.8 in thesis to understand the setup.
%--------------------------------------------------------------
%Assigns an actual number to each node
for i=1:length(lats)-1
    for j=1:NumCenPts
        nodeS(i,j)=j+1+(i-1)*NumCenPts;
    end
end
nodeS;
nodeR=nodeS;
nodeR(:,1)=[];
nodeL=nodeS;
nodeL(:,NumCenPts)=[];
```

```
SizeNodes=size(nodeS);
NumberNodes=SizeNodes(1,1)*SizeNodes(1,2);

%Node connection
%Define from nodes the straight points
for i=1:((length(lats(:,1))-1)*NumCenPts)
    if i<NumCenPts+1;
        fromS(i)=1;
    else
    fromS(i)=i-NumCenPts+1;
    end
end

%Defining to nodes for straight arcs and cost
for i=1:((length(lats(:,1))-1)*NumCenPts)
    if i<NumberNodes-NumCenPts+1
        toS(i)=1+i;
        fuelS(i)=arcFuelS(nodeS==(i+1));
    else
        toS(i)=((length(lats(:,1)))-2)*NumCenPts+2;
        fuelS(i)=arcFuelS(nodeS==(i+1));
    end
end

%Define the right arcs
for j=1:((length(lats(:,1))-2)*(NumCenPts-1))
    fromR(j)=j+1;
end

for j=1:((length(lats(:,1))-2)*(NumCenPts-1))
    if j<((length(lats(:,1))-2)*(NumCenPts))-NumCenPts+1
        toR(j)=NumCenPts+j+1;
    else
        toR(j)=((length(lats(:,1)))-2)*NumCenPts+2;
    end

end

%Editing the to and from nodes and cost for
%right and left arcs
fromR;
toR;
fromL=fromR;
toL=toR;
fromR(NumCenPts:NumCenPts:end)=[];
toR(1:NumCenPts:end)=[];
fromL(1:NumCenPts:end)=[];
toL(NumCenPts:NumCenPts:end)=[];

%Assigning fuel cost for right and left arcs
for i=1:length(toR)
```

```
    fuelR(i)=arcFuelR(nodeR==toR(i));
    fuelL(i)=arcFuelL(nodeL==toL(i));
end

%Defining the final to, from and cost vectors to be
%used in the sparse matrix.
for i =1:length(fuelS)+length(fuelR)+length(fuelL)
    if i<length(fuelS)+1
        fuel(i)=fuelS(i);
        from(i)=fromS(i);
        to(i)=toS(i);
    elseif i<length(fuelS)+length(fuelR)+1
        fuel(i)=fuelR(i-length(fuelS));
        from(i)=fromR(i-length(fuelS));
        to(i)=toR(i-length(fuelS));
    else
        fuel(i)=fuelL(i-length(fuelS)-length(fuelR));
        from(i)=fromL(i-length(fuelS)-length(fuelR));
        to(i)=toL(i-length(fuelS)-length(fuelR));
    end
end
%Defining a sparse to use in shortest path function
SparseGraph = sparse(from,to,fuel);
MatrixGraph = full(SparseGraph);
%Adding an additional row to the full matrix so that the matrix
%is symetrical. The added row is defined as an edge.
MatrixGraph((length(lats)-2)*NumCenPts+2,:)=0;
%Converting matrix back to sparse
SparseGraph=sparse(MatrixGraph);

%Calculating the fuel consumption and path for the
%optimization problem with Dijkstra's algorithm
[FC,path,pred] = graphshortestpath(SparseGraph,1,...
(length(lats)-2)*NumCenPts+2,'Method','Dijkstra');

FC;
path;

%Plotting the graph of nodes and connecting arcs
h = view(biograph(SparseGraph,[],'ShowWeights','on'))
set(h.Nodes(path),'Color',[1 0.4 0.4])
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)

%Assigning geographical coordinates to the nodes in order
%to plot in map

for i=2:length(lats)-1
   %locating index for nodes via path
   [~,indnodeS]=find(nodeS(i-1,:)==path(i));
   OptFuelPathLat(i,1)=lats(i,indnodeS);
   OptFuelPathLon(i,1)=lons(i,indnodeS);
end
OptFuelPathLat(1,1)=startLat;
OptFuelPathLon(1,1)=startLon;
OptFuelPathLat(length(lats),1)=endLat;
OptFuelPathLon(length(lats),1)=endLon;

%Map plot with nodes and route

S=8;
figure (2)
hold on
geoshow('landareas.shp', 'FaceColor', [0.5 1.0 0.5]);
hold on
geoshow(GClats,GClons)
hold on
geoshow(OptFuelPathLat,OptFuelPathLon,'displaytype','line','color',
  'r')

for i=1:NumCenPts
hold on
scatter(lons(:,i),lats(:,i),S,'fill','b')
end

hold on
scatter(lonC,latC,S,'fill','r')

end
```

A.8. seakeeping.m

```
%-------------------------------------------------------------
%                seakeeping.m
%                Hege Eskild
%                Spring 2014
%
% The seakeeping function calculates the standard deviation
% in roll and vertical acceleration, as well as the probability
% for slamming and deck wetness. The main output of this
% function is a routeOK variable defining whether or not the
% given arc satisfies the seakeeping criteria. Output from this
% function is also the Seakeeping values used to evaluate the
% entire route and the feasible region.
%
%-------------------------------------------------------------
%   Variable name                   Definition
%----------------------Input-----------------------------
%   T1                              Mean wave period
%   Hs                              Significant wave height
%   V                               Vessel speed
%   head                            relative heading between waves and
%                                   vessel
%   ProbSlamming                 Seakeeping criterion for slamming
%   ProbGreenWater               Seakeeping criterion for deck
%                                wetness
%   VAccFP                       Seakeeping criterion for vertical
%                                acceleration
%   RMSRoll                      Seakeeping criterion for roll

%   TRANS(..)                    Matrices including transfer
%                                functions for roll and heave/
%                                pitch for different frequencies
%                                heading and velocity.
%   NOHEAD                       Number of heading in transfer
%                                file
%   NOFREQ                       Number of frequencies in
%                                transfer files
%   Lpp                          Length per perpendiculars
%   Xp                           X coordinate of FP (negative from
%                                CG)
%   freeboard                    freeboard at FP
%                                deduction
%--------------------Output------------------------------
%   RouteOK                      Defines whether or not the
  seakeeping
%                                criteria are satisfied
%   ProbSlam                     Probability of slamming for arc
%   probGW                       Probability of green water for arc
%   RMSRelVertAcc                Standard deviation for relative
%                                vertical acceleration for arc
%   RMSroll                      Standard deviation for roll for
%                                arc
%--------------------Internal----------------------------
%   Tp                           Peak period
%   omegap                       Peak frequency
%   omega                        Frequency from ShipX file
%   m0,1,2,4Roll                 Vectors for calculating
%                                spectral moments for roll
%   m0,1,2,4RelVert              Vectors for calculating
%                                spectral moments for relative
%                                vertical motions
%   sigma                        Spectral width parameter
%   gamma                        Peakedness parameter


function [RouteOK,ProbSlam,ProbGW,RMSRelVertAcc,RMSroll]=seakeeping
  ...
    (T1,Hs,V,head,ProbSlamming,ProbGreenWater,VAccFP,...
    RMSRoll,TRANSR2,TRANSR4,TRANSR6,TRANSR8,TRANSR10,TRANSHP,NOFREQ
      ,...
    NOHEAD,Lpp,Xp,freeboard)

