NTNU – Trondheim
Norwegian University of
Science and Technology

Vortex Induced Fatigue Damage of a Steel Catenary Riser near the Touchdown Point

Emir Lejlic

Marine Technology

Submission date:	June 2013
Supervisor:	Carl Martin Larsen, IMT
Co-supervisor:	Per Erlend Voie, DNV

Norwegian University of Science and Technology
Department of Marine Technology

NTNU Trondheim
Norwegian University of Science and Technology
Faculty of Engineering Science and Technology
Department of MarineTechnology

M.Sc. thesis 2013
for

Stud.tech. Emir Lejlic

VORTEX INDUCED FATIGUE DAMAGE OF STEEL CATENARY RISERS NEAR THE TOUCHDOWN POINT.

Slender flexible structures in a marine environment like steel catenary risers (SCR) will experience conditions with insignificant wave forces in combination with strong current. In such cases the structural response will be dominated by vortex-induced vibrations (VIV). Although amplitudes are small compared to wave induced stresses, fatigue damage can be high because of the large number of stress cycles that may occur. State-of-the-art VIV fatigue design programs such as VIVANA, make the best use of available knowledge, but remain limited for several reasons in their capacity to capture the broad complexity of the phenomenon. In addition to more generic uncertainties with regard to structural capacity, like variations in material properties or S-N curves, much uncertainty is linked to the amount and relevance of data available to understand VIV itself. Time domain computation of VIV have been attempted, however most empirical models for VIV prediction are based on frequency domain analysis which imply certain limitations as to how geometry, boundary conditions and loads can be modelled.

A key issue in design of SCRs is to control stresses and fatigue damage in the touch down area. Dynamic bending stresses will vary along the SCR but a large peak will almost always be seen near the touch down point. This peak is caused by the restrictions on riser displacements from the presence of the seafloor, and the local bending stresses will be influenced by stiffness and damping properties of the bottom. Analysis models based on finite elements will represent the interaction between riser and seafloor by discrete springs, which in linear frequency domain analysis will remain constant independent of the displacements. This type of model may give a significant over-prediction of bending stresses at the touch down point since a linear spring will give tensile forces instead of being released and allowing the pipe to lift off from the bottom. Actually the touch down point moves with time, due to platform motions mainly, leading to more evenly distributed fatigue damage.

The objective of the master thesis will be to:

1. Find relevant literature about VIV and SCRs and give an overview of the topics.
2. Learn to use relevant software, such as SIMA, RIFLEX, VIVANA and BATCH-scripting.
3. Identify key variables and perform a fatigue sensitivity analysis with the VIVANA software.
4. Use data given from DNV and create a probability distribution of mean floater position to be used for calculation of fatigue from VIV.
5. Describe and investigate a method for distributing accumulated fatigue damage in the touch down area based on considerations of platform motions statistics.

The work may show to be more extensive than anticipated. Some topics may therefore be left out after discussion with the supervisor without any negative influence on the grading.

The candidate should in her/his report give a personal contribution to the solution of the problem formulated in this text. All assumptions and conclusions must be supported by mathematical models and/or references to physical effects in a logical manner. The candidate should apply all available sources to find relevant literature and information on the actual problem.

The report should be well organised and give a clear presentation of the work and all conclusions. It is important that the text is well written and that tables and figures are used to support the verbal presentation. The report should be complete, but still as short as possible.

The final report must contain this text, an acknowledgement, summary, main body, conclusions and suggestions for further work, symbol list, references and appendices. All figures, tables and equations must be identified by numbers. References should be given by author name and year in the text, and presented alphabetically by name in the reference list. The report must be submitted in two copies unless otherwise has been agreed with the supervisor.

The supervisor may require that the candidate should give a written plan that describes the progress of the work after having received this text. The plan may contain a table of content for the report and also assumed use of computer resources. From the report it should be possible to identify the work carried out by the candidate and what has been found in the available literature. It is important to give references to the original source for theories and experimental results. The report must be signed by the candidate, include this text, appear as a paperback, and - if needed - have a separate enclosure (binder, DVD/ CD) with additional material.

The work will be carried out in cooperation with Det Norske Veritas, Trondheim
Contact person at DNV is Per Erlend Voie.
Supervisor at NTNU is Professor Carl M. Larsen.

Carl M. Larsen

Submitted: January 2013
Deadline: 10 June 2013

Abstract

In this thesis, the fatigue damage of a steel catenary riser was investigated, especially in the vicinity of the touchdown point.

The steel catenary riser is a promising riser solution due to its simplicity and low cost. This is in particular true if the motions in heave direction are modest. As the offshore industry is moving into deeper waters, riser solutions will need to be designed with high safety, to ensure interests both economically and environmentally. Understanding how the structure reacts to forces from the environment is therefore of great concern. It is experienced that the steel catenary riser will have a noticeable increase in the fatigue damage in the proximity of the touchdown zone. This is the area where the riser has contact with the sea bottom. The sea bottom will impose forces on the riser, and because of the increased curvature of the structure in this area, the stresses will be magnified. This pipe–soil interaction seems to be highly non-linear and it is complex to model the riser realistically in this area.

When an ocean current passes the riser, it may form vortices that are shed from both sides of the pipe. If these vortices are shed with a frequency near the natural frequency of the riser, the pipe may start to vibrate. This response is called vortex induced vibration, and the phenomenon is important in riser design. Even if the amplitudes are small compared to vessel induced motions, the high number of cycles may give a significant contribution to the total fatigue damage.

In this work, the fatigue damage initiated by the vortex induced vibrations has been investigated, and fatigue analyses of a steel catenary riser have been carried out. In order to perform such analysis taking into account vortex induced vibrations, the computer program VIVANA has been used. VIVANA is an empirical computer program based on experiments and computes the vortex induced response in the frequency domain. In the frequency domain, the pipe–sea floor interaction is modelled with discrete springs at the nodes where the riser touches the ground. However, these springs are not released if the riser is lifted, something which gives non-realistic stresses in the area.

An alternative way of analysing the fatigue damage near the touchdown area has been proposed and performed. The analysis was based on floater motion statistics, and simulations were done changing the floater offset. The final fatigue result was found by summation of individual fatigue results for different coordinates of the floater in the XY-plane. The results indicated that the fatigue damage was evenly distributed in the proximity of the touchdown area, and the maximum and overall damage were reduced.

Sammendrag

Denne masteroppgaven har fokusert på utmattingsskade av et stigerørkonsept (Steel Catenary Riser) i nærheten av havbunnen.

Stigerøret har en buet form og er en lovende stigerørsløsning grunnet dens enkelthet og lave kostnad. Dette er spesielt sant dersom den vertikale fartøysbevegelsen er moderat. Da offshore-næringen beveger seg mot stadig dypere farvann vil det kreves nye stigerørsløsninger, som er designet med høy sikkerhet, for å sikre økonomiske og miljømessige interesser. Å forstå hvordan konstruksjonen reagerer på miljøkrefter er derfor av stor interesse. Det er observert at stigerøret vil ha en betydelig økning i utmattingsskaden i nærheten av landingssonen. Dette er området hvor stigerøret møter sjøbunnen. Sjøbunnen vil påføre krefter på stigerøret og grunnet den høye kurvaturen i området, vil man merke en økning i trykkspenningene. Denne interaksjonen hvor røret møter havbunnen er av ikke-lineær natur, og modelleringen av stigerøret er vanskelig og må gjøres i tidsdomenet.

Når en havstrøm driver forbi stigerøret, kan det oppstå virvler som blir avløst på hver side av konstruksjonen. Dersom virvlene blir avløst med en frekvens som er i nærheten av egenfrekvensen til stigerøret, kan det begynne å vibrere. Denne responsen er kalt for virvelindusert vibrasjon, og har stor betydning for potensielle utmattingsskader. Selv om responsamplitudene er små i forhold til fartøyinduserte bevegelser, kan det høye antallet sykluser gi et signifikant bidrag til utmattingsskaden.

I denne oppgaven har utmattingsskaden grunnet virvelavløsninger blitt undersøkt. For å utføre slike analyser har dataprogrammet VIVANA blitt brukt. VIVANA er et empirisk dataprogram og er basert på eksperimenter. Det beregner virvelindusert respons i frekvensdomenet. I frekvensdomenet er rør-havbunn interaksjonen modellert med diskrete fjær ved nodene som er i kontakt med havbunnen. Problemet er at disse fjærene ikke vil slippe stigerørsmodellen dersom den er løftet fra bakken. Dette skaper dermed falske krefter som gir konservative resultater i dette området.

En alternativ analysemetode av utmattingsskaden i landingssonen har blitt foreslått og undersøkt i oppgaven. Analysen baserte seg på statistikk av fartøysbevegelse, og simuleringer ble gjort i VIVANA. Den endelige utmattingsskaden ble funnet ved summering av de individuelle utmattingsskadene ved forskjellige posisjoner av fartøysbevegelsen. Resultatet indikerte at skaden ble jevnere fordelt i nærheten av landingssonen, og den samlede og maksimale skaden på stigerøret ble redusert.

Acknowledgement

This work was carried out at the Department of Marine Technology, at the Norwegian University of Science and Technology (NTNU), in cooperation with Det Norske Veritas (DNV) under the supervision of Professor Carl Martin Larsen, and under co-supervision of engineer Per Erlend Voie from DNV in Trondheim. I would like to thank them both for the assistance I have received during the semester. Professor Carl Martin Larsen has helped me throughout the semester with weekly meetings and conversations. This has been greatly appreciated. I am thankful to Per Erlend Voie for his scripting knowledge which has been helpful.

Thanks to my friends at the office and school for five good years here at the university. The discussions with the guys from the office have been of great importance. I also want to thank my family for all their support during my study years.

Lastly, I would like to thank my beautiful Idunn for always believing in me.

Emir Lejlić

Emir Lejlic,
Trondheim, Monday 10th June, 2013

Notation

Abbreviations

CF	Cross flow
FE	Finite Element
FPSO	Floating Production, Storage and Offloading
IL	In line
JONSWAP	Joint North Sea Wave Project
RAO	Response Amplitude Operator
RHoP	Riser hang off point
RP	Recommended practice
SCF	Stress concentration factor
SCR	Steel Catenary Riser
St	Strouhal number
TDP	Touchdown point
TDZ	Touch down zone
TLP	Tension leg plattform
VIM	Vortex induced motion
VIV	Vortex induced vibrations

Greek Symbols

α	Proportionality factor
ϵ	Phase angle
ν	Kinematic viscosity.
ω_n	Natural frequency n

ξ	Structural damping ratio

Roman symbols

$\hat{f}_i$	Non-dimensional frequency
$\mathbf{C}$	Damping matrix
$\mathbf{K}$	Stiffness matrix
$\mathbf{M}$	Mass matrix
$\mathbf{r}$,$\dot{\mathbf{r}}$,$\ddot{\mathbf{r}}$	Displacement, velocity and acceleration vectors.
$\mathbf{X}$	Excitation force vector
A/D	Amplitude ratio
A_{st}	Cross section steel area
D	Diameter of structure
E	Elastic modulus
$E_{i/n}$	Excitation parameter for frequency i or n
f_0	Eigenfrequency in still water
f_v	Vortex shedding frequency
$F_{e,CF}$	CF excitation force
f_{osc}	Oscillating frequency
H_s	Significant wave height
I	Second moment of area.
k	Spring stiffness
l	Length of riser
m	Mass per length
m_{a0}	Added mass
n	Mode number
T	Top tension
T_p	Peak period

t_{ref}	Reference thickness
U	Current velocity
U_r	Reduced velocity
$W_{y/z}$	Section modulus for bending about y- and z-axis
Re	Reynold number

Contents

List of Figures

List of Tables

Chapter 1
Introduction

The offshore industry is moving towards deeper waters, and floating facilities have therefore become an integral part of many field developments that are working as an oil and gas production facility, and as a hubs for developments of several remote wells or fields. The steel catenary riser (SCR) has become a good option for oil and gas export from these floating facilities to shallow water platforms, or to subsea hubs. The advantage of the SCR is its low cost compared to other riser designs such as flexible pipes that have complex set of layers, and which do not have the same resistance to hydrostatic pressure as rigid steel pipes. The use of SCRs within the industry has created a need of understanding their behaviour during installation, operation and nevertheless, under extreme conditions [1].

In the touchdown zone (TDZ), where the riser meets the sea bottom, the catenary shape of the riser will impose high stresses to the riser structure. This will be of particular importance due to the motions of the floating structure with varying degrees of freedom. The vessel motion causes the riser to move in different directions, and the touchdown zone will therefore vary in time. The dynamics of the floater and the ocean current will have a great effect on the riser. The restraints that the sea floor causes will give large fatigue damage in this area. One uses finite element (FE) models to represent the interaction between the sea floor and the riser. However, modelling the interaction is a difficult task with many uncertainties. The interaction model may be represented by discrete springs.

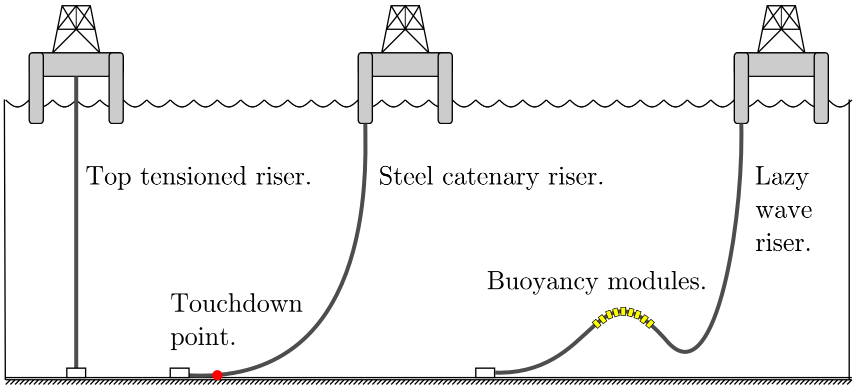


Figure 1.1: Example of three riser types.

In this work it has been investigated how vortex induced vibrations (VIV) influence the fatigue damage near the sea floor. VIV is a resonance phenomenon that is caused by vortices that are shed from both sides of the riser structure as the ocean current drifts past the pipe. In the offshore industry it is common to consider the ocean current coming from one direction, either parallel to the riser plane or orthogonal. Today, risers are being used in extreme depths, where the ocean current velocity may vary along the depth. It is therefore challenging to estimate an exact damage the riser will accumulate by VIV that is initiated by the ocean current.