%
  ------------------------------------------------------------------
```

```
%defining wariables for wave spectrum
ZETAa=Hs/2;
gamma=1;
Tp=1.3*T1;
omegap=2*pi()/Tp;
V=V+0.0001;

%-------------------------------------------------------------

%First part calculates RMS for roll

%Choosing the correct transfer file from ShipX
if Hs<3
    TRANSROLL=TRANSR2;
elseif Hs<5
    TRANSROLL=TRANSR4;
elseif Hs<7
    TRANSROLL=TRANSR6;
elseif Hs<9
    TRANSROLL=TRANSR8;
else
    TRANSROLL=TRANSR10;
end

ROLL=zeros(NOFREQ,1);
m0ROLL=zeros(NOFREQ,1);
m1ROLL=zeros(NOFREQ,1);
m2ROLL=zeros(NOFREQ,1);
m4ROLL=zeros(NOFREQ,1);

%Defining step in velocity and heading from transfer
deltaV=(TRANSROLL(NOFREQ*NOHEAD+1,1)-TRANSROLL(1,1))/2;
deltaHEAD=(TRANSROLL(NOFREQ+1,2)-TRANSROLL(1,2))/2;

%finding minimum and maximum speed in trans file
minV=TRANSROLL(1,1);
maxV=TRANSROLL(length(TRANSROLL),1);
if V>maxV
    V=maxV-0.001;
elseif V<minV
    V=minV+0.001;
end


ind=zeros(NOFREQ,1);
%Locating correct transfer function index
for i=1:length(TRANSROLL)
   if (TRANSROLL(i,1)>(V-deltaV) && TRANSROLL(i,1)<(V+deltaV) &&...
      TRANSROLL(i,2)>(head-deltaHEAD) && TRANSROLL(i,2)<(head+
        deltaHEAD))
      ind(i)=i;
   end
end
ind(ind==0)=[];
ind;

%Calculating the motion spectrum with given heading, Hs and Tp.
%Using the JONSWAP spectrum formulation
for j=1:NOFREQ
    i=j+ind(1)-1;
    RAO=TRANSROLL(i,4);
    omegaR=TRANSROLL(i,3);
    omegae=omegaR-omegaR^2*V/9.81*cos(head/180*pi());
    if omegaR<omegap
        sigma=0.07;
    else
        sigma=0.09;
    end
    Y=-0.5*((omegaR-omegap)/(sigma*omegap))^2;
    sROLL=RAO^2*5.061*Hs^2/(Tp^4)*(1-0.287*log(gamma))...
        *9.81^2*omegaR^(-5)*...
        exp(-1.25*(omegap/omegaR)^4)*gamma^exp(Y);


m0ROLL(j)=sROLL;
m1ROLL(j)=omegaR*sROLL;
m2ROLL(j)=omegaR^2*sROLL;
m4ROLL(j)=omegaR^4*sROLL;

OmegaR(j,1)=omegaR;

end
OmegaR;
m0ROLL;
h=(TRANSROLL(NOFREQ,3)-TRANSROLL(1,3))/NOFREQ;
%Calculating spectral moment M0 for roll
M0roll=trapz(OmegaR,m0ROLL);

RMSroll=sqrt(M0roll)*180/pi();



%
  -------------------------------------------------------------------------


%Third part calculates RMS Vertical acceleration at bow

%Define X and Y Position for calculating acceleration,
%probability of slamming and probability of green water on deck.
Yp=0;
```

```
RelVert=zeros(NOFREQ,1);
m0RelVert=zeros(NOFREQ,1);
m1RelVert=zeros(NOFREQ,1);
m2RelVert=zeros(NOFREQ,1);
m4RelVert=zeros(NOFREQ,1);

%Defining step in velocity and heading from transfer
deltaV=(TRANSHP(NOFREQ*NOHEAD+1,1)-TRANSHP(1,1))/2;
deltaHEAD=(TRANSHP(NOFREQ+1,2)-TRANSHP(1,2))/2;

ind=zeros(NOFREQ,1);


%Locating correct transfer function index
for i=1:length(TRANSHP)
   if (TRANSHP(i,1)>(V-deltaV) && TRANSHP(i,1)<(V+deltaV) &&...
      TRANSHP(i,2)>(head-deltaHEAD) && TRANSHP(i,2)<(head+deltaHEAD
       ))
      ind(i)=i;
   end
end
ind(ind==0)=[];
ind;

%Calculating the relative at FP motion spectrum with given
%heading, Hs and Tp. Using the JONSWAP spectrum formulation
for j=1:NOFREQ
     i=j+ind(1)-1;
     omega=TRANSHP(i,3);
     %Defining real and imaginary part of heave and pitch
     heaveR=TRANSHP(i,4);
     heaveIm=TRANSHP(i,5);
     pitchR=TRANSHP(i,6);
     pitchIm=TRANSHP(i,7);

     %Defining phase angle for heave and pitch
     phaseHeave=atan(heaveIm/heaveR);
     phasePitch=atan(pitchIm/pitchR);

     %Calculating the three parts of the RAO
     RAO1=sqrt(heaveR^2+heaveIm^2)*exp(1i*phaseHeave);
     RAO2=Xp*sqrt(pitchR^2+pitchIm^2)*exp(1i*phasePitch);
     RAO3=exp(-1i*omega^2/9.81*(Xp*cos(head/180*pi())+...
          Yp*sin(head/180*pi())));

     RAO4=RAO1-RAO2-RAO3;
     %RAO used for calculations
     RAO=abs(RAO4);

         if j<NOFREQ
     Deltaomega=TRANSHP(j+1,3)-TRANSHP(j,3);
     else
```

```
    Deltaomega=TRANSHP(j,3)-TRANSHP(j-1,3);
    end

    if omega<omegap
        sigma=0.07;
    else
        sigma=0.09;
    end
    Y=-0.5*((omega-omegap)/(sigma*omegap))^2;
    sRelVert=RAO^2*5.061*(Hs^2)/(Tp^4)*(1-0.287*log(gamma))*9.81^2*
      omega^(-5)*...
        exp(-1.25*(omegap/omega)^4)*gamma^exp(Y);

    S(j,1)=5.061*(Hs^2)/(Tp^4)*(1-0.287*log(gamma))*9.81^2*omega
      ^(-5)*...
        exp(-1.25*(omegap/omega)^4)*gamma^exp(Y);

m0RelVert(j)=sRelVert;
m1RelVert(j)=omega*sRelVert;
m2RelVert(j)=omega^2*sRelVert;
m4RelVert(j)=omega^4*sRelVert;

Omega(j,1)=omega;
SR(j,1)=sRelVert;

end

Omega;
SR;

%Calculating the motion standard deviation
m0RelVert;
M0RelVert=trapz(Omega,m0RelVert);
RMSRelVertMo=sqrt(M0RelVert);

%Calculating the velocity standard deviation
m2RelVert;
M2RelVert=trapz(Omega,m2RelVert);
RMSRelVertVel=sqrt(M2RelVert);

%Calculating the acceleration standard deviation
m4RelVert;
%M4RelVert=sum(m4RelVert)
M4RelVert=trapz(Omega,m4RelVert);
RMSRelVertAcc=sqrt(M4RelVert);

%------------------------------------------------------------
%Probability of slamming

Vcr=0.093*sqrt(9.81*Lpp);
ProbSlam=exp(-(Vcr^2/(2*RMSRelVertVel^2)+11^2/(2*RMSRelVertMo^2)));
```

```
%-------------------------------------------------------------------
%Probability of green water on deck

ProbGW=exp(-(freeboard)^2/(2*RMSRelVertMo^2));


%Defines whether the arc is an option
if ProbSlamming>ProbSlam &&...
    ProbGreenWater>ProbGW &&...
    VAccFP>RMSRelVertAcc &&...
    RMSRoll>RMSroll
    RouteOK=1 ;             %RouteOK equals 1 means optional
else
    RouteOK=0 ;              %RouteOK equals 0 delete option

end

end
```

A.9. optfuel2.m

```
%-------------------------------------------------------------
%                optfuel2.m
%                Hege Eskild
%                Spring 2014
%
% This is the optimal fuel function for actual wind. The method
% is described in chapter 3 in the thesis. The only difference
% between this function and the optfuel function is that the
% speed is calculated and not read from the ShipX file.
%
%-------------------------------------------------------------
%   Variable name                Definition
%----------------------Input-----------------------------
%   startLat                     Latitude coordinate departure
%   startLon                     Longitude coordinate departure
%   endLat                       Latitude coordinate arrival
%   endLon                       Longitude coordinate arrival
%   ProbSlamming                 Seakeeping criterion for slamming
%   ProbGreenWater               Seakeeping criterion for deck
%                                wetness
%   VAccFP                       Seakeeping criterion for vertical
%                                acceleration
%   RMSRoll                      Seakeeping criterion for roll
%   NumCenPts                    Number of center points for opt
%                                route
%   DeltaNM                      Distance in NM between center
%                                points
%                                practical GCR
%   Pb                           Desired brake power
%   bsfc                         Specific fuel consumption (g/kWh)
%   Xp                           X coordinate of FP (negative from
%                                   CG)
%    freeboard                      freeboard at FP
%    A_T                            Transverse projected area
%    D                              Propeller diameter
%    Eta_M                          Mechanical efficiency
%    Eta_R                          Relative rotational efficiency
%    AVspeed                        Matrix with attainable speed for
%                                   different sea states
%    AddedRes                       Matrix with increased resistance
%                                   due to waves
%    C_wind                         Matrix with wind drag coefficient
%                                   for different headings
%    R_calm                         Calm water resistance matrix
%    OpenWater                      Matrix with open water values
%    C_wake                         Wake coefficient
%    C_t                            Thrust deduction coefficient
%    lon                            Vector with longitude values from
%                                   GRIB
%    lat                            Vector with latitude values from
%                                   GRIB
%    time                           Vector with time values from GRIB
%    Uwind                          Matrix with U component of wind
%                                   from GRIB
%    Vwind                          Matrix with V component of wind
%                                   from GRIB
%    SWH                            Matrix with Hs from GRIB
%    MWD                            Matrix with mean wave direction
%                                   from GRIB
%    MWP                            Matrix with mean wave period
%                                   from GRIB
%    TRANS(..)                      Matrices including transfer
%                                   functions for roll and heave/
%                                   pitch for different frequencies
%                                   heading and velocity.
%    NOHEAD                         Number of heading in transfer
%                                   file
%    NOFREQ                         Number of frequencies in
%                                   transfer files
%    Lpp                            Length per perpendiculars
%    GCFC                           Fuel consumption for GCR
%    GClats                         Latitude vector for GCR
%    GClons                         Longitude vector for GCR
%---------------------Output--------------------------------
%    FC                             Fuel consumption for opt. route
%---------------------Internal--------------------------------
%    i                              Counting integer
%    j                              Counting integer
%    k                              Counting integer
%    latC                           Vector for latitude values of
%                                   center points
%    lonC                           Vector for longitude values of
%                                   center points
%    lats                           Matrix with all latitude values in
%                                   the feasible region
%    lons                           Matrix with all longitude values in
%                                   the feasible region
%    Hs                             Significant wave height for
%                                   the given arc
%    T1                             Mean wave period for the
%                                   given arc
%    Tp                             Peak wave period for the
%                                   given arc
%    head                           Relative heading between wave
%                                   and vessel or given arc
%    V                              Ship's speed
%    arcTime(S/R/L)                 Matrix with traveled time for
%                                   all arcs. One for straight, left
  and
%                                   right
%    arcFuel(S/R/L)                 Matrix with fuel consumption for
%                                   all arcs One for straight, left and
%                                   right
%    from(S/R/L)                    Departure nodes in sparse matrix
%    to(S/R/L)                      Arrival nodes in sparse matrix
%    fuel(S/R/L)                    Fuel cost between from and to

function [FC]=optfuel2(startLat,startLon,endLat,endLon,NumCenPts,Pb
  ,...
 DeltaNM,bsfc,AVspeed,lon,lat,time,Uwind,Vwind,...
 SWH,MWD,MWP,ProbSlamming,ProbGreenWater,VAccFP,...
 RMSRoll,TRANSR2,TRANSR4,TRANSR6,TRANSR8,TRANSR10,TRANSHP,NOFREQ
   ,...
 NOHEAD,Lpp,Xp,freeboard,AddedRes,C_wind,R_calm,...
 OpenWater,A_T,Eta_M,C_wake,C_t,Eta_R,D,GClats,GClons);


%
  -------------------------------------------------------------------

%Create the network of nodes. Ref step 1 in fig. 3.4 in thesis
%-------------------------------------------------------------------
[GClat,GClon]=gcwaypts(startLat,startLon,endLat,endLon,50);
%Finding the center point in the route, step 1 in algorithm 3.2 in
%the thesis
CenterLat=GClat(26);
CenterLon=GClon(26);
%Finding the course between center point, and one before
%center point, step 2 in algorithm 3.2 in thesis
Course=azimuth('rh',GClat(25),GClon(25),CenterLat,CenterLon);

%Calculating point along perpendicular to center point, step 4
%in algorithm 3.2 in thesis
%Points above center line
```

```
for i=1:(NumCenPts-1)/2
    [latCenterN(i,1),lonCenterN(i,1)]=reckon('rh',CenterLat,...
    CenterLon,i*nm2deg(DeltaNM),(Course-90));
end
%Points below centerline
for i=1:(NumCenPts-1)/2
    [latCenterS(i,1),lonCenterS(i,1)]=reckon('rh',CenterLat,...
    CenterLon,i*nm2deg(DeltaNM),(Course+90));
end

%Creating one vector with center points for both above
%and under the line
for i=1:NumCenPts
    if i<(NumCenPts+1)/2
        latC(i,1)=latCenterN((NumCenPts+1)/2-i,1);
        lonC(i,1)=lonCenterN((NumCenPts+1)/2-i,1);
    elseif i==(NumCenPts+1)/2
        latC(i,1)=CenterLat;
        lonC(i,1)=CenterLon;
    else
        latC(i,1)=latCenterS(i-(NumCenPts+1)/2,1);
        lonC(i,1)=lonCenterS(i-(NumCenPts+1)/2,1);
    end
end

latC;
lonC;