In order to calculate VIV problems, several computer programs have been created. They are mostly based on empirical models on the assumption that VIV will appear as a response at one or a limited number of discrete frequencies. These computer programs are constantly changing as researches get new experimental results. In the work done during this master thesis, the computer programme VIVANA has been used to investigate how fatigue damage is accumulated near the touchdown zone of a steel catenary riser.

1.1 Steel Catenary Risers

Floating production systems must rely on some kind of marine risers for transport of the well-stream from the sea floor to the platform. In many cases they also need risers to transport processed oil and gas down to a pipeline. Among many different riser concepts, the steel catenary riser is promising due to its simplicity and low cost. This is in particular true if the heave motions of the floater are moderate, which is the case for tension leg platforms (TLP), SPAR buoys and deep drought floaters.

A key issue in the design of catenary risers is to control the stresses and the fatigue damage in the touchdown zone. Vessel motions and waves will cause time varying stresses and because of the boundary conditions such stresses will often have the largest values in this area. Another important phenomenon that may contribute significantly to the fatigue is the VIV caused by the ocean current. Although the stress amplitudes are smaller compared to wave motion induced stresses, fatigue damage may be high due to the high number of cycles that may occur.

A catenary riser is different from a tensioned riser in the sense that the current will have a flow direction relative to the pipe axis different from 90°. The basis for almost all empirical VIV models are experiments with oncoming flow perpendicular to the cylinder [2]. A consequense of this is that there are many uncertainties related to the hydrodynamics of catenary risers. Some of these uncertainties are the Strouhal number (St), added mass (M_A)and the lift coefficients. Comparing tests and analyses is therefore of great interest.

1.2 Pipe – sea floor iteration.

It is well known that the fatigue process is controlled by local stress variations at the point where the crack growth takes place. It is also known that maximum bending stress in catenary risers will in most cases occur close to the touchdown point (TDP), and that the stress gradient is large in the vicinity of this point. Another fact is that vessel motions and riser dynamics will cause the TDP to move, and thus move the stress profile along the riser. Fatigue analyses must therefore consider true movement of the TDP as well as dynamic stress variations.

If a finite element method is used for stress analysis, one may model the pipe – sea floor interaction by introducing non-linear springs and dashpots at nodes with bottom contact. This is illustrated in Figure 1.2. Both the springs and dashpots are only active when there is sea floor contact at the node. This means that the nodes may lift of from the sea floor without experiencing extra tensile forces from the springs and dashpots. On the other hand, if the riser is pressed downwards against the seabed, the springs and dashpots will get active and prevent penetration of the sea floor by compression forces.

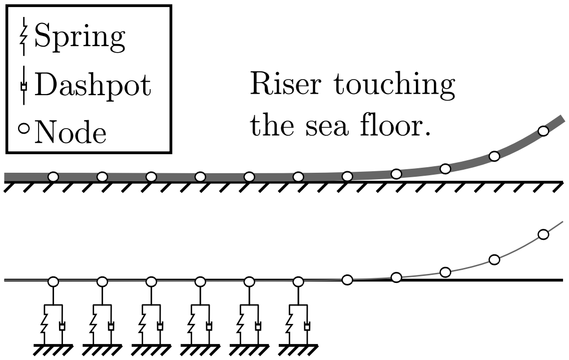


Figure 1.2: Pipe – sea floor interaction model.

This type of model will require a fully non-linear dynamic analysis in time domain. Such analyses are demanding with regard to computer resources and are therefore often replaced by linear approaches, which is used in this work. In linear approaches the springs and dashpots will not let go of the pipe if it is lifted from the sea floor, creating unrealistic tensile forces in the TDP area.

1.3 Time or frequency domain analysis

Empirical models used for VIV analysis are today almost exclusively based on the assumption that VIV will appear as a response at discrete frequencies. The benefit of applying a frequency domain method is that needed coefficients are found from fixed frequency experiments and hence directly applicable. A time domain approach needs in principle transient hydrodynamic coefficients or a strategy for frequency identification. This makes the hydrodynamic model complicated, but the advantage is that this model can be combined with a non-linear finite element code and thereby take into account the non-linear pipe – sea floor interaction. In [3] the authors describe how a new approach using both a linear frequency domain model for VIV calculations and a non-linear model for time domain analysis is used. The first steps were to carry out the VIV analysis according to linear theory and then introduce the calculated hydrodynamic forces to a non-linear structural model. The results showed that

- The combined use of frequency and time domain were well suited for investigating the influence from local non-linearities on stresses from VIV.
- The TDP represented a non-linear boundary condition for the catenary riser. Local effects may be important and lead to more fatigue damage than elsewhere on the riser.

- Differences were observed between global response found in time and frequency domain for catenary risers. There is hence a need to find out what caused this in order to improve the reliability of the methods.
- The stresses caused by contact forces between the riser and seabed were found by a linear model. However, since the nature of the contact is non-linear, the linear method may not give good results in a general case.

1.4 Non-stationary VIV of slender beams

Experiments done on VIV have shown that the response of slender structures at high modes will appear as a non-stationary response process. Amplitudes, dominating frequencies and mode composition are seen to vary in time and the complete understanding of this process is still not understood. It is also known that the response of slender structures under realistic flow conditions will appear as a stochastic process where both amplitudes and dominating frequencies vary in time. In order to describe this mechanism, there is a need to understand the multi-frequency VIV process. By using wavelet and modal analyses on experimental data, it is possible to describe the time variation of the peak frequency acting on the model, and also find the relative period of time this frequency falls into discrete frequency slots. The data from these experiments have resulted in two different ways of analysing multi-frequency VIV. The options are called time sharing and space sparing. Both ways are implemented in VIVANA and information about these methods can be found in [4] and in Section 2.9. In the work done, space sharing is the method used.

1.5 Ocean current

One of the major difficulties today regarding VIV fatigue analysis is the uncertainty connected with the way the ocean current is modelled. Models created for analysing the effects from VIV on non-symmetrical slender structures such as SCRs have shortcomings. The current can only be modelled as unidirectional over the water depth, and can only come either parallel to the riser plane or orthogonal to it.

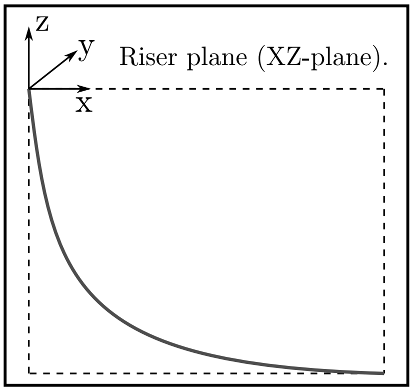


Figure 1.3: Steel catenary riser plane.

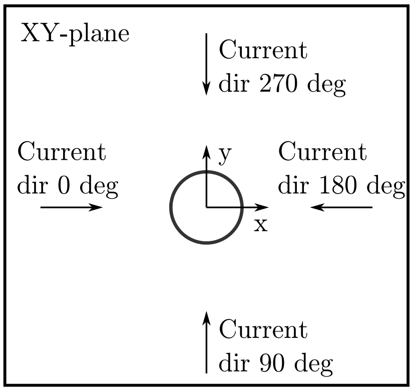


Figure 1.4: Definition of current direction.

A top tensioned riser has the advantage that it does not matter from which angle the current comes from since the forces will be the same either way. An SCR riser, however, will experience different response depending on which direction the current comes from. This is due to its non-symmetrical appearance. This is easy to observe by looking at Figure 1.5. As the water depth increases the curvature of the SCR will change and increase till the TDP. The top tensioned riser on the other hand will go straight down to the sea floor. Another challenge with the SCR is that it can only be analysed with four different current directions. This is a great problem, since the current from other directions will give excitation coefficients which are coupled, and this is still an unknown issue (prof. C.M Larsen, pers. comm.). The directions that may be used are two parallel directions to the riser plane (0° and 180°) and two orthogonal to the riser plane (90° and 270°), as seen in Figure 1.4. These current headings are investigated in the parameter variation analysis in this master thesis. The results indicated that the fatigue damage is different for current heading 0°, 90° and 180°. A current direction of 270° was not examined, as it would have given the same result as for 90°. This is explained by that the riser has the same projection either looking from a negative y-coordinate into the riser plane as for a positive y-coordinate.

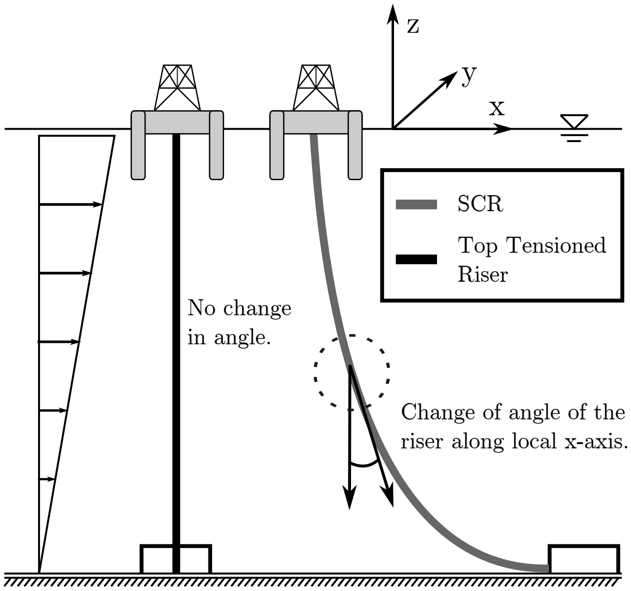


Figure 1.5: Top tensioned riser and steel catenary riser.

1.5.1 High velocity current in the North Sea

The information below is found from [5]. At the Troll oil and gas field outside the western part of Norway, large current velocities near $1.5 - 2.0$ [m/s] have been experienced in the upper ocean layer. These extraordinary flow speeds are due to a large outflow from the Skagerak into the North Sea. A strong, north-westerly wind in the North Sea that persists over time forces water into Skagerak. The outflow from Skagerak to the North Sea gets reduced in the same instance. This creates a pile-up of the water which may give a difference in sea water height around 1 [m]. When a change occurs in the wind direction, or a reduction of wind strength, the excess water will flow back into the North Sea and create a coastal jet. A complex pattern of vortices get formed around the southern tip of Norway and then they proceeds northwards along the coast. Based on direct measurements of the current in the Troll area, it has been estimated that such strong outflowing fronts occur 1-2 times per month on average. Most of the analyses in this work have used a current profile which has a maximum velocity of 1.478 [m/s].

1.5.2 Current variability

Offshore engineers involved in design and planning of marine structures and operations need information about the current flow at the cites where structures are to be installed. The long-term distribution of the current speed and direction is important, but the variability on short scale in time and space should also be considered for deep-water fields [5]. It has been a common practice to use averaging periods of 10 minutes and longer when recording the current flow directly. This is sufficient for observations of e.g. tidal components and other features of the flow which have a return period in order of hours. However, this is too long for resolving variations on time scales which correspond to dynamic response periods of marine structures.

Chapter 2
VIV

2.1 Introduction

The theory in this chapter is found from [6].

Slender marine structures like anchor lines, risers and free spanning pipelines that are exposed to ocean currents may experience structural vibrations. These vibrations are caused by forces from vortices that are shed from both sides of the pipe-structure. Hence the name, vortex induced vibrations. The vortex shedding is related to a full periodic cycle for the shedding process. This means that two vortices are shed per cycle, one from each side of the cylinder, as seen in figure 2.1.

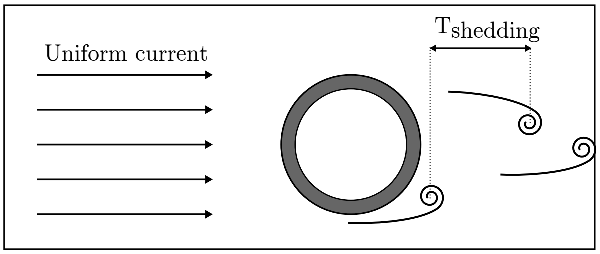


Figure 2.1: Vortex shedding.

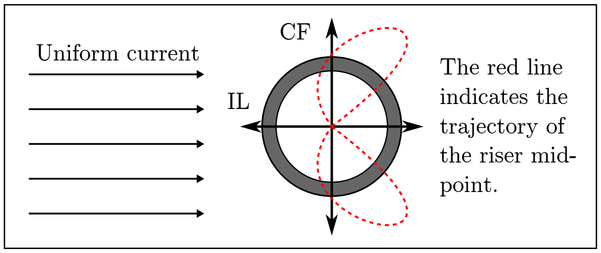

Uniform current
CF
IL
The red line indicates the trajectory of the riser mid-point.

Figure 2.2: CF and IL motion.

The shedding frequency is the frequency between two vortices, one on each side of the cylinder. Because these vortices change the pressure distribution around the pipe, there will be a development of forces. The structure will experience a lift force and a drag force. They are defined in local in-line (IL) and cross-flow (CF) direction, relative to the incoming flow [6]. This is shown in Figure 2.2. The lift forces will have the same frequency as the vortex shedding, while the drag forces will oscillate with twice the frequency. VIV is a vibration at resonance, and the vortex shedding frequency will increase for increasing current velocity. Since the IL frequency is twice the CF frequency, the structure will start to oscillate in the IL direction at a lower reduced velocity than CF.

Figure 2.3 displays the development of the amplitude ratio A/D for a pipe as the velocity increases. The y-axis is the amplitude ratio A/D, while the x-axis is the reduced velocity U_r.

The reduced velocity is given by equation

$$U_r = \frac{U}{D \cdot f_0} \tag{2.1}$$

where

U_r	[m/s]	Reduced velocity.
U	[m/s]	Current velocity.
D	[m]	Diameter of structure.
f_0	[1/s]	Eigenfrequency in still water.

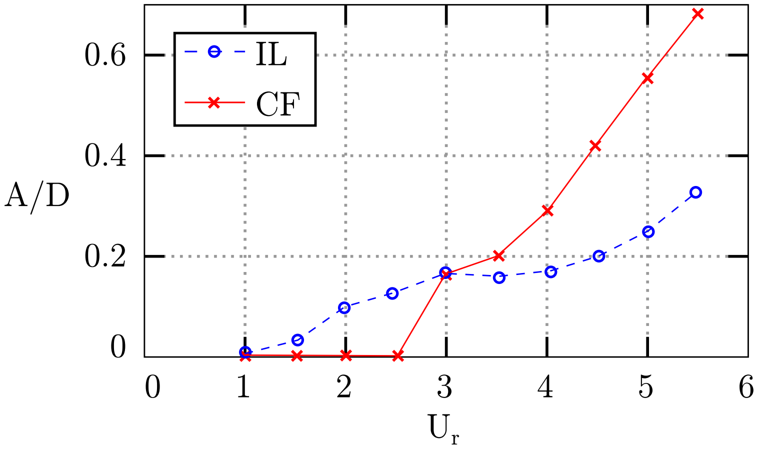


Figure 2.3: Amplitude ratio for a specific reduced velocity.