%Defining point along the rhumb lines between start and center
%points and center points and end. Step 6 in algorithm 3.2
%in thesis.
for i=1:NumCenPts
    %points for first half
    [arclen1(i,1),az1(i,1)] = distance('rh',startLat,startLon,...
    latC(i),lonC(i));
    [lat1,lon1] = track2('rh',startLat,startLon,latC(i),lonC(i),...
    referenceSphere('unit sphere'),'degrees',round(arclen1(1,1)));
    lats1(:,i)=lat1;
    lons1(:,i)=lon1;

    %points for second half
    [arclen2(i,1),az2(i,1)] = distance('rh',latC(i),lonC(i),...
    endLat,endLon);
    [lat2,lon2] = track2('rh',latC(i),lonC(i),endLat,endLon,...
    referenceSphere('unit sphere'),'degrees',round(arclen1(1,1)));
    lats2(:,i)=lat2;
    lons2(:,i)=lon2;
end

%Creating two matrices with all nodes in the feasible region.
%Where the column represents the number of center points
%and the rows are the stages in the voyage, i.e. the same time
```

```
%step. Ref. fig 3.8 in the thesis.
s=length(lats1(:,1));

for i=1:(2*s-1)
    if i<s
        lats(i,:)=lats1(i,:);
        lons(i,:)=lons1(i,:);
    else
        lats(i,:)=lats2(i-s+1,:);
        lons(i,:)=lons2(i-s+1,:);
    end
end
lats;
lons;

%----------------------------------------------------------------
%Defining arcs between nodes. Three matrices created, one
%for choosing the node to the left, one for straight, and
%for choosing the node to the right. The definitions
%of right and left is based on how the lats and lons vectors
%are set up. The actual direction is dependent on the sailing
%route.

%Second part is to add a cost to the given arc. This cost is
%the fuel consumption which is calculated by the weather and
%available speed. If the arc does not satisfy the seakeeping
%criteria the fuel consumption is set to 1000 so that this arc
%will not be chosen.

%This part of optfuel refers to the second and third step
%in fig. 3.4 in the thesis

for i=1:length(lats)-1
    for j=1:NumCenPts
        %-----------------------------------------------------
        %Defining arcs and cost for going straight
        %-----------------------------------------------------
        [arcStraight(i,j),azS(i,j)]=distance('gc',lats(i,j),...
        lons(i,j),lats(i+1,j),lons(i+1,j));
        %Finding the index to the latitude and longitude
        %weather vector
        [~, idxlat] = min(abs(lat - lats(i+1,j)));
        [~, idxlon] = min(abs(lon - lons(i+1,j)));
        az = azimuth(lats(i,j),lons(i,j),lats(i+1,j),lons(i+1,j));
        %Finding Hs, T1 and direction for the given arc
        %Hs, T1 and dir is assigned value if NaN
        idxtime=round(i/2);
        Hs=SWH(idxtime,idxlat,idxlon);
                if Hs<1 || isnan(Hs)
            Hs=1;
        end
        T1=MWP(idxtime,idxlat,idxlon);
```

```
if  isnan(T1)
    T1=5.4;
end
Tp=T1*1.3;
dir=MWD(idxtime,idxlat,idxlon);
if  isnan(dir)
    dir=90;
end
U_wind=Uwind(idxtime,idxlat,idxlon);
V_wind=Vwind(idxtime,idxlat,idxlon);
%Finding relative heading between wave and vessel
head=sqrt(az^2+dir^2-2*az*dir);
    if head>180
        head=360-head;
    end
headS(i,j)=head;
TpS(i,j)=Tp;
HsS(i,j)=Hs;

%Assigning the speed to the arc for the given weather
%condition
[V,head_wind,Vel_wind]=calculateSpeed(Hs,T1,head,U_wind,...
    V_wind,AddedRes,C_wind,R_calm,...
    OpenWater,C_wake,C_t,A_T,Eta_M,Eta_R,D,Pb,az);

VS(i,j)=V;
Head_windS(i,j)=head_wind;
Vel_windS(i,j)=Vel_wind;

%Assigning the fuel cost for the given arc
arcTimeS(i,j)=deg2nm(arcStraight(i,j))/V;
arcFuelS(i,j)=Pb*arcTimeS(i,j)*bsfc/1000000;

%Seakeeping function, checking criteria
[RouteOK,ProbSlam,ProbGW,RMSRelVertAcc,RMSroll]=...
seakeeping(T1,Hs,V,head,ProbSlamming,ProbGreenWater,...
VAccFP,RMSRoll,TRANSR2,TRANSR4,TRANSR6,TRANSR8,...
TRANSR10,TRANSHP,NOFREQ,NOHEAD,Lpp,Xp,freeboard);
%Giving the arcFuel a value of 1000 if the routeOK
%is not satisfied
if RouteOK==0
    arcFuelS(i,j)=1000;
end
ProbSlamS(i,j)=ProbSlam;
ProbGWS(i,j)=ProbGW;
RMSRelVertAccS(i,j)=RMSRelVertAcc;
RMSrollS(i,j)=RMSroll;


%-------------------------------------------------------
%Defining arcs and cost for going right
%------------------------------------------------------
```

```
if j<NumCenPts
[arcRight(i,j),azR(i,j)]=distance('gc',lats(i,j),lons(i,j)
  ,...
lats(i+1,j+1),lons(i+1,j+1));
%Finding the index to the latitude and longitude weather
  vector
[~, idxlat] = min(abs(lat - lats(i+1,j+1)));
[~, idxlon] = min(abs(lon - lons(i+1,j+1)));
az = azimuth(lats(i,j),lons(i,j),lats(i+1,j+1),lons(i+1,j
  +1));
%Finding Hs, T1 and direction for the given arc
idxtime=round(i/2);
Hs=SWH(idxtime,idxlat,idxlon);
        if Hs<1 || isnan(Hs)
    Hs=1;
end
T1=MWP(idxtime,idxlat,idxlon);
if isnan(T1)
    T1=5.4;
end
Tp=T1*1.3;
dir=MWD(idxtime,idxlat,idxlon);
if isnan(dir)
    dir=90;
end
U_wind=Uwind(idxtime,idxlat,idxlon);
V_wind=Vwind(idxtime,idxlat,idxlon);
%Finding relative heading between wave and vessel
head=sqrt(az^2+dir^2-2*az*dir);
    if head>180
        head=360-head;
    end
headR(i,j)=head;
TpR(i,j)=Tp;
HsR(i,j)=Hs;


%Assigning the speed to the arc for the given weather
%condition
[V,head_wind,Vel_wind]=calculateSpeed(Hs,T1,head,U_wind,...
    V_wind,AddedRes,C_wind,R_calm,...
    OpenWater,C_wake,C_t,A_T,Eta_M,Eta_R,D,Pb,az);

%Assiging the fuel cost for the given arc
arcTimeR(i,j)=deg2nm(arcRight(i,j))/V;
arcFuelR(i,j)=Pb*arcTimeR(i,j)*bsfc/1000000;

%Seakeeping function, checking criteria
[RouteOK,ProbSlam,ProbGW,RMSRelVertAcc,RMSroll]=...
seakeeping(T1,Hs,V,head,ProbSlamming,ProbGreenWater,...
VAccFP,RMSRoll,TRANSR2,TRANSR4,TRANSR6,TRANSR8,...
TRANSR10,TRANSHP,NOFREQ,NOHEAD,Lpp,Xp,freeboard);
```