One can see from Figure 2.3, that the IL response starts at a lower reduced velocity, but the CF motions have the largest amplitude ratios.

2.2 Important remarks

The requirement of CF VIV is that the vortex shedding frequency is close to an eigenfrequency of the structure. In most practical cases the Strouhal number (St) will be close to 0.2. The Strouhal number is given by

$$St = \frac{D \cdot f_v}{U} \tag{2.2}$$

where

D Diameter of structure.
f_v Vortex shedding frequency.
U Current velocity.

Knowing that, one may say that VIV will occur if

$$U \geq 5Df_0 \tag{2.3}$$

Another important property of VIV is that it is a self-limiting response. When the vortex shedding process starts, it will transfer energy from the fluid to the structure. However, when the amplitude exceeds a certain level, the energy transport reverses and leads to damping of the response. One has seen that the response amplitude for a circular cylinder is limited to

$$\frac{A}{D} \leq 1.2 \tag{2.4}$$

This limitation is an approximation and not valid for all cross sections and Reynold numbers (Re). The Reynold number is a dimensionless number that gives a measure of the ratio of inertial forces to viscous forces and is given as

$$Re = \frac{UD}{\nu} \tag{2.5}$$

where

ν $[\mathrm{m^2 s^{-1}}]$ Kinematic viscosity.

The following two figures illustrate how the Strouhal number changes with the Reynolds number, and how the response amplitude ratio changes with reduced velocity. It is seen that the Strouhal number has a value of around 0.2 in a large interval of the Reynolds number, which is the reason for using this number in many calculations. VIVANA has built-in curves defining the Strouhal number at different Reynolds values. The values are displayed in Figure 2.4.

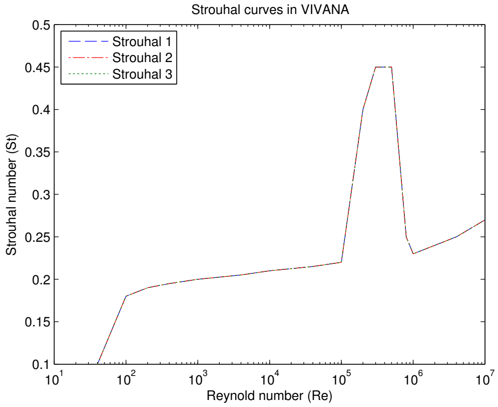


Figure 2.4: Strouhal numbers in VIVANA

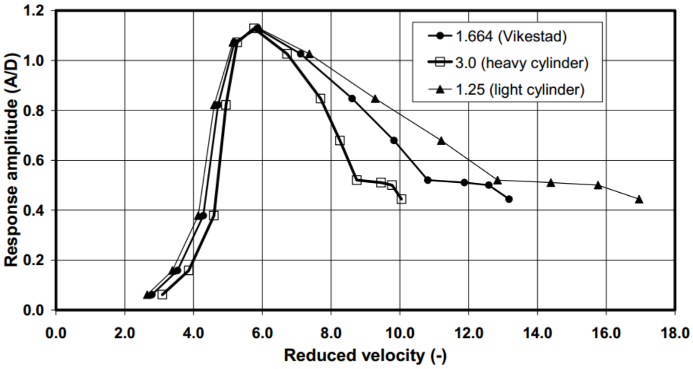


Figure 2.5: Response amplitude versus U_r. Courtesy of Larsen (2011).

VIV may occur in typical slender structures such as lines, risers, free spanning pipelines and conductors. The response will have a frequency close to an eigenfrequency of the structure, which will be linked to an associated mode shape. The vortex induced vibration will contribute to fatigue accumulation which is the one of the main topics of this work.

Details how VIV is analysed in VIVANA is given in Chapter 5.

2.3 Eigenfrequency

The natural frequencies are the fundamental frequencies connected to the undamped harmonic vibrations of the structure. The mode shape represents the waveform connected to a given natural frequency. Natural frequencies for a top tensioned riser are governed by two important factors; the top tension and the riser length. Considering the riser to be a vertical beam with constant tension, the natural frequencies ω_n (for mode n) are defined in [7] as,

$$\omega_n = \frac{n\pi}{l}\sqrt{\frac{T}{m} + \frac{n^2\pi^2}{l} \cdot \frac{EI}{m}} \tag{2.6}$$

where

ω_n	[rad/s]	Natural frequency n.
l	[m]	Length of riser.
T	[N]	Top tension.
m	[kg/m]	Mass per length.
n	[-]	Mode number.
E	[Pa]	Elastic modulus.
I	[m^4]	Second moment of area.

When the vortex shedding is close to an eigenfrequency of the riser, the VIV response will be excited. Figure 2.6 illustrates three modes of an imaginary structure which oscillate with frequencies ω, 2ω and 3ω.

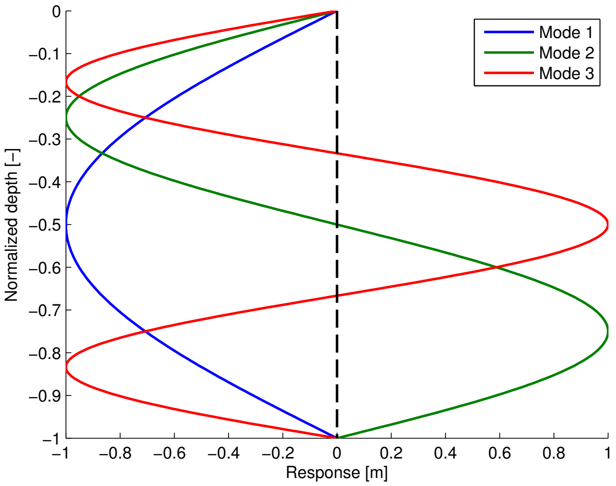


Figure 2.6: Example of three modes.

2.4 Excitation force

The force creating the CF vibration is called the dynamic lift force, or the CF excitation force. It is given by the equation

$$F_{e,CF} = \frac{1}{2}\rho C_{e,CF} D U^2 dl \tag{2.7}$$

where

ρ	[kg/m^3]	Fluid density.
$C_{e,CF}$	[–]	Cross flow excitation coefficient.
D	[m]	Structure diameter.
U	[m/s]	Current velocity.
dl	[m]	Length of a small section of the structure.

The excitation coefficient is traditionally mentioned as the lift coefficient in VIV literature. It is defined as the component of the hydrodynamic force that is in phase with the cross flow velocity of the structure. The value can be positive, which means that energy is added to the motion. The coefficient can also be negative which will lead to damping. This is easy to see from Figure 2.5, where the amplitude of the motion first increases until a certain value and then gets smaller. This implies that the excitation coefficient is dependant on the response amplitude. In VIVANA, the empirical lift coefficient curve is defined as illustrated in Figure 2.7.

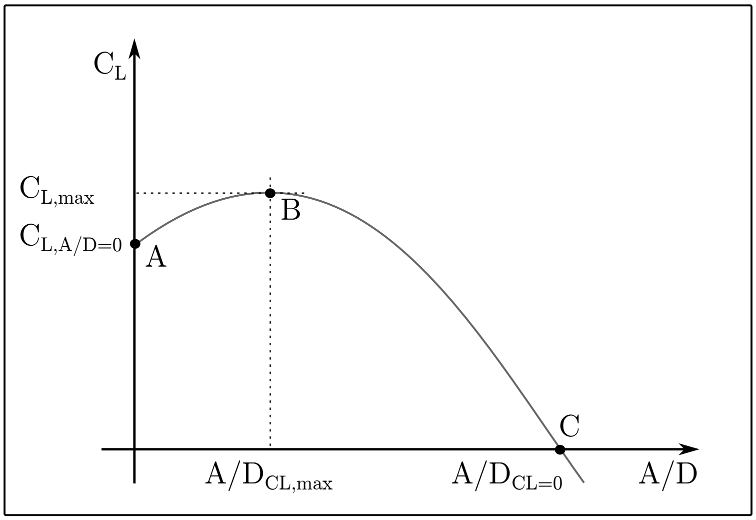


Figure 2.7: Empirical lift coefficient curve.

2.5 Free oscillation test

To understand how VIV works, experiments need to be done that can help us understand the physics of the phenomenon. One of these experiments is called the free oscillation test. The free oscillation test is performed by having an elastically supported rigid cylinder in constant current. When the current acts on the cylinder it will force the pipe to vibrate. When these test are performed, three different frequencies are found.

The still water frequency

$$f_0 = \frac{1}{2\pi}\sqrt{\frac{k}{m + m_{a0}}} \tag{2.8}$$

where

k	[N/m]	Spring stiffness.
m	[kg]	Mass of the structure.
m_{a0}	[kg]	Added mass.

The vortex shedding frequency

$$f_v = \frac{St \cdot U}{D} \tag{2.9}$$

The terms in the equation are explained in Section 2.2.

The oscillating frequency

$$f_{osc} = \frac{1}{2\pi}\sqrt{\frac{k}{m + m_{a0}}} \tag{2.10}$$

where the added mass is valid for the actual flow and oscillation condition.

It is noticed that the oscillation frequency is not identical to the eigenfrequency in still water since the added mass changes depending on the flow and oscillation conditions. The added mass is defined as the component of the hydrodynamical force that is in phase with the CF acceleration of the structure. There is a possibility for negative added mass, which is only a consequence of the phase between the motion of the pipe and the total hydrodynamic force. By performing free oscillation tests of rigid cylinders it is possible to get information about parameters such as

- Cross-flow amplitudes and frequencies.
- In-line amplitudes and frequencies.
- Drag coefficient for an oscillating cylinder.

When the tests are performed it is possible to measure the forces acting on the cylinder. The added mass as a function of reduced velocity may also be found. It is important to note that CF and IL excitation coefficients cannot be found during this type of test because they are zero during the free oscillation. One way of dealing with that is to perform forced oscillation tests. In this way a complete set of coefficients for any combination of frequency and amplitude may be obtained.

2.6 Forced oscillation test

This test is performed by oscillating a cylinder in a stationary and uniform flow with prescribed motions. These motions may be harmonic motions in CF and IL direction, or a combination of these. Examples of CF and IL motion may be,

$$\begin{aligned} x &= x_0 sin(4\pi\omega_{osc}t - \epsilon) \\ y &= y_0 sin(2\pi\omega_{osc}t) \end{aligned} \tag{2.11}$$

where

x and y	[m]	Motions in IL and CF directions.
x_0 and y_0	[m]	Amplitudes of IL and CF.
ω	[rad/s]	Oscillating frequency.
t	[s]	Time.
ϵ	[rad]	Phase angle between IL and CF.

When oscillating the pipe, as described by the equations above, the path of the cross-section will be an eight-figured motion as illustrated in Figure 2.2 and 2.8. The forces can then be measured and by processing the data, one can identify force components that are in phase with the forced motion accelerations and velocities in the CF and IL direction. It is then possible to find the added mass, lift, drag and dynamic force coefficients of the cylinder.

An example of what happens when there is a phase angle between the CF and IL motion is also illustrated. The equations from above were used to create Figure 2.8.

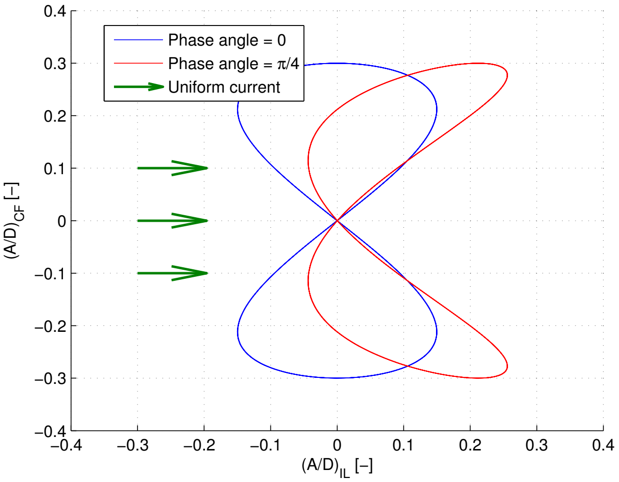


Figure 2.8: Examples of CF + IL trajectory.

The results from the forced motion test are usually presented as contour plots for the coefficients in a non-dimensional amplitude/frequency plane.

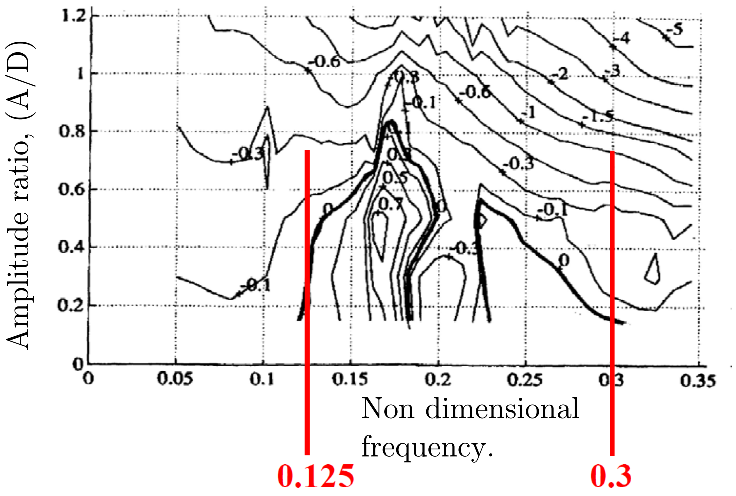


Figure 2.9: Contour plot of CF coefficient. Courtesy of Larsen (2011)

Figure 2.9 illustrates the CF excitation coefficient for tests done by Gopalkrishnan [8]. These coefficients are often applied in empirical methods to calculate the response in CF direction. IL motions are known to influence the CF response, however, it is still reasonable to use the CF coefficients since the CF response is less sensitive to the response in its orthogonal direction than IL is [6]. The values between 0.125-0.3 [-] are used in the VIVANA software. It means that all eigenfrequencies that give a non-dimensional frequency between these two values are taken as response frequency candidates.

2.7 Added mass

When a structure becomes excited by vortex shedding in a current that is not uniform, the added mass will vary along its length. The response frequency will appear as an eigenfrequency that is influenced by this added mass distribution. However, since added mass depends on the frequency, the correct distribution of added mass cannot be found directly. Consistency between the frequency and added mass distribution must therefore be obtained through an iterative process for each response frequency candidate. In this work the added mass coefficient value has been set to a constant value of 0.8. This is done after a discussion with the supervisor. Choosing this value gives the best prediction for high mode VIV (prof. C. M. Larsen, pers. comm).