```
%Giving the arcFuel a value of 1000 if the routeOK is
%not satisfied
if RouteOK==0
    arcFuelR(i,j)=1000;
end
ProbSlamR(i,j)=ProbSlam;
ProbGWR(i,j)=ProbGW;
RMSRelVertAccR(i,j)=RMSRelVertAcc;
RMSrollR(i,j)=RMSroll;
end

if j>1
%-------------------------------------------------------
%Defining arcs and costs for going right
%------------------------------------------------------
[arcLeft(i,j),azL(i,j)]=distance('gc',lats(i,j),lons(i,j)
  ,...
lats(i+1,j-1),lons(i+1,j-1));
%Finding the index to the latitude and longitude weather
  vector
[~, idxlat] = min(abs(lat - lats(i+1,j-1)));
[~, idxlon] = min(abs(lon - lons(i+1,j-1)));
az = azimuth(lats(i,j),lons(i,j),lats(i+1,j-1),lons(i+1,j
  -1));
%Finding Hs, T1 and direction for the given arc
idxtime=round(i/2);
Hs=SWH(idxtime,idxlat,idxlon);
        if Hs<1 || isnan(Hs)
Hs=1;
end
T1=MWP(idxtime,idxlat,idxlon);
if isnan(T1)
    T1=5.4;
end
Tp=T1*1.3;
dir=MWD(idxtime,idxlat,idxlon);
if isnan(dir)
    dir=90;
end
U_wind=Uwind(idxtime,idxlat,idxlon);
V_wind=Vwind(idxtime,idxlat,idxlon);
%Finding relative heading between wave and vessel
head=sqrt(az^2+dir^2-2*az*dir);
if head>180
    head=360-head;
end
headL(i,j)=head;
TpL(i,j)=Tp;
HsL(i,j)=Hs;


%Assigning the speed to the arc for the given weather
```

```
            %condition
            [V,head_wind,Vel_wind]=calculateSpeed(Hs,T1,head,U_wind,...
                V_wind,AddedRes,C_wind,R_calm,...
                OpenWater,C_wake,C_t,A_T,Eta_M,Eta_R,D,Pb,az);

            %Assiging the fuel cost for the given arc
            arcTimeL(i,j)=deg2nm(arcLeft(i,j))/V;
            arcFuelL(i,j)=Pb*arcTimeL(i,j)*bsfc/1000000;    %Fuel
              consumption in tonn

            %Seakeeping function, checking criteria
            [RouteOK,ProbSlam,ProbGW,RMSRelVertAcc,RMSroll]=...
            seakeeping(T1,Hs,V,head,ProbSlamming,ProbGreenWater,...
            VAccFP,RMSRoll,TRANSR2,TRANSR4,TRANSR6,TRANSR8,...
            TRANSR10,TRANSHP,NOFREQ,NOHEAD,Lpp,Xp,freeboard);
            %Giving the arcFuel a value of 1000 if the routeOK is not
              satisfied
            if RouteOK==0
                arcFuelL(i,j)=1000;
            end
            ProbSlamL(i,j)=ProbSlam;
            ProbGWL(i,j)=ProbGW;
            RMSRelVertAccL(i,j)=RMSRelVertAcc;
            RMSrollL(i,j)=RMSroll;
            end
        end
end

arcFuelS;
arcFuelR;
arcFuelL(:,1)=[];

%Next section is used to verify the different arcs and their
%calculated values
%Matrices corresponding to the straight arcs:
arcStraight; %Arc lengths
headS;
HsS;
TpS;
VS;

Head_windS;
Vel_windS;

%Optinal to write values to excel spreadsheet
% xlswrite('optfuel2.xls',Head_windS,'head')
% xlswrite('optfuel2.xls',Vel_windS,'Vel')
% xlswrite('optfuel2.xls',VS,'V')

%Responses
ProbSlamS;
ProbGWS;
```

```
RMSRelVertAccS;
RMSrollS;

%Matrices corresponding to the right arcs:
arcRight;
headR;
HsR;
TpR;
%responses
ProbSlamR;
ProbGWR;
RMSRelVertAccR;
RMSrollR;

%Matrices corresponding to left arcs
arcLeft;
headL;
HsL;
TpL;
%Responses
ProbSlamL;
ProbGWL;
RMSRelVertAccL;
RMSrollL;

%------------------------------------------------------------
%Optimization. The last part of the optfuel function refers
%to the fourth step in fig. 3.4. To use Dijstras algorithm
%in Matlab a sparse matrix must be defined. This sparse matrix
%includes a vector with nodes that are departed (from) and nodes
%that are arrival nodes (to). The sparse matrix also includes a
%vector of the cost of going from to. This part of the function is
%difficult to understand, mostly trixing with matrices and vectors
%See fig 3.8 in thesis to understand the setup.
%------------------------------------------------------------
%Assigns an actual number to each node
for i=1:length(lats)-1
    for j=1:NumCenPts
        nodeS(i,j)=j+1+(i-1)*NumCenPts;
    end
end
nodeS;
nodeR=nodeS;
nodeR(:,1)=[];
nodeL=nodeS;
nodeL(:,NumCenPts)=[];

SizeNodes=size(nodeS);
NumberNodes=SizeNodes(1,1)*SizeNodes(1,2);

%Node connection
%Define the straight points
```

```
for i=1:((length(lats(:,1))-1)*NumCenPts)
    if i<NumCenPts+1;
        fromS(i)=1;
    else
    fromS(i)=i-NumCenPts+1;
    end
end


%Defining to nodes for straight arcs and cost
for i=1:((length(lats(:,1))-1)*NumCenPts)
    if i<NumberNodes-NumCenPts+1
    toS(i)=1+i;
    fuelS(i)=arcFuelS(nodeS==(i+1));
    else
        toS(i)=((length(lats(:,1)))-2)*NumCenPts+2;
        fuelS(i)=arcFuelS(nodeS==(i+1));
    end
end

%Define the right arcs
for j=1:((length(lats(:,1))-2)*(NumCenPts-1))
    fromR(j)=j+1;
end

for j=1:((length(lats(:,1))-2)*(NumCenPts-1))
    if j<((length(lats(:,1))-2)*(NumCenPts))-NumCenPts+1
        toR(j)=NumCenPts+j+1;
    else
        toR(j)=((length(lats(:,1)))-2)*NumCenPts+2;
    end

end

%Editing the to and from nodes and cost for
%right and left arcs
fromR;
toR;
fromL=fromR;
toL=toR;
fromR(NumCenPts:NumCenPts:end)=[];
toR(1:NumCenPts:end)=[];
fromL(1:NumCenPts:end)=[];
toL(NumCenPts:NumCenPts:end)=[];

%Assigning fuel cost for right and left arcs
for i=1:length(toR)
    fuelR(i)=arcFuelR(nodeR==toR(i));
    fuelL(i)=arcFuelL(nodeL==toL(i));
end

%Defining the final to, from and cost vectors to be
```

```
%used in the sparse matrix.
for i =1:length(fuelS)+length(fuelR)+length(fuelL)
    if i<length(fuelS)+1
        fuel(i)=fuelS(i);
        from(i)=fromS(i);
        to(i)=toS(i);
    elseif i<length(fuelS)+length(fuelR)+1
        fuel(i)=fuelR(i-length(fuelS));
        from(i)=fromR(i-length(fuelS));
        to(i)=toR(i-length(fuelS));
    else
        fuel(i)=fuelL(i-length(fuelS)-length(fuelR));
        from(i)=fromL(i-length(fuelS)-length(fuelR));
        to(i)=toL(i-length(fuelS)-length(fuelR));
    end
end
%Defining a sparse to use in shortest path function
SparseGraph = sparse(from,to,fuel);
MatrixGraph = full(SparseGraph);
%Adding an additional row to the full matrix so that the matrix
%is symetrical. The added row is defined as an edge.
MatrixGraph((length(lats)-2)*NumCenPts+2,:)=0;
%Converting matrix back to sparse
SparseGraph=sparse(MatrixGraph);

%Calculating the fuel consumption and path for the
%optimization problem with Dijkstra's algorithm
[FC,path,pred] = graphshortestpath(SparseGraph,1,...
(length(lats)-2)*NumCenPts+2,'Method','Dijkstra');

FC;
path;

%Plotting the graph of nodes and connecting arcs
h = view(biograph(SparseGraph,[],'ShowWeights','on'))
set(h.Nodes(path),'Color',[1 0.4 0.4])
edges = getedgesbynodeid(h,get(h.Nodes(path),'ID'));
set(edges,'LineColor',[1 0 0])
set(edges,'LineWidth',1.5)

%Assigning geographical coordinates to the nodes in
%order to plot in map

for i=2:length(lats)-1
   %locating index for nodes via path
   [~,indnodeS]=find(nodeS(i-1,:)==path(i));
   OptFuelPathLat(i,1)=lats(i,indnodeS);
   OptFuelPathLon(i,1)=lons(i,indnodeS);
end
OptFuelPathLat(1,1)=startLat;
OptFuelPathLon(1,1)=startLon;
OptFuelPathLat(length(lats),1)=endLat;
```

```
OptFuelPathLon(length(lats),1)=endLon;


%Map plot with nodes and route
S=8;
figure (2)
geoshow('landareas.shp', 'FaceColor', [0.5 1.0 0.5]);
hold on
geoshow(GClats,GClons)
hold on
geoshow(OptFuelPathLat,OptFuelPathLon,'displaytype','line','color',
  'r')

for i=1:NumCenPts
hold on
scatter(lons(:,i),lats(:,i),S,'fill','b')
end

hold on
scatter(lonC,latC,S,'fill','r')


end
```

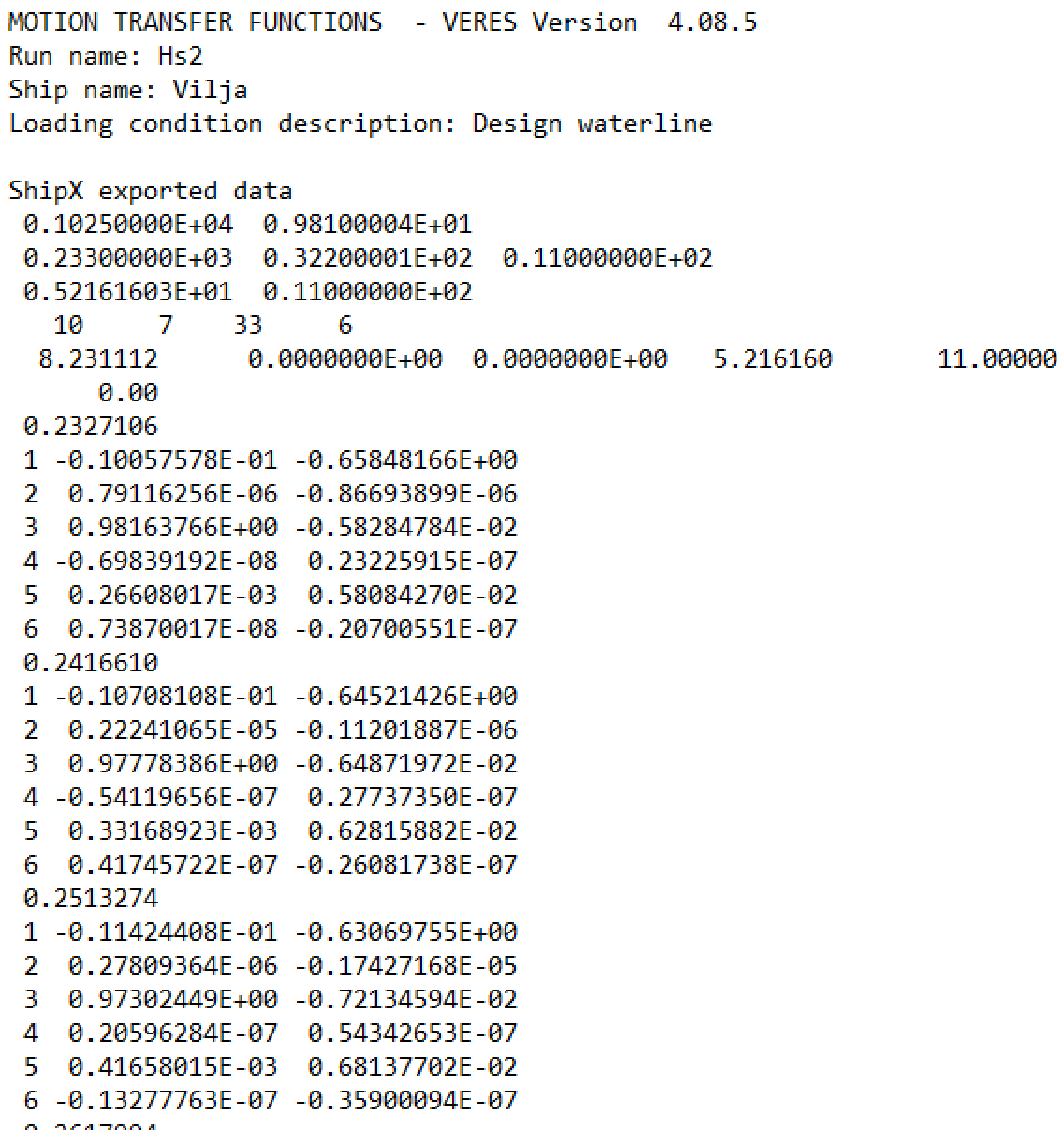
A.10. calculateSpeed.m

```
%--------------------------------------------------------------
%                calculatespeed.m
%                Hege Eskild
%                Spring 2014
%
% This is the underlying function in optfuel2. It calculates the
% attainable speed for a given sea state. The added resistance
% from waves is defined as a percenteage incease of the calm water
% resitance. The added resistance from wind is calculated.
% Propulsion factors are find by using cubic spline interpolation
% from the open water test and the propulsion test
%
%--------------------------------------------------------------
%   Variable name                   Definition
%----------------------Input------------------------------
%   T1                              Mean wave period
%   Hs                              Significant wave height
%   V                               Vessel speed
%   head                            relative heading between waves and
%                                   vessel
%   Pb                              Desired brake power
%   D                               Propeller diameter
%   Eta_M                           Mechanical efficiency
%   Eta_R                           Relative rotational efficiency
%   AddedRes                        Matrix with increased resistance
%                                   due to waves
%    C_wind                          Matrix with wind drag coefficient
%                                    for different headings
%    R_calm                          Calm water resistance matrix
%    OpenWater                       Matrix with open water values
%    U_wind                          U component of wind
%    V_wind                          V component of wind
%    C_wake                          Wake coefficient
%    C_t                             Thrust deduction coefficient
%---------------------Output-------------------------------
%    V                               Vessel velocity
%    head_wind                       Heading of wind
%    Vel_wind                        Wind velocity


function [V,head_wind,Vel_wind]=calculateSpeed(Hs,T1,head,U_wind
  ,...
         V_wind,AddedRes,C_wind,R_calm,OpenWater,C_wake,C_t,A_T,...
         Eta_M,Eta_R,D,Pb,az)