The positive outcome by choosing a constant value of the added mass, is that iteration is unnecessary to find consistency between the correct added mass and the response frequency. This leads to much shorter computation time. Both methods have been tried out in this work, and the computation time went from 20 to 2 minutes per simulation.

2.8 Suppression of VIV

There are several ways of reducing the vibrations from vortex shedding. One way to avoid VIV is by not being in the resonant area. This is possible if the highest Strouhal frequency for the cross section is lower than the first natural eigenfrequency. Being a great deal above the resonant area is complicated. This is because there will always be a higher natural mode that corresponds with to the vortex shedding frequency [9].

Another way of reducing the response is to add suppressing devices to the riser. Examples are fairing or helical strakes attached to the riser as illustrated in the following figure.

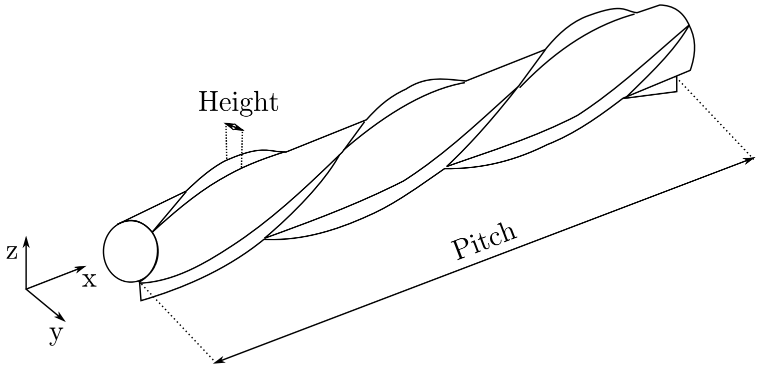


Figure 2.10: Riser with helical strakes.

2.9 Time and space sharing

The characteristics of the VIV response for long, slender structures subjected to a non-uniform current are complicated, and difficult to model. It is observed that the response of the structure will be a mixture of different modes and frequencies. This response will change its characteristics with varying current profile, order of dominating modes, influence from tension and bending stiffness, mass ratio, Reynlds number and most probably other parameters as well [6]. There is no model today that can take all of these effects into account and replicate the response that has been observed. The approach that has been used, is to define an excitation zone for a specific frequency that is initiated, and calculate the response of it independently of other active frequencies. The key has been to define this excitation zone. Two different approaches exist. They are called

Time sharing: The competing frequencies will dominate one period of time. Since there is a set of frequencies, the domination will shift. The analysis needs a way of finding the relative duration for each competing frequency.

Space sharing: When using the space sharing option, all frequencies will act simultaneously, but every frequency will have a designated length where they are excited. This means that the structure length will be shared among the set of response frequencies that are acting.

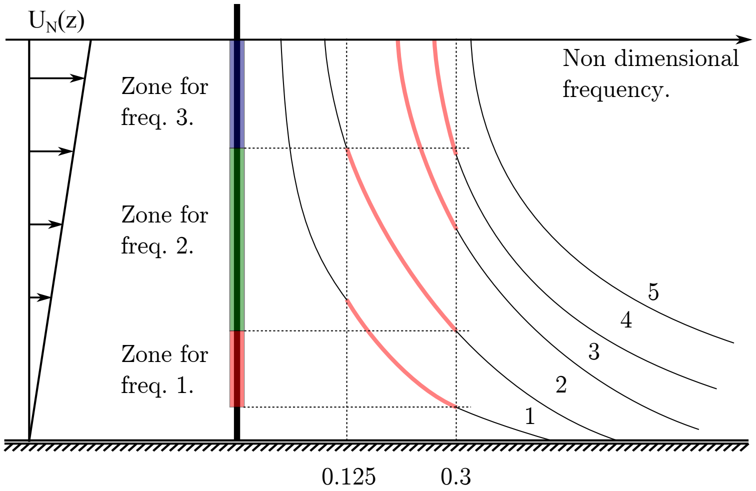


Figure 2.11: Excitation zones for space sharing frequencies.

In Figure 2.11, it is illustrated how the excitation zones are defined when using space sharing. It is seen that some zones overlap each other. This means that the excitation zones may be different between space and time sharing, something which will give an effect on the fatigue damage. The zones are defined by specifying an active interval for the non-dimensional frequency. They are then ranked by using an energy criterion given as

$$E_i = \int_{L_{e,i}} U^3(z) D_H^2(z) \left(\frac{A}{D}\right)_{C_e=0} \mathrm{d}z \tag{2.12}$$

The excitation parameter is integrated over the excitation zone for every frequency. The $(A/D)_{C_e=0}$ is the non-dimensional amplitude where the excitation coefficient shifts from a positive to negative value. This is shown as point C in Figure 2.7.

The time sharing process is shown in the following figure. It is illustrated how the frequencies compete over the time.

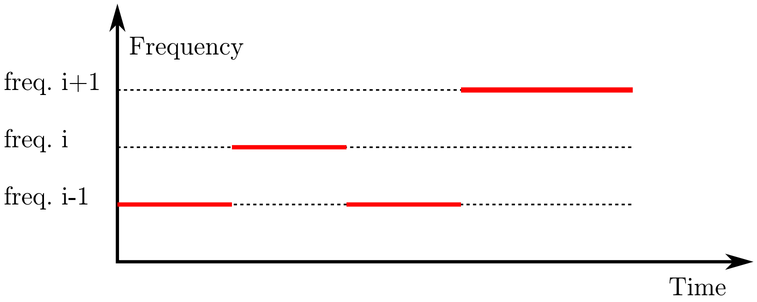


Figure 2.12: Time sharing process.

The same energy criterion is used to rank the response frequencies, and the duration of each frequency is found by

$$T_i = T \cdot \frac{E_i}{\sum_{n=1}^{k} E_n} \tag{2.13}$$

where

T_i	[s]	Time duration of response frequency i.
T	[s]	Total time.
E_n	[m^7/s^3]	Excitation parameter for frequency n.

In time sharing, the response frequencies will be active over their entire excitation zone. However, they will not be active at the same time. The excitation zones for time sharing are illustrated in Figure 2.13.

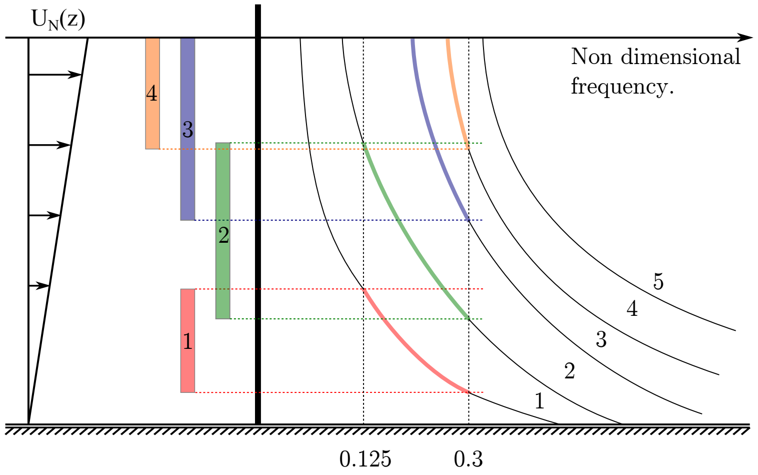


Figure 2.13: Excitation zones for time sharing frequencies.

Chapter 3

Soil Interaction

3.1 Introduction

One of the great challenges with SCR design is the assessment of fatigue damage due to repetitive loading over the lifetime of the riser. This depends significantly on the assumed pipe–soil interaction behaviour at the location where the riser reaches the seabed surface. This is generally known as the touchdown zone. There is still considerable uncertainty over the riser–soil mechanics in this region and it is of major concern to the offshore industry. The pipe–soil interaction response is dependant on a range of parameters, such as seabed soil strength, loading conditions and pipe displacement magnitude. At the touchdown area the riser is typically modelled as a riser with a series of springs [10] as shown in figure 3.1.

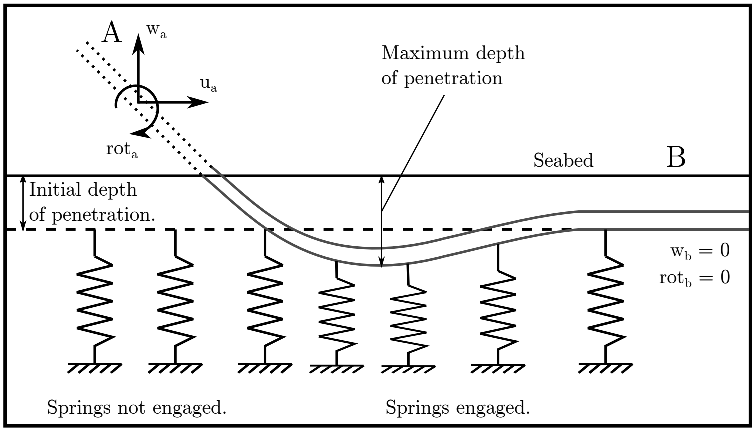


Figure 3.1: Touchdown zone interaction.

As the riser is laid, it will penetrate a certain distance into the seabed. The enhanced embedment is a consequence of two effects which occur during pipe laying.

- Concentration of pipe-soil contact stress in the touchdown zone.
- The dynamic motion of the pipe.

A distance from the TDP, point B on Figure 3.1, the curvature will become zero and the vertical displacement relative to the initial penetration will also be zero. The loads on the riser are initiated by the motion of the floater and the ocean current acting on the riser structure. They will result in cyclic rotations as well as vertical and horizontal displacements at a location above the seabed. This can be seen in Figure 3.1 as point A.

3.2 Pipe–soil response

Hodder and Byrne investigated the pipe–soil response between point A and B [11]. They did an experiment on the pipe–soil interaction in a long flume at the Oxford University. There, it was possible to have a bed of soil that was fully liquefied. On the pipe they tested, an actuator was installed on one end to apply specified displacements at point A. The actuator was controlled by a computer that made it possible to move the pipe end with either a constant velocity, or cyclically in the form of sine waves with different amplitudes and frequencies.

To compare the pipe–soil interaction force experienced by an element of pipe during the experiment with a numerical model, it was necessary to quantify the distribution of the vertical soil reaction throughout the touchdown zone. To create an appropriate form of the distribution of the soil bearing pressure below the riser model, a simple numerical model was constructed using the finite element program ABAQUS. The model allowed for a comparison between the numerical and experimental results. The soil was modelled as a series of linear springs at discrete nodes that were able to be released if the pipe was lifted above the soil surface. It was crucial to model the pipe that way, since disabling the pipe to be lifted would have given extra stresses in the area.

In the experiment by Hodder and Byrne, the stiffness k for invert elevations was calculated as a secant from the theoretical bearing capacity curve for a strip footing in drained soil with width B [11]. Due to the linearity of the penetration curve, k was assumed to be constant as seen from figure 3.2.

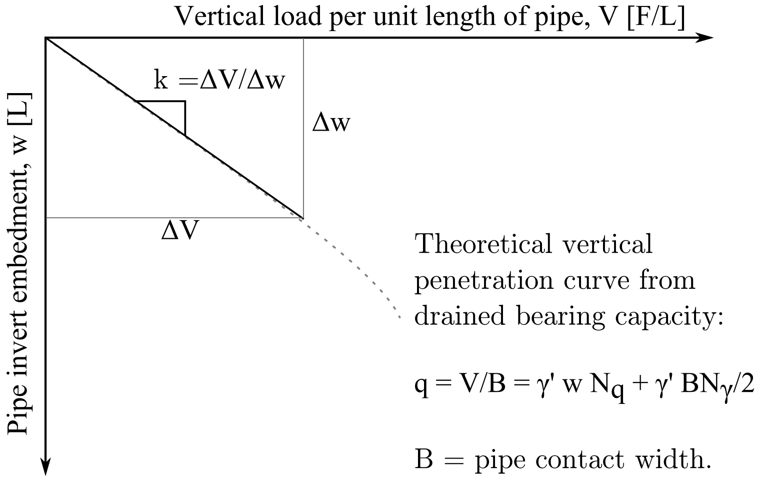

Vertical load per unit length of pipe, V [F/L]
k =ΔV/Δw
Δw
ΔV
Pipe invert embedment, w [L]
Theoretical vertical penetration curve from drained bearing capacity:
q = V/B = γ' w Nq + γ' BNγ/2
B = pipe contact width.

Figure 3.2: Penetration curve for determination of linear spring stiffness.

The experimental data gave results enabling the researches to back-calculate the bearing pressure distribution and touchdown point, given the bending moments at discrete points along the riser model. The bending moments were calculated by placing strain gauges at these discrete points. The numerical results provided insight into the load the soil experienced at various positions along the pipe.

Initially, the soil reaction was equal to the soil weight. As the pipe was lifted, the results showed that the soil reaction did not decrease immediately along the pipe, but increased due to the bending stiffness before going to zero. This showed that the load transmitted between the soil and the pipe was a combination of the load and displacement and is complex.

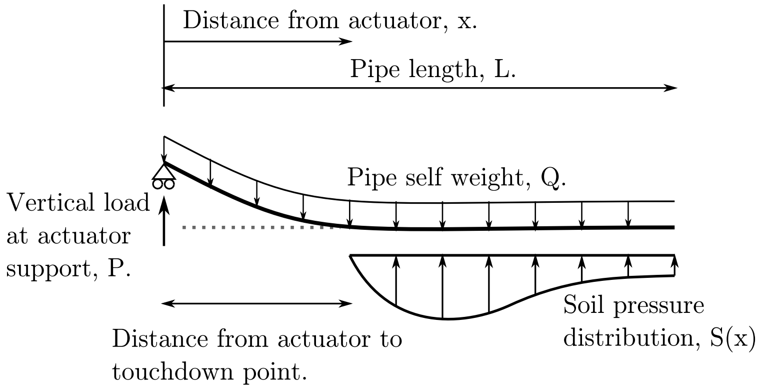

Distance from actuator, x.
Pipe length, L.
Pipe self weight, Q.
Vertical load at actuator support, P.
Soil pressure distribution, S(x)
Distance from actuator to touchdown point.

Figure 3.3: Free body diagram of riser model and soil.