Tp=1.3*T1;
V=19;
P_B=0;

rho_air=1.2250;

%finding resistance increase due to waves

for k=1:length(AddedRes)
    if AddedRes(k,1)>(head-15) && AddedRes(k,1)<(head+15)...
       && AddedRes(k,2)>(Hs-0.5) && AddedRes(k,2)<(Hs+0.5)...
       && AddedRes(k,3)>(Tp-0.5) && AddedRes(k,3)<(Tp+0.5)
          R_w=AddedRes(k,4);
    end
end

%Calculating the wind velocity and heading in a 360 degrees range
Vel_wind=sqrt(U_wind^2+V_wind^2);
head_wind1=atan(abs(U_wind/V_wind))*180/pi();

if V_wind<0 && U_wind>0
    head_wind1=atan(abs(V_wind/U_wind))*180/pi();
    head_wind=head_wind1+90;
elseif V_wind<0 && U_wind<0
    head_wind1=atan(abs(U_wind/V_wind))*180/pi();
    head_wind=head_wind1+180;
elseif V_wind>0 && U_wind<0
    head_wind1=atan(abs(V_wind/U_wind))*180/pi();
    head_wind=head_wind1+270;
else
    head_wind1=atan(abs(U_wind/V_wind))*180/pi();
    head_wind=head_wind1;
```

```
end

head_wind;

%Calculating wind heading relative to ship heading
head_wind=head_wind-az-180;
    if head_wind<-180
        head_wind=abs(head_wind+360);
    elseif head_wind<0
        head_wind=abs(head_wind);
    end
head_wind;
%finding wind resistance coefficient with interpolation
C_Rwind= spline(C_wind(:,1),C_wind(:,2),head_wind);


while P_B<Pb && V<25.5
    V=V+0.02;
    Vs=V*1852/3600;
    %Resistance due to wind
    R_wind=C_Rwind*0.5*rho_air*(Vel_wind^2+2*Vel_wind*Vs)*A_T/1000;

    %Calm water resistance
    %Interpolating restistance to find accurate resistance for
    %given speed in Kn
    R_C= spline(R_calm(:,1),R_calm(:,2),V);

    R_wave=R_w/100*R_C;

    %Finding total resistance
    R_T=R_wind+R_wave+R_C;

    %Calculating efffective power
    P_E=R_T*Vs;

    %Finding the RPM
    KtoverJ2=R_T*10^3/(1025*(1-C_t)*D^2*Vs^2*(1-C_wake)^2);
    J_full=spline(OpenWater(:,5),OpenWater(:,1),KtoverJ2);
    RPM=60*(1-C_wake)/D*Vs/J_full;
    n=RPM/60;

    %Calculating J
    V_A=Vs*(1-C_wake);
    J=V_A/(n*D);

    Eta_0= spline(OpenWater(:,1),OpenWater(:,4),J);

    %Hull efficiency
    Eta_H=(1-C_t)/(1-C_wake);

    %Calculating the propulsive efficiency
    Eta_D=Eta_H*Eta_0*Eta_R;
    P_D=P_E/Eta_D;

    %Brake power
    P_B=P_D/Eta_M;

end
end
```

B. Input files

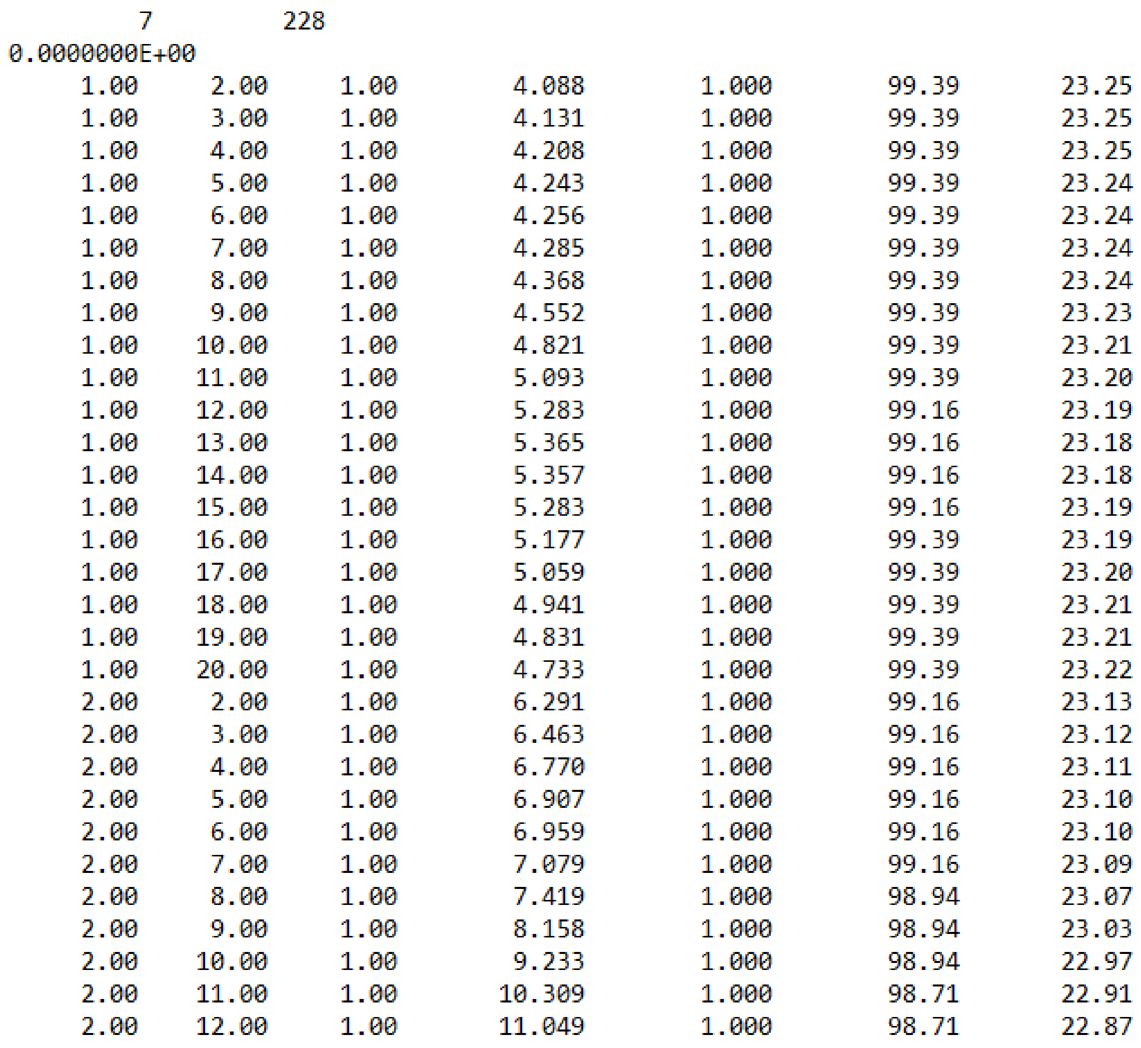
B.1. Transfer file