The experiment done by Hodder and Byrne was the first presented experiment of a riser–soil interaction problem in a laboratory exploring three-dimensional effects. The instrumentation was used to quantify riser performance, trench formation and the development of excess water/pore pressure. The results showed that it was possible to back-calculate an estimate of the touchdown point and the distribution of the soil reaction that occurred during the experiment.

3.3 Dynamic interaction of catenary risers with the sea floor

Another study was done by Katifeoglou and Chatjigeorgiou [12] on the dynamic interaction between a catenary riser and the sea floor. They made a global dynamic behaviour of a SCR. The sea floor–structure phenomenon was given particular attention. The geotechnical computations showed an amplification of the reaction force with the depth of penetration and the arc of the contact edge. They made several assumptions due to the uncertainties with the interaction problem. The numerical experiments they performed showed that the primarily affected component, the bending moment, was the most sensitive factor for fatigue.

Chapter 4

Fatigue

4.1 Introduction

Fatigue occurs when a material is subjected to a loading and unloading which is of repetitive nature. If these loads are above a certain limit, there will be an initiation of microscopic cracks. When the crack growth is allowed to proceed, the structure may suddenly fracture. The cyclic load that is applied on the structure and which initiates the fatigue is less than the ultimate tensile stress limit, and may be below the yield stress limit of the material. Examples of cyclic loading important to offshore structures can be loading from waves, or as in this work, loading induced by vortex shedding.

4.2 SCR fatigue design

Fatigue damage is normally estimated by using either S-N curves or fracture mechanics. Today, almost all fatigue analyses of SCRs are performed using S-N curves [13]. When using S-N curves, the fatigue damage is estimated by using the Palmgren-Miner rule summation as shown in Equation 4.14. The S-N curves form the basis for description of the SCR fatigue capacity. The curves relate the number of stress cycles to failure to the corresponding stress range including the effects of stress concentration factors (SCF). The relevant S-N curve applicable to risers have either a single slope or a two-slope (bilinear) behaviour. The single sloped curve, which is mostly used can be expressed as

$$N = a \cdot S^{-k} \tag{4.1}$$

where a and k are parameters defining the curve which are dependant on the material and details obtained from fatigue experiments. S is the stress range. Most of the S-N data are derived by fatigue testing of small specimen in test laboratories. For simple test specimens, the testing is performed until the specimens have failed. In these specimens there is no possibility for redistribution of stresses during crack growth. This means that most of the fatigue life is associated with growth of a small crack that grows faster and faster as the crack size increases until fracture [14].

4.3 Selection of SN curve

From [14] we know that for practical purposes, welded joints are divided into several classes, where each class has its own S-N curve. Tubular joints such as risers are assumed to be class T. Other types of joints, including tube to plate, may fall in one of the other 14 classes defined in the recommended practice (RP-C203). Each construction detail where fatigue cracks potentially may develop should be placed into its relevant joint class. In any welded joint, there are several locations at which the fatigue crack may develop. This may be at the weld toe in each of the parts joined, at the weld ends and at the weld itself. Each location should be classified separately.

The basic design SN curve is given as

$$log(N) = log(\bar{a}) - m \cdot log(\Delta\sigma) \tag{4.2}$$

where

N	[–]	Predicted number of cycles to failure for stress range $\Delta\sigma$.
$\Delta\sigma$	[MPa]	Stress range.
m	[–]	Negative inverse slope of SN curve.
$log(\bar{a})$	[–]	Intercept of log N-axis by SN curve.

The fatigue strength is somewhat dependant on the plate thickness. This effect is determined by the local geometry of the weld toe in relation to the thickness of the adjoining plates. The thickness effect is accounted for by a modification of the stress such that the design S-N curve for thickness larger than the reference thickness is given by

$$log(N) = log(\bar{a}) - m \cdot log\left[\Delta\sigma\left(\frac{t}{t_{ref}}\right)^k\right] \tag{4.3}$$

where

t_{ref}	[mm]	Reference thickness is 25 [mm] for welded connections other than tubular joints. For tubular joints the reference thickness is 32 [mm]. For bolts, the thickness is 25 [mm].
t	[mm]	Thickness through which a crack will most likely grow. $t = t_{ref}$ is used for thicknesses less than the refernese thickness.
k	[–]	0.1 for tubular butt welds made from one side.

An example of a single sloped S-N curve is illustrated in Figure 4.1. The S_2-value is the stress value that after 10^7 cycles with the indicated stress, will give fracture. It is also seen that a higher stress value will need a smaller number of cycles to fracture.

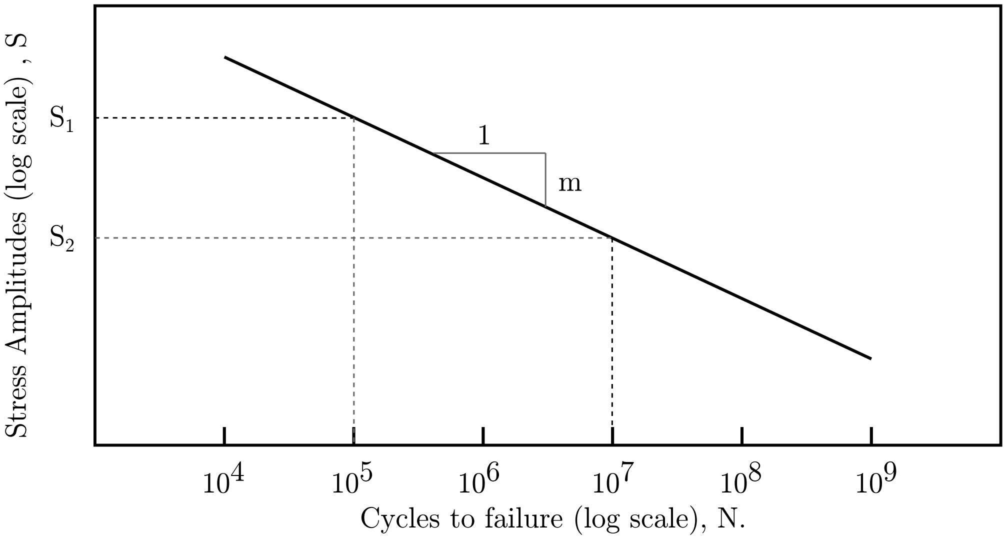


Figure 4.1: Example of SN curve

In general the thickness exponent is included in the design equation to account for a situation where the actual size of the structural component considered is different in geometry than the S-N data are based on. To some extent it also accounts for the size of the weld and the attachment. However, it does not account for weld length or length of components different from the ones tested. An example is a mooring system where the number of chain links in the actual mooring are significantly larger than in the test case. The size effect should therefore be carefully considered using probabilistic theory to achieve reliable design.

4.4 Effect of floater type on SCR fatigue

Typical SCR fatigue analyses include first order and second order floater motion induced fatigue, VIV fatigue, deep draft floater vortex induced motion (VIM) fatigue, slugging fatigue, fatigue accumulated by start-up and shut-down, installation and other kinds of dynamic loading imposed to the SCR [13].

Depending both on the geographical location and floater type, the fatigue damage might also vary from one floater concept to another. The author from [13] listed in his article which primary factors to fatigue are of great interest for different vessel types.

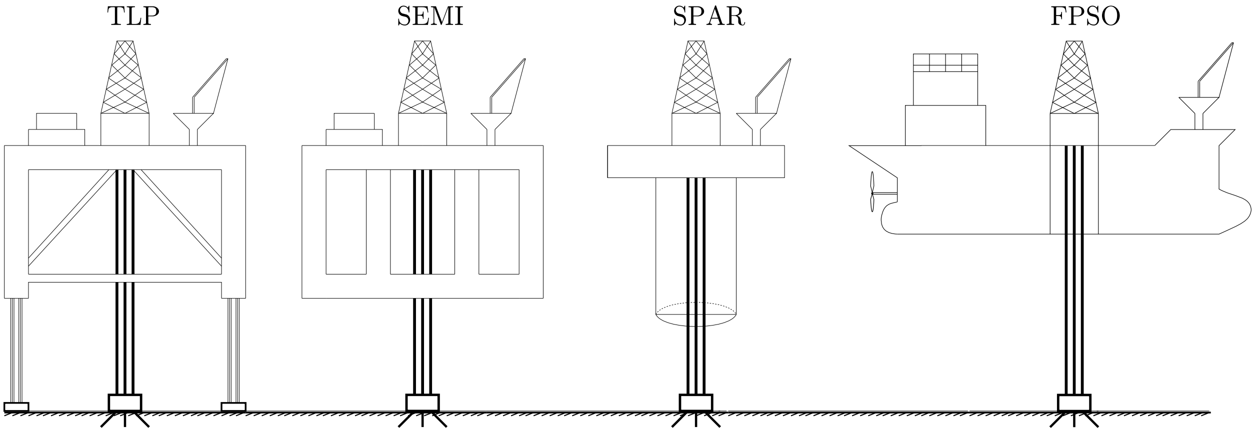


Figure 4.2: Examples of offshore drilling types.

- SPAR buoy: VIM fatigue.
- Semi-submersible: Motion fatigue and heave induced VIV fatigue.
- Deep draft semi-submersible: Motion fatigue and VIM fatigue.
- TLP: Motion Fatigue.
- FPSO: Motion fatigue.

4.5 Fatigue Reliability Analysis

Fatigue design of a steel catenary riser at the touchdown point is a challenging problem. One of the greatest challenges is the fatigue assessment near the touchdown point. The TDP will often experience the worst cyclic loads, is inaccessible for inspection and is subjected to many uncertainties.

Even though there have been several advancements in the development of realistic soil models, there are still many uncertainties because of the complexity of soil mechanisms. Some examples are that models are built on idealized loading sequences which may not represent what is encountered by a riser. Another example is that empirical coefficients have not been estimated to precision, since they are obtained from specific soil conditions. Acquisition of actual site data might be difficult, especially in great depths. Another problem is how the touchdown point is effected by vortex induced vibrations. Because of the uncertainties in riser fatigue analysis, a safety factor of 10 is recommended on the predicted fatigue life for critical structural components [15].

Li and Low [16] did a reliability analysis to find what parameters were of importance for fatigue assessment. They were motivated to do this due to the lack of fatigue reliability analysis that addressed the seabed uncertainties. It was important for the researchers to identify the critical parameters that contributed to the uncertainty of fatigue damage. This was succeeded by characterizing the probability distributions of the parameters, something which was not straightforward. In their research they used a fatigue methodology that was based on a S-N curve that relates the stress range to the number of cycles to failure. For irregular stress, the rainflow counting algorithm was used, and the Palmgren-Miner rule was used to sum up the total damage.

The basic idea behind the study was to observe how the fatigue damage changed when one of the critical parameters was varied. In Table 4.1 it is displayed which parameters were used in the sensitivity study.

No.	Variable	Distribution	Mean	CoV	Recommended CoV
1	Trench	Beta	0.50	0.30	0.10 - 0.30
2	Soil suction	Beta	0.40	0.30	0.10 - 0.30
3	Soil stiffness	Lognormal	1 [kPa]	0.50	0.20 - 0.50
4	Drag coefficient	Lognormal	0.80	0.20	0.15 - 0.20
5	RAO amplitude factor	Lognormal	1.00	0.10	0.05 - 0.10
6	Static floater offset	Normal	0.00	$\sigma = 0.5$ [m]	1% of depth
7	Riser mass	Normal	182.5 [kg/m]	0.10	0.05 - 0.10
8	VIV parameter	Beta	0.20	0.15	Nil
9	S-N parameter A	Lognormal	$1.205 \cdot 10^{16}$	0.50	0.50
10	Wall thickness	Normal	0.023 [m]	0.03	0.03
11	Outer diameter	Normal	0.356 [m]	0.002	0.002

Table 4.1: Parameters for the sensitivity study.

The riser data used in the study are shown in table 4.2. A JONSWAP sea state with significant wave height (H_s) 4.3 [m] and peak period (T_p) 14 [s] was used.

Parameter	Value	Parameter	Value
Riser length [m]	1 700	I [cm^4]	$3.467 \cdot 10^4$
Outer diameter [mm]	356	E [N/m^2]	$2.1 \cdot 10^{11}$
Wall thickness [mm]	24	C_D	0.8
Mass [kg/m]	182.5	C_M	1.8
Area [cm^2]	250.3	Hang off angle [°]	14

Table 4.2: Riser data in parameter study

After running analyses with the computer program OrcaFlex, the authors were able to see how the different parameters effected the riser. The data showed that:

- The seabed variables were influential at the TDP. The reason was that the seabed model acted as a boundary condition. The authors meant that this was critical to the structure near the boundary, and did not effect the global SCR response.

- The static offset was another variable that had great influence near the TDP. This was explained by that the offset modified the riser configuration at the seabed.

- The middle section was mostly affected by the drag coefficient and the riser mass, which controlled the damping and the natural frequencies.

- The response amplitude operator (RAO) and VIV had the most effect at the top. The RAO controls the vessel motions, and the current is largest at the surface. They saw that near the TDP, "cushioning" of the seabed tended to buffer the changes in the loading condition.

- The S-N parameter showed most variability in the fatigue damage. However, these uncertainties are generally accounted for in fatigue design, the authors explained.

- The wall thickness and outer diameter had negligible effect on the damage uncertainty, and they are also easy to determine.

- The result showed that the fatigue damage increased with soil stiffness, since soft soil would distribute the stress over a longer section.

- The trench depth was favourable for the fatigue since it allowed for a more natural transition for the riser.

- The soil suction showed an negative effect. This was due to the extra loads on the riser during uplift from the trench.

The authors made a table showing where each parameter had its main effect. In this way, they were able to set a category on the parameters. The categories were seabed, general and deterministic.

Variable	CoV of fatigue damage			Category
	TDP	On top	Middle part	
Trench	0.090	0.002	0.003	Seabed
Soil suction	0.088	0.003	0.004	Seabed
Soil stiffness	0.156	0.004	0.008	Seabed
Drag coefficient	0.066	0.089	0.06	General
RAO amplitude factor	0.081	0.041	0.033	General
Static floater offset	0.314	0.442	0.298	General
Riser mass	0.512	0.511	0.512	General
VIV parameter	0.051	0.038	0.201	General
S-N parameter A	0.084	0.187	0.258	General
Wall thickness	0.003	0.004	0.003	Deterministic
Outer diameter	0.001	0.007	0.007	Deterministic

Table 4.3: Results from the sensetivity study.

4.6 Fatigue Analysis in VIVANA

4.6.1 Introduction

VIVANA calculates the fatigue damage assuming that the cross section is circular with a diameter defined from the cross section data from SIMA. This is given in the input file sima_inpmod.inp. The stresses are calculated on the outer surface.

The method for calculating the fatigue depends on the analysis options which are presented in the table below.

Fatigue analysis:	**VIV analysis option:**		
	Pure IL	Pure CF	Combined CF and IL
Space sharing	1	1	1
Time sharing	2	2	3

Table 4.4: VIVANA methods for fatigue calculation

The numbers in the table above are explained by the following.

1. Generation of time histories, rainflow counting.

2. Closed form solution.

3. Closed form solution (Gouyang 1989).

In this work, only CF has been investigated and space sharing has been used. This means the rainflow counting is used (option 1).

4.6.2 Generation of time series

When the VIV analysis is finished, there are N complex response vectors $\mathbf{x}_i$, where $i = 1 \dots N$. By using them in combination together with the element stiffness matrices and the cross section properties, one can get time series of the stress at the element ends. The stress cycles are counted by the rainflow method and the fatigue damage is found by the Palmgren-Miner summation.

The procedure follows these steps.

1: Complex vectors of element forces for all elements at all response frequencies are calculated by

$$\boldsymbol{s}_{i,iel} = \boldsymbol{k}_{iel} \cdot \boldsymbol{x}_{i,iel} \tag{4.4}$$

Where $i = 1, 2 \dots, N$ and $iel = 1, 2 \dots, N_{elements}$. $\mathbf{k}_{iel}$ is the element stiffness matrix and $\mathbf{x}_{i,iel}$ contains the components from the total $\mathbf{x}$-matrix, and represents the local degrees of freedom to element iel. The dimension of the different components is; $\mathbf{s}$ and $\mathbf{x}$ is 12×1, $\mathbf{k}$ is 12×12.

2: The stresses at the element ends are found by

$$\sigma_{i,iel,iend,ipt} = \boldsymbol{p}_{iel,ipt}\boldsymbol{s}_{i,iel,iend} \tag{4.5}$$

$ipt = 1, 2, \dots, N_{pt}$ and $iend = 1, 2$. $\boldsymbol{s}_{i,iel,iend}$ is a vector of dimension 6 at each end of the element (iend = 1 and iend = 2). $\boldsymbol{p}_{iel,ipt}$ is a vector of dimension 6 containing the cross section properties of the element and N_{pt} is the number of places on the cross section that needs to be checked for fatigue. $\boldsymbol{p}_{iel,ipt}$ is given by

$$\boldsymbol{p}_{iel,ipt} = \begin{bmatrix} \frac{SCFAX}{A_{st,iel}} & 0.0 & 0.0 & 0.0 & \frac{SCFMY}{W_{y,iel,ipt}} & \frac{SCFMZ}{W_{z,iel,ipt}} \end{bmatrix} \tag{4.6}$$

where

$$W_{y,iel,ipt} = \frac{W_{y,iel}}{sin} \cdot (\frac{2\pi \cdot ipt}{N_{pt}}) \tag{4.7a}$$

$$W_{z,iel,ipt} = \frac{W_{z,iel}}{cos} \cdot (\frac{2\pi \cdot ipt}{N_{pt}}) \tag{4.7b}$$

$A_{st,iel}$ is the cross section steel area and $W_{y/z}$ are the section modulus for bending about the y- and z-axis. SCFAX, SCFMY and SCFMZ are stress concentration factors for axial tension, bending about local y-axis and bending about local z-axis.

3: The stresses at the elements are represented as complex numbers, and the time series of the stress can be written as a summation of the contributions from all response frequencies,

$$\sigma_{iel,iend,ipt}(t) = \sum_{i=1}^{N} \sigma_{i,iel,iend,ipt} \cdot e^{i(\omega_i t + \epsilon_i)} \tag{4.8}$$

ω_i is the response frequency number i, and ϵ_i is the phase angle at the respective frequency and is a random number between 0 and 2π.

The complex number can be written as

$$\sigma = \sigma_{Re} + i\sigma_{Im} \tag{4.9}$$

In the end, the stresses are given by

$$\sigma_{iel,iend,ipt}(t) = \sum_{i=1}^{N} \sigma_{amp,i,iel,iend,ipt} \cdot cos(\omega_i t + \theta_{i,iel,iend,ipt} + \epsilon_i) \tag{4.10}$$

where

$$\sigma_{amp,i,iel,iend,ipt} = \sqrt{\sigma^2_{Re,i,iel,iend,ipt} + \sigma^2_{Im,i,iel,iend,ipt}} \tag{4.11}$$

$$\theta_{i,iel,iend,ipt} = atan(\frac{\sigma_{Im,i,iel,iend,ipt}}{\sigma_{Re,i,iel,iend,ipt}}) \tag{4.12}$$

4: In this step, a time series of length T [s] is generated. Then one has to define a number of stress ranges $\Delta\sigma_i$, where $i = 1, 2, \ldots, N_{\Delta\sigma}$. The rainflow counting method is then used for finding the number of occurrences for each cycle $n_{i,iel,iend,ipt}$.

The total number of cycles for stress range i during one year is then calculated as

$$n_{i,year,iel,iend,ipt} = \frac{n_{i,iel,iend,ipt} \cdot 365 \cdot 24 \cdot 60 \cdot 60}{T} \tag{4.13}$$

The accumulated damage is then calculated by the Palmgren-Miner sum:

$$D_{iel,iend,ipt} = \sum_{i=1}^{N_{\Delta\sigma}} \frac{n_{i,year,iel,iend,ipt}}{N_i} \tag{4.14}$$

N$_i$ is the number of cycles to failure for stress cycle i, and may be found by use of the following equation

$$logN_i = logC + m \cdot log\left[\Delta\sigma \cdot \left(\frac{t_{iel}}{t_{ref}}\right)^k\right] \tag{4.15}$$

$logC, m, t_{ref}$ and k are found from the S-N curve used in VIVANA, t_{iel} is the wall thickness for the cross section and $\Delta\sigma_i$ is the stress range.

Chapter 5
Software

This chapter gives a short introduction to the different computer programs used in the thesis.

5.1 SIMA and RIFLEX

RIFLEX was developed as a tool for analysis of flexible marine riser systems. However, it is well suited for other type of slender structures as well. These slender structures are mooring lines, umbilicals, steel pipelines and conventional risers. In the thesis the SIMA program is used to model the SCR riser. SIMA has a graphical interface which helps the user get easily familiar with the software. RIFLEX is used for calculations of the model created in SIMA. RIFLEX gives the user the possibility to calculate the structural response. The program may compute static and dynamic characteristics of the structure [17].

5.2 SIMO

This computer program was used by the supervisor in DNV that produced the vessel motion coordinates. Details about SIMO are not given here.

5.3 VIVANA

VIVANA is a computer program for evaluation and analysis of vortex induced vibrations in slender marine structures. We know from experience that an ocean current might cause VIV on slender marine structures. VIVANA requires RIFLEX to do a static analysis first. A brief introduction how VIVANA does its calculations is given in the following section.

5.4 Analysis with RIFLEX and VIVANA

A flow chart illustrating which computer programs are run, and what kind of input files (blue) are used and which output files (red) are created when doing an analysis with RIFLEX and VIVANA [18].

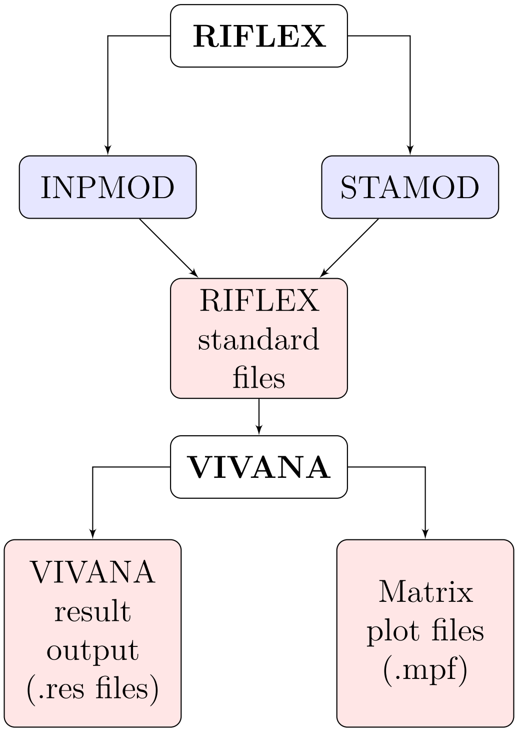


Figure 5.1: RIFLEX–VIVANA flow chart.

5.4.1 Analysis Procedure

Step 1
The first step is to find the static shape of the riser. The modelling of the riser will be done in SIMA where it will be created two input files: inpmod.inp and stamod.inp. The two input files will be used by RIFLEX for a static analysis of the riser.

Step 2
In this step an eigenvalue analysis is carried out. The results are a set of eigenvectors and eigenfrequencies f_{0i}. It is important to find enough frequencies in order to find all possible response frequencies. The modeshapes will not be used for response calculation.

Step 3
There is now a subset of calculated eigenfrequencies that might define the complete set of possible active eigenfrequencies. The added mass under VIV will be different then under the still water condition. This implies that the response frequency will be different if there is a

current, than in still water. One must therefore apply iteration for each response frequency candidate in order to find the actual response frequency. This is because the added mass depends on the frequency and a correct added mass distribution cannot be found directly. The iteration has converged when there is consistency between the modified eigenfrequency and modified added mass distribution. In this work, the added mass coefficient has been set to a constant value of 0.8 after conversations with the supervisor. The dominating frequency will be identified according to an energy criteria linked to the integral:

$$E_i = \int_{L_{e,i}} U_N(z)^2 D_H(z)^3 \left(\frac{A}{D}\right)_{C_{e,CF=0}} \mathrm{d}z \tag{5.1}$$

The frequency with the highest numerical value is chosen. $(\frac{A}{D})_{C_{e,CF=0}}$ is the non-dimensional amplitude that gives zero dynamic lift force. All eigenfrequencies that have a zone along the structure with $\hat{f}_i$ (non-dimensional frequency) between 0.125 - 0.3 [Hz] are taken as response frequencies. Figure 5.2 shows five different non-dimensional frequency candidates. However, only four of them are within the zone. This is seen by the red color. The non-dimensional frequency changes along the depth because the current velocity changes (see equation 5.2).

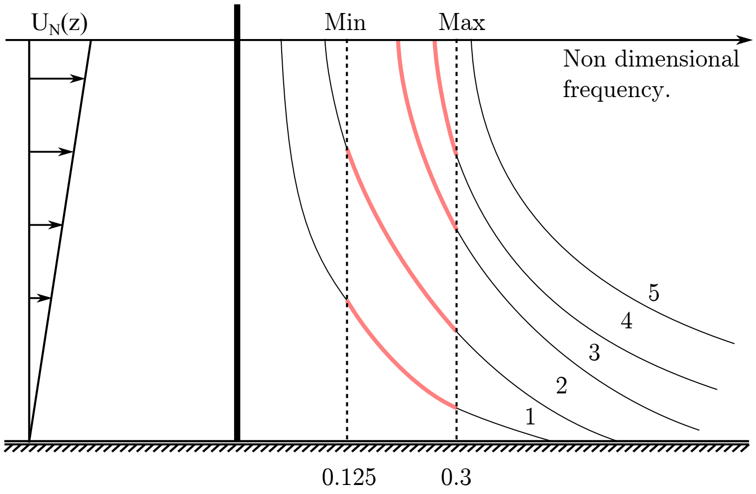


Figure 5.2: Response frequencies.

The non-dimensional frequency is given as

$$\hat{f} = \frac{f_{osc} D}{U} \tag{5.2}$$

where f_{osc} is the oscillation frequency. D and U are two parameters that may vary along the length of the riser. This is because all coefficients depend on the non-dimensional frequency[19]. The reason why $\hat{f} = 0.125 - 0.3$ [-], is beacuse of Gopalkrishnan's tests from 1993 with forced CF motions of a rigid cylinder [8]. He found that a positive excitation coefficient was in this interval for the non-dimensional frequency.

Step 4

To analyse the response at the dominating frequencies calculated in step 3, the frequency response method is used. An iteration is applied and it converges when the response is in accordance with the non-linear models for excitation and damping. The reason for using iteration is because the lift and damping coefficient depend on the local response amplitude, (A/D). A part of the iteration is to have correct phase between the lift force and response at all locations on the riser. The frequency response method is well suited for this kind of calculation since the loads are assumed to act at known, discrete frequencies. By using finite elements and the frequency response method, the dynamic equilibrium may be written as

$$\mathbf{M}\ddot{\mathbf{r}} + \mathbf{C}\dot{\mathbf{r}} + \mathbf{K}\mathbf{r} = \mathbf{R} \tag{5.3}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{C}$ is the damping matrix and $\mathbf{K}$ is the stiffness matrix. $\mathbf{r}$ is the displacement vector. By differentiating the latter once and twice, one will get the velocity and acceleration vector. The external load is harmonic, but loads at all degrees of freedom are not necessarily in phase. It is therefore convenient describing this type of load pattern by a complex load vector $\mathbf{X}$ that has a harmonic time variation at a known frequency ω. The load vector is defined as

$$\mathbf{R}(t) = \mathbf{X}e^{i\omega t} \tag{5.4}$$

The phase of the load at a specific degree of freedom is defined by the relative magnitude of the real and and imaginary components of the associated element in the complex load vector, equation 5.4.

The response vector is also a complex vector with a harmonic time variation. It is defined as

$$\mathbf{r}(t) = \mathbf{x}e^{i\omega t} \tag{5.5}$$

By inserting the response vector in equation 5.3 and then using equation 5.4, we will get

$$-\omega^2(\mathbf{M}_S + \mathbf{M}_H)\mathbf{x} + i\omega(\mathbf{C}_S + \mathbf{C}_H)\mathbf{x} + \mathbf{K}\mathbf{x} = \mathbf{X} \tag{5.6}$$

where

$\mathbf{M}_S$ Structural mass matrix.

$\mathbf{M}_H$ Hydrodynamic mass matrix.

$\mathbf{C}_S$ Sstructural damping matrix.

$\mathbf{C}_H$ Hydrodynamic damping matrix.

$\mathbf{X}$ Excitation force vector. Non-zero terms are present within an excitation zone. The excitation force must always be in phase with the local response velocity.

Step 5

When the responses are known it is possible to calculate the fatigue damage. A user defined S-N curve and the Palmgren-Miner rule is applied. For the space sharing frequencies a rainflow counting algorithm is used. For the time sharing option, an analytical solution is used.

5.5 BATCH-scripting

Learning how to create a BATCH-script has been important in this work to do analyses with many different parameters. The BATCH-scripts are used to run RIFLEX and VIVANA. Before they are run, the script will get the correct variables from a text-file containing the relevant information, and these parameters are then inserted into the input files. The parameters are inserted into variables called wild-cards, and they can be found in the initial input files. Every wild-card starts with a dollar sign and has a name in capital letters. An example is $CURDIR which is the wild-card for the ocean current direction. A part of the INPMOD input file is illustrated below to show how the wild-cards are used.

```
 NEW CURRENT STATE
'-----------------------------------------------------------------------
'Always remeber that first curlev has to be -0.0 and not 0.0!
'icusta nculev
 1       7
'curlev    curdir    curvel
-0.0    $CURDIR 0.242
-10.0   $CURDIR 0.210
-50.0   $CURDIR 0.182
-200.0  $CURDIR 0.171
-500.0  $CURDIR 0.137
-1000.0 $CURDIR 0.116
-1500.0 $CURDIR 0.000
```

The variables that are inserted to the wild-cards are found in text-files that the BATCH-script loops through. These files are called variations.txt files. More explanation about them can be found in appendix A.2.

When the correct variables are assigned using the *change_str_in_file.m* function, the script will active RIFLEX and then VIVANA. When these analyses are finished, the BATCH-script will activate Matlab and use functions and scripts which are created for purposes such as to plot the data in the matrix plot files created by VIVANA and RIFLEX. Figure 5.3 displays a flow chart that illustrates how the BATCH-process is executed and how it communicates with the functions created in Matlab. The name of all the BATCH-scripts used is *run_with_matlab.bat*. The BATCH-scripts are placed in their respective analysis folder, where they are executed. Thus, even if the names of several BATCH-scripts used in this work are the same, the BATCH-scripts may have different functions in the different folders. Both BATCH-scripts used in this work are attached in the appendix. This is done in order to give the reader an better explanation on how the scripts are constructed.

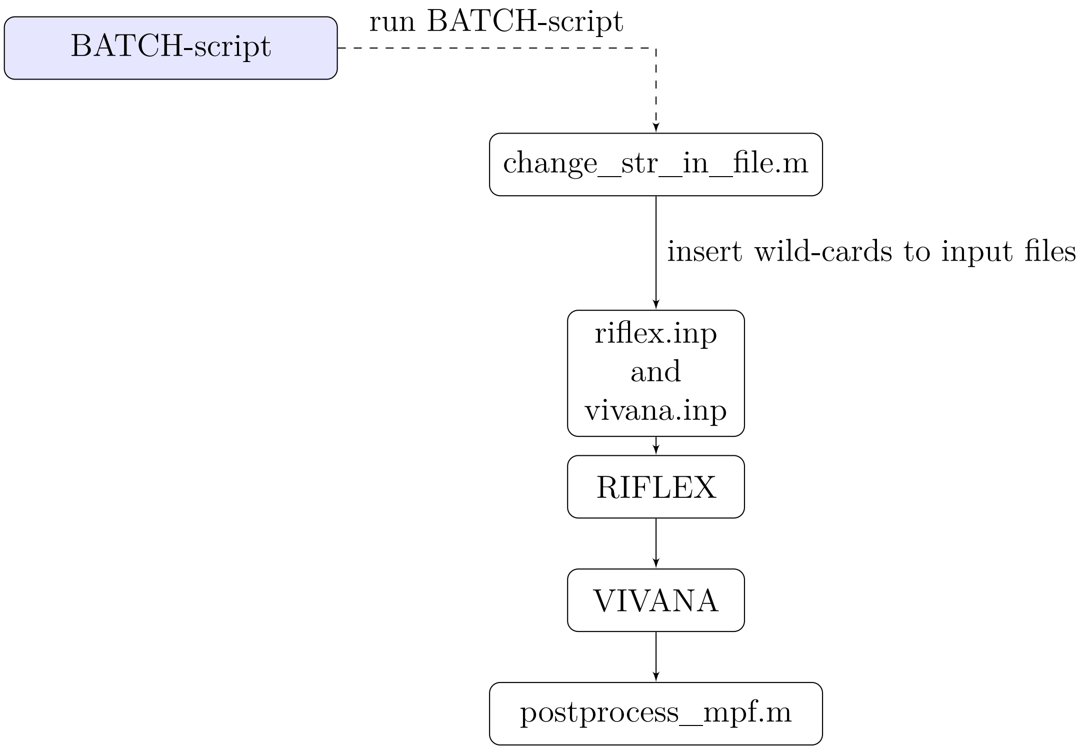


Figure 5.3: Flow chart of a BATCH-process.

This flow chart explains also how the parameter variation analysis was performed.

5.6 MATLAB

Matlab has been widely used in this master thesis. The main application of the program in this work has been to

- Create functions that communicate together with the BATCH-script in order to run RIFLEX and VIVANA with different parameters.
- Create plots from the matrix plot file (.mpf) file that VIVANA produces.
- Process the vessel motion coordinates.
- Create necessary plots of the vessel statistics.
- Process results from the parameter variation analysis and variation of vessel coordinate analysis.

All functions and scripts have been attached in the appendix. They have been uploaded to a zip-file to DAIM as well. Every function and script has a small explanation at the beginning to help the user understand how they are used.

A list of Matlab scripts with a small description is given on the following pages.

change_str_in_file.m This function will read a input file and replace the wild-cards with given parameters.

postprocess_mpf The scripts imports the results from the .mpf-file created by RIFLEX,VIVANA and etc. It creates a struct with the data from the .mpf-file and plots the results in a folder.

read_xy_data.m This Matlab script is created to read through the Xpos.asc and Ypos.asc (from the SIMO analysis) files that contain the data of the vessel motion and save them in a struct for further processing.

find_extreme_coord.m Find the largest extreme coordinates from the data.

transform_xy_data.m The function takes in the original (x, y) data, and transforms them according to the new transformed coordinate system.

count_in_large_matrix.m This function goes through all the transformed, multiplied coordinates, and inserts the number of occurrences into a large matrix.

xy_count_and_prob.m The function goes through all the coordinates in the counted_x_y_data- matrix and finds the number of occurrences in the large_transformed_coord_matrix. By knowing the total number of occurrences it is possible to find the probability of the motion $p(x, y)$.

convert_xy_to_polar_xy.m This function extracts the coordinates, their counts and probability, and converts the coordinates to polar coordinates, for further processing.

search_xy_in_sectors.m This function will go through the sectors and the discrete points defined in the input, and count the occurrences of the (x, y) position of the vessel.

extract_fatigue_from_mpf.m This function extracts the fatigue info from the input .mpf-file and creates a new file with only the fatigue data. This is created in order to minimize the space when running the riser analysis with many (x, y) coordinates.

save_run_xy_info_in_struct.m This function goes through all the folders containing the info of the (x, y) coordinate variation runs and saves it in a struct for further processing.

add_fatigue_damage_to_struct.m The function goes to the respective folder, and extracts the fatigue damage saved in the fatigue_damage.txt file. It then saves this info in the struct for further processing.

sum_fatigue_damage_from_xy_variation.m The function goes through the run_data struct and sums the fatigue damage.

find_important_parameters_resfile.m This function finds important parameters from the stamod.res file, which contains results from the static analysis of the riser. This function was created because the large .mpf-files are deleted in the variation of vessel coordinate analysis due to large data size.

5.6.1 Flow chart for processing the vessel motion data

In this work, it was necessary to process the vessel motion data given by DNV in order to create a discrete probability distribution. More info about the motion data is given in Section 8.3. A flow chart is created under to give the reader information of how the data was processed. The green boxes are I/O files which are in a simple text format. The red boxes are the main Matlab-script which runs the functions beneath them. The blue box is the BATCH-script.

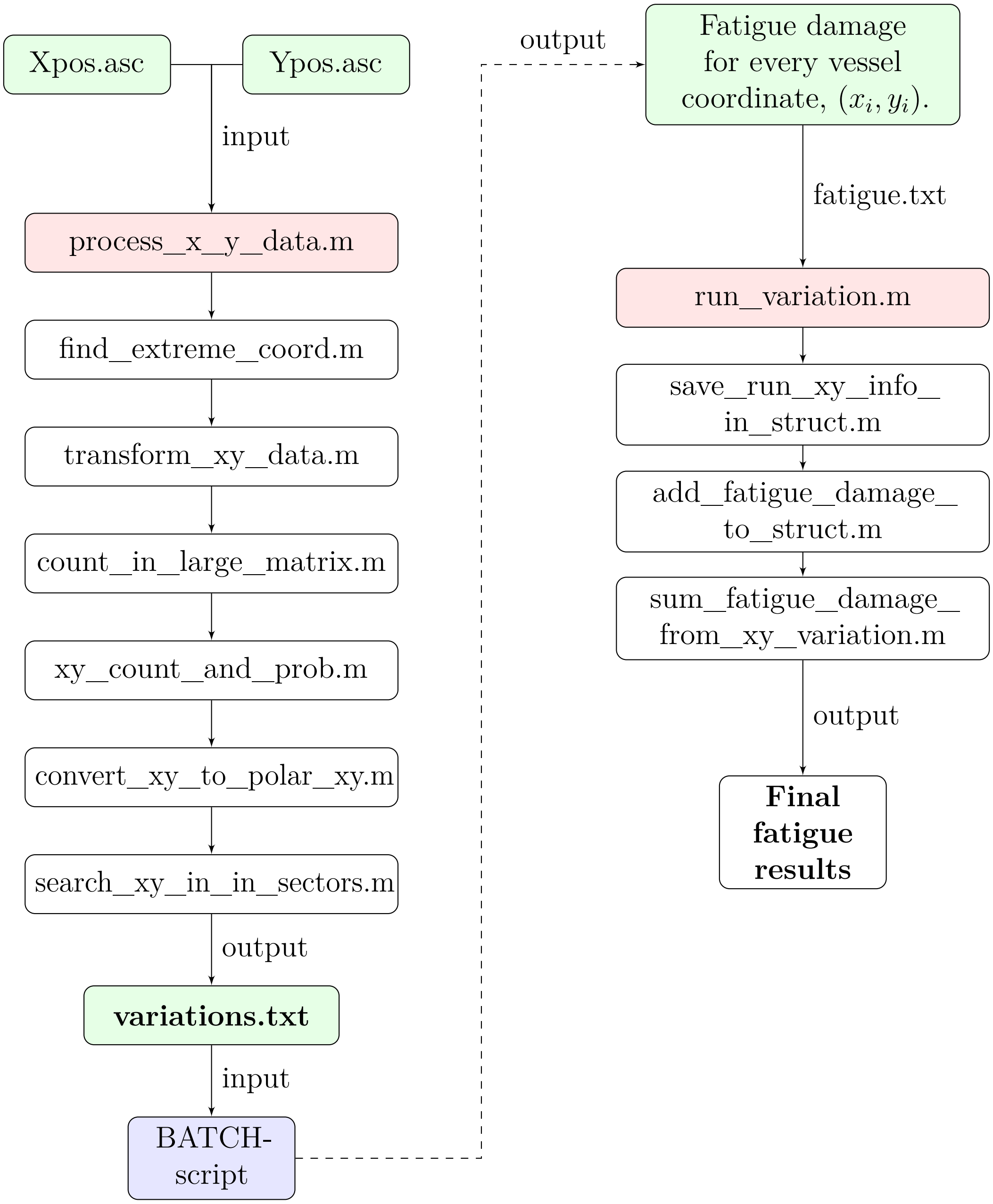


Figure 5.4: Flow chart of Matlab functions used in the variation of vessel coordinate analysis.

Chapter 6

Riser configuration

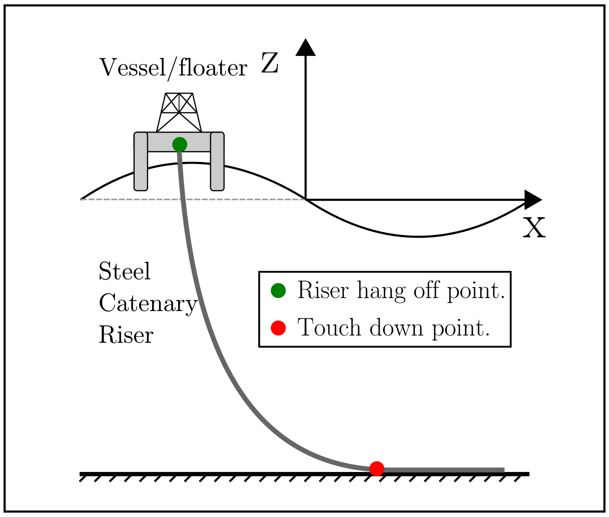
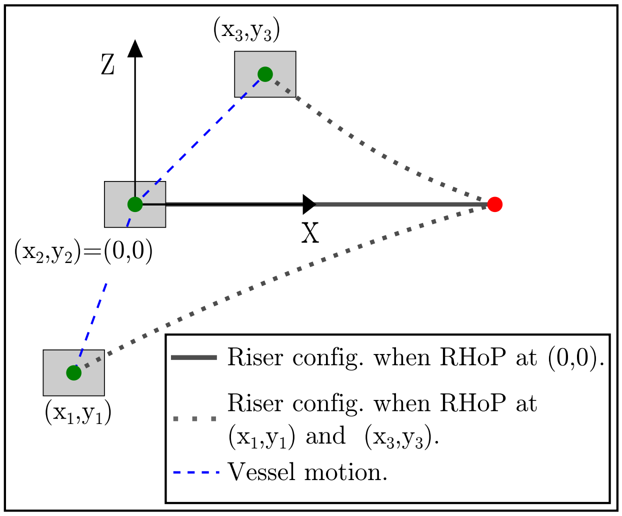


Figure 6.1: Riser configuration example, ZX-plane.



(x3,y3)
Z
X
(x2,y2)=(0,0)
(x1,y1)
Riser config. when RHoP at (0,0).
Riser config. when RHoP at (x1,y1) and (x3,y3).
Vessel motion.

Figure 6.2: Riser configuration example, XY-plane.

External diameter	[mm]	230.0
Wall thickness	[mm]	25.4
Internal diameter	[mm]	179.2
Density	[kg/m^3]	7850.0
E-modulus	[Pa]	2.00E+11
Second moment of area, I	[m^4]	8.67E-05
Mass per length	[kg/m]	128.2
Axial stiffness	[kN]	3.27E+06
Bending stiffness	[kNm^2]	1.73E+04
Torsional stiffness	[Nm^2/rad]	1.00E+05

Table 6.1: Data of the SCR.

Figure 6.1 illustrates the configuration of the SCR, and how it was originally modelled in SIMA. The original riser hang off point (RHoP) was set to the origin, and the termination point was set to a value of x = 896.0 [m]. The static shape was originally found by using the catenary equations given in [20]. Figure 6.1 illustrates the riser seen from the side, where the x-axis points to the right and the z-axis upwards. The figure to the right illustrates the same floater and riser seen from above. In that figure there are three different positions displaying an example of the vessel motion. When the riser was modelled in SIMA the RHoP was set to the origin as one can see from the (x_2, y_2) position above. The data used for modelling the riser have been found after conversations with the supervisor at the department of Marine Technology, and the co-supervisor from DNV.

Chapter 7

Parameter variation analysis

7.1 Introduction

A significant effort has been made over the last decades to develop new VIV models or improve existing ones. However, it is still obvious that deterministic estimates of full scale riser VIV fatigue damage will include great uncertainties. These uncertainties are mainly due to the lack of knowledge of the ocean currents generating the VIV and the complex nature of VIV itself. Much of the uncertainty is linked to the amount and relevance of data available to understand, characterize VIV and capture the broad complexity of the phenomenon. In this master thesis one of the tasks is to find out how different parameters used in the for the fatigue analysis will effect the results.

One of the challenges when doing a parameter analysis is finding which parameters are important. In the thesis, the variables that have been tested are assumed to be parameters which are difficult to predict precisely and which will have an notable impact on the fatigue results. The paramters tested in the thesis are displayed in the following.

- Ocean current (only unidirectional).
- Current heading.
- Bottom properties.
- Relative damping.
- Number of elements of second segment.
- S-N curve.

All parameters and their values are displayed in appendix A.2.1.

7.2 Current profile and direction

The current profile in VIVANA can only be modelled unidirectionally. From studies done by [5], this is seen to be unrealistic. The current might vary with the depth, both in magnitude and direction. The latter is not possible to do with VIVANA at the moment. This is of course a problem, because the vortex shedding is influenced by the current. Another problem with current modelling, is that it changes direction with time. It is therefore crucial to create good statistical models of the current which can be used in the analysis.

The way this analysis was performed, was to run one analysis with one current at the time. One of the questions when this master thesis was in its starting phase, was to check if it was possible to use many current profiles, each with its own probability. One would have to do one analysis for each current profile, and in the end, summarize the total damage as

$$D_{tot} = \sum_{i=1}^{N} D_{cur,i} \cdot p_{cur,i} \tag{7.1}$$

where

$D_{cur,i}$	[1/yeas]	Fatigue damage for current i.
$p_{cur,i}$	[-]	Probability for current i.

However, only one current profile was used in each analysis, and summation of the damage from different current profiles was not done. It has simply been tested how current profiles with different magnitudes and directions effect the fatigue results. The current profile was modelled with a velocity distribution with seven points, where each point had a specified magnitude. The values are displayed in Table 7.1.

	Current 1	**Current 2**	**Current 3**	**Current 4**
Depth [m]	[m/s]			
0.0	0.242	0.662	1.082	1.478
-10.0	0.210	0.545	0.887	1.190
-50.0	0.182	0.449	0.726	0.957
-200.0	0.171	0.415	0.668	0.866
-500.0	0.137	0.306	0.489	0.618
-1000.0	0.116	0.247	0.393	0.488
-1500.0	0.000	0.000	0.000	0.000

Table 7.1: Current profiles.

The current direction is also of importance when doing the parameter analysis. It is only possible to consider four directions with a SCR. This because the CF and IL excitation are not known for other directions for a SCR since the there will be a coupling effect between the coefficients.

7.3 Bottom properties

In VIVANA, the bottom is modelled with springs as bottom stiffness and with dashpots as bottom damping. The starting data (STFBOT = 300.0 [kN], STFAXI = 200.0 [kN] and STFLAT = 200.00 [kN]) used in the analysis was provided by the co-supervisor from DNV. In this thesis only bottom stiffness has been used in the analysis. The bottom damping has been set to zero. One of the problems with the bottom properties in VIVANA is that in frequency analysis, the springs are not allowed to be released if the riser is lifted from the sea floor. This creates tensile forces which are not realistic. The reason for changing the bottom properties has been to see how and where the fatigue damage is affected.

7.4 Relative damping

The relative structural damping is specified by the user when doing the VIV analysis with VIVANA. The relative damping is the internal damping of the structure due to friction and local strains. The damping is modelled for each response frequency as a damping matrix proportional to the stiffness matrix [21]

$$\mathbf{C}_i = \alpha_i \mathbf{K} \tag{7.2}$$

where

$\mathbf{C}_i$	Damping matrix for frequency i.
$\mathbf{K}$	Stiffness matrix.
α_i	Proportionality factor for frequency i.

The damping is known as Rayleigh or proportional damping. It might also include a contribution from the mass matrix as well. From [21] it is found that the inertia part is not needed since the damping level should be valid for one specific frequency. From standard theory for structural dynamics, the proportionality factor is given as

$$\alpha_i = \frac{2\xi}{\omega_i} \tag{7.3}$$

where

ξ	User defined structural damping ratio.
ω_i	Response frequency i.

The model used will give the same relative damping contribution for all response frequencies.

7.5 Number of elements

This parameter was chosen to find out how the FE model in VIVANA converges as the number of elements alters. It is only the second segment of the modelled riser that has a varying number of elements. This is also the segment that covers the area where the riser reaches the seabed.

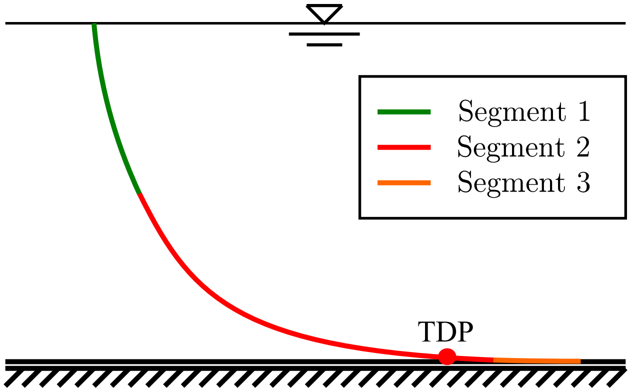


Figure 7.1: Riser segments

7.6 S-N curve selection

The S-N curves relates the number of stress cycles to failure to the corresponding stress range. There exists many S-N curves after numerous tests on structural specimen in laboratories. Still, there is much uncertainty linked to which curve is best fitted for the structure in ones analyses. How can one know which curve gives the best result? And how does different curves effect the final result? The main focus was to figure out how much does the S-N curve changed the result.

7.7 Info about the results

Every parameter tested consist of 3-4 runs. For each parameter varied, four figures are displayed illustrating the results. They figures are.

Top left figure: Displaying the fatigue damage of the entire riser for the respective runs.

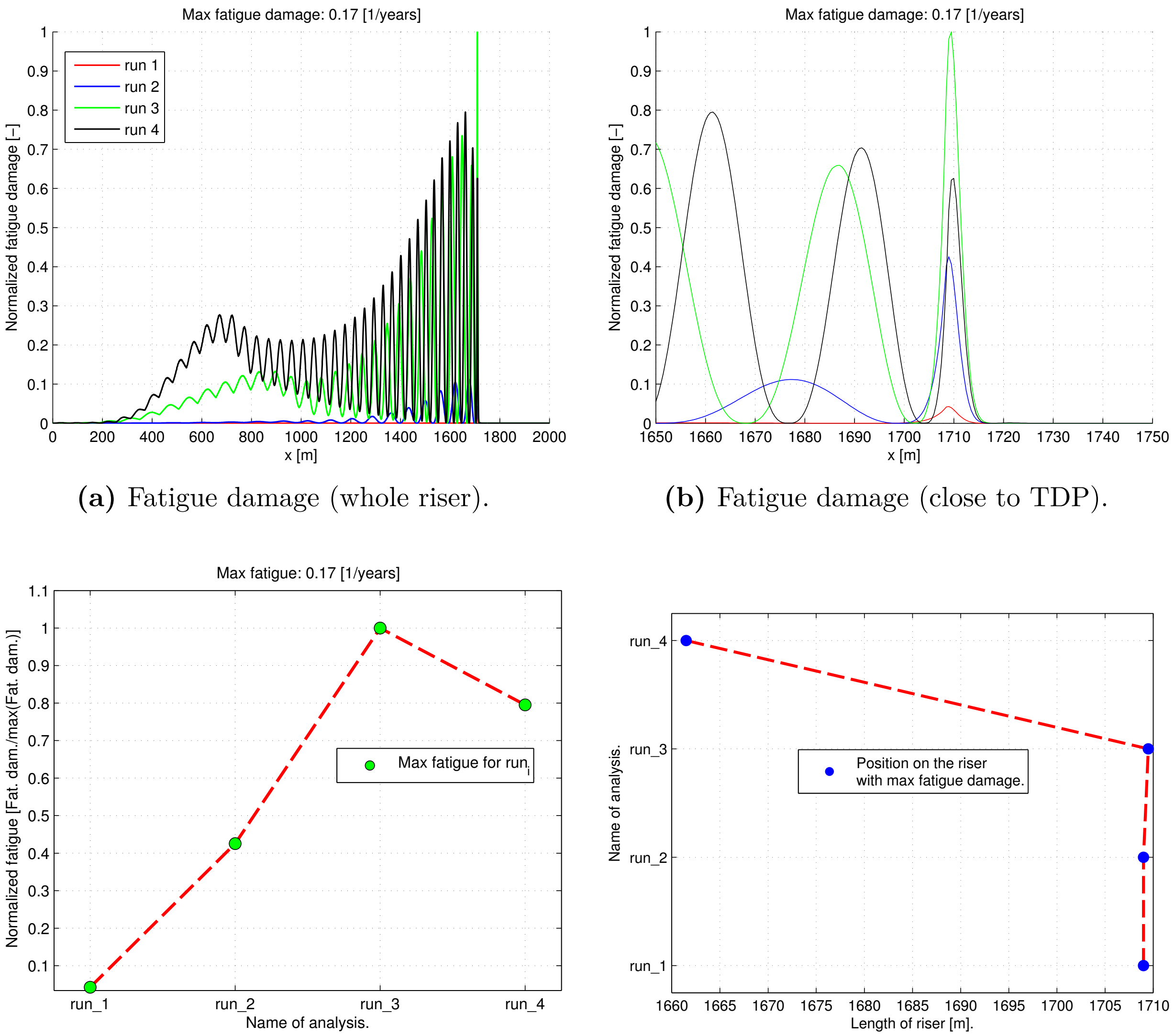
Top right figure: Displaying the fatigue damage near the TDP area for the respective runs.

Lower left figure: Illustrating the maximum fatigue for every respective run. The values are normalized, such that the largest value equals to unity. Used to easily compare the maximum damage and look for trends in the results.

Lower right figure: Illustrating how the point on the structure with maximum fatigue moves.

7.8 Results

7.8.1 Run 1-4



(a) Fatigue damage (whole riser).

(b) Fatigue damage (close to TDP).

(c) Max fatigue damage.

(d) Riser point with max fatigue.

Figure 7.2: Results from parameter run 1-4.

Run 1-4 is tested with variations in the current magnitude. The values of the current are found at page 44. The currents have the same form, but are only scaled differently. From the results in the figures above one can observe that the fatigue damage increases for increasing current velocity. This is an expected result since the number of modes will increase with current velocity, which will give larger curvature and fatigue damage. On the other hand, it is noticed that the maximum fatigue is seen for the third current profile (Run 3). Figure 7.2c illustrates this very clearly. By looking at the figure 7.2b and 7.2d it is seen that the maximum fatigue point for the greatest current has a substantial change in position. It seems that this is caused by the modes excited by the current. This indicates that there may be a connection between which modes are excited and where the maximum fatigue damage will appear. This is important to notice, because it means that the largest current will not necessarily give the largest fatigue damage.

There is also an interesting peak around an length of 700 [m]. This can be explained by looking at the results from the matrix plot file produced by VIVANA. By looking at which excitation zones were given to the vortex shedding frequencies, it is clearly seen that the excitation zone is large in this area, and that the riser will experience largest response amplitudes here.

The cross flow amplitudes are seen for two frequencies excited, and the excitation zones for a large part of frequencies are shown. It is illustrated that the greatest response is induced by the frequency that oscillates with 0.769 [Hz].

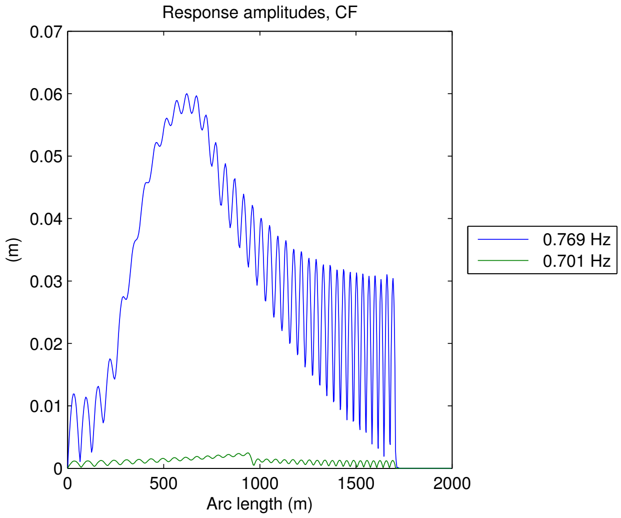


Figure 7.3: Response amplitudes of Run 4.

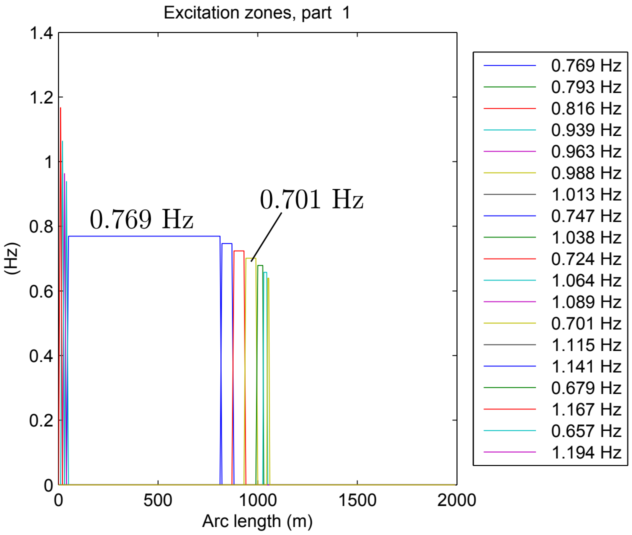