```
MOTION TRANSFER FUNCTIONS  - VERES Version  4.08.5
Run name: Hs2
Ship name: Vilja
Loading condition description: Design waterline

ShipX exported data
 0.10250000E+04  0.98100004E+01
 0.23300000E+03  0.32200001E+02  0.11000000E+02
 0.52161603E+01  0.11000000E+02
   10     7    33     6
  8.231112      0.0000000E+00  0.0000000E+00   5.216160       11.00000
      0.00
 0.2327106
 1 -0.10057578E-01 -0.65848166E+00
 2  0.79116256E-06 -0.86693899E-06
 3  0.98163766E+00 -0.58284784E-02
 4 -0.69839192E-08  0.23225915E-07
 5  0.26608017E-03  0.58084270E-02
 6  0.73870017E-08 -0.20700551E-07
 0.2416610
 1 -0.10708108E-01 -0.64521426E+00
 2  0.22241065E-05 -0.11201887E-06
 3  0.97778386E+00 -0.64871972E-02
 4 -0.54119656E-07  0.27737350E-07
 5  0.33168923E-03  0.62815882E-02
 6  0.41745722E-07 -0.26081738E-07
 0.2513274
 1 -0.11424408E-01 -0.63069755E+00
 2  0.27809364E-06 -0.17427168E-05
 3  0.97302449E+00 -0.72134594E-02
 4  0.20596284E-07  0.54342653E-07
 5  0.41658015E-03  0.68137702E-02
 6 -0.13277763E-07 -0.35900094E-07
```

B.2. Speed loss files

B.2.1. Estimated wind



```
          7          228
 0.0000000E+00
      1.00      2.00      1.00         4.088         1.000         99.39         23.25
      1.00      3.00      1.00         4.131         1.000         99.39         23.25
      1.00      4.00      1.00         4.208         1.000         99.39         23.25
      1.00      5.00      1.00         4.243         1.000         99.39         23.24
      1.00      6.00      1.00         4.256         1.000         99.39         23.24
      1.00      7.00      1.00         4.285         1.000         99.39         23.24
      1.00      8.00      1.00         4.368         1.000         99.39         23.24
      1.00      9.00      1.00         4.552         1.000         99.39         23.23
      1.00     10.00      1.00         4.821         1.000         99.39         23.21
      1.00     11.00      1.00         5.093         1.000         99.39         23.20
      1.00     12.00      1.00         5.283         1.000         99.16         23.19
      1.00     13.00      1.00         5.365         1.000         99.16         23.18
      1.00     14.00      1.00         5.357         1.000         99.16         23.18
      1.00     15.00      1.00         5.283         1.000         99.16         23.19
      1.00     16.00      1.00         5.177         1.000         99.39         23.19
      1.00     17.00      1.00         5.059         1.000         99.39         23.20
      1.00     18.00      1.00         4.941         1.000         99.39         23.21
      1.00     19.00      1.00         4.831         1.000         99.39         23.21
      1.00     20.00      1.00         4.733         1.000         99.39         23.22
      2.00      2.00      1.00         6.291         1.000         99.16         23.13
      2.00      3.00      1.00         6.463         1.000         99.16         23.12
      2.00      4.00      1.00         6.770         1.000         99.16         23.11
      2.00      5.00      1.00         6.907         1.000         99.16         23.10
      2.00      6.00      1.00         6.959         1.000         99.16         23.10
      2.00      7.00      1.00         7.079         1.000         99.16         23.09
      2.00      8.00      1.00         7.419         1.000         98.94         23.07
      2.00      9.00      1.00         8.158         1.000         98.94         23.03
      2.00     10.00      1.00         9.233         1.000         98.94         22.97
      2.00     11.00      1.00        10.309         1.000         98.71         22.91
      2.00     12.00      1.00        11.049         1.000         98.71         22.87
```

B.2.2. Actual wind

```
           H E A D I N G  =   0.00

    Hs        Tp      Gamma        R/R0      Rwind/R0     Rwave/R0
    m         s         -           %           %            %

   1.00      2.00     1.00        4.118       4.115        0.003
   1.00      3.00     1.00        4.217       4.115        0.102
   1.00      4.00     1.00        4.285       4.114        0.171
   1.00      5.00     1.00        4.298       4.114        0.184
   1.00      6.00     1.00        4.299       4.114        0.185
   1.00      7.00     1.00        4.321       4.114        0.207
   1.00      8.00     1.00        4.402       4.113        0.288
   1.00      9.00     1.00        4.583       4.112        0.471
   1.00     10.00     1.00        4.852       4.110        0.742
   1.00     11.00     1.00        5.125       4.108        1.017
   1.00     12.00     1.00        5.317       4.107        1.210
   1.00     13.00     1.00        5.400       4.106        1.294
   1.00     14.00     1.00        5.393       4.106        1.287
   1.00     15.00     1.00        5.319       4.107        1.212
   1.00     16.00     1.00        5.212       4.107        1.105
   1.00     17.00     1.00        5.093       4.108        0.985
   1.00     18.00     1.00        4.973       4.109        0.864
   1.00     19.00     1.00        4.862       4.110        0.752
   1.00     20.00     1.00        4.762       4.111        0.652
   2.00      2.00     1.00        6.342       6.331        0.012
   2.00      3.00     1.00        6.733       6.327        0.406
   2.00      4.00     1.00        7.003       6.324        0.679
   2.00      5.00     1.00        7.055       6.323        0.732
   2.00      6.00     1.00        7.061       6.323        0.738
   2.00      7.00     1.00        7.154       6.322        0.832
   2.00      8.00     1.00        7.481       6.319        1.162
   2.00      9.00     1.00        8.213       6.311        1.902
   2.00     10.00     1.00        9.289       6.299        2.989
   2.00     11.00     1.00       10.370       6.288        4.082
   2.00     12.00     1.00       11.116       6.280        4.837
```

B.3. Wind resistance coefficient

```
0       0.58
10      0.65
20      0.78
30      0.75
40      0.7
50      0.56
60      0.4
70      0.25
80      0.1
90      0
100     -0.05
110     -0.18
120     -0.36
130     -0.45
140     -0.46
150     -0.52
160     -0.58
170     -0.5
180     -0.45
```

B.4. Calm water resistance

```
 16.00   552.910
 16.11   561.731
 16.22   570.657
 16.34   579.688
 16.45   588.830
 16.56   598.084
 16.67   607.453
 16.79   616.939
 16.90   626.547
 17.01   636.279
 17.12   646.137
 17.24   656.122
 17.35   666.237
 17.46   676.483
 17.57   686.861
 17.69   697.374
 17.80   708.026
 17.91   718.820
 18.02   729.760
 18.13   740.847
 18.25   752.087
 18.36   763.483
 18.47   775.036
 18.58   786.752
```

B.5. Open water

```
9
.5  .3597  .2431  0  0
.55  .3276  .2169  0  0
.6  .2944  .19  0  0
.65  .2603  .1624  0  0
.7  .2252  .1342  0  0
.75  .1891  .1054  0  0
.8  .152  .0759  0  0
.85  .114  .0459  0  0
.9  .075  .0154  0  0
```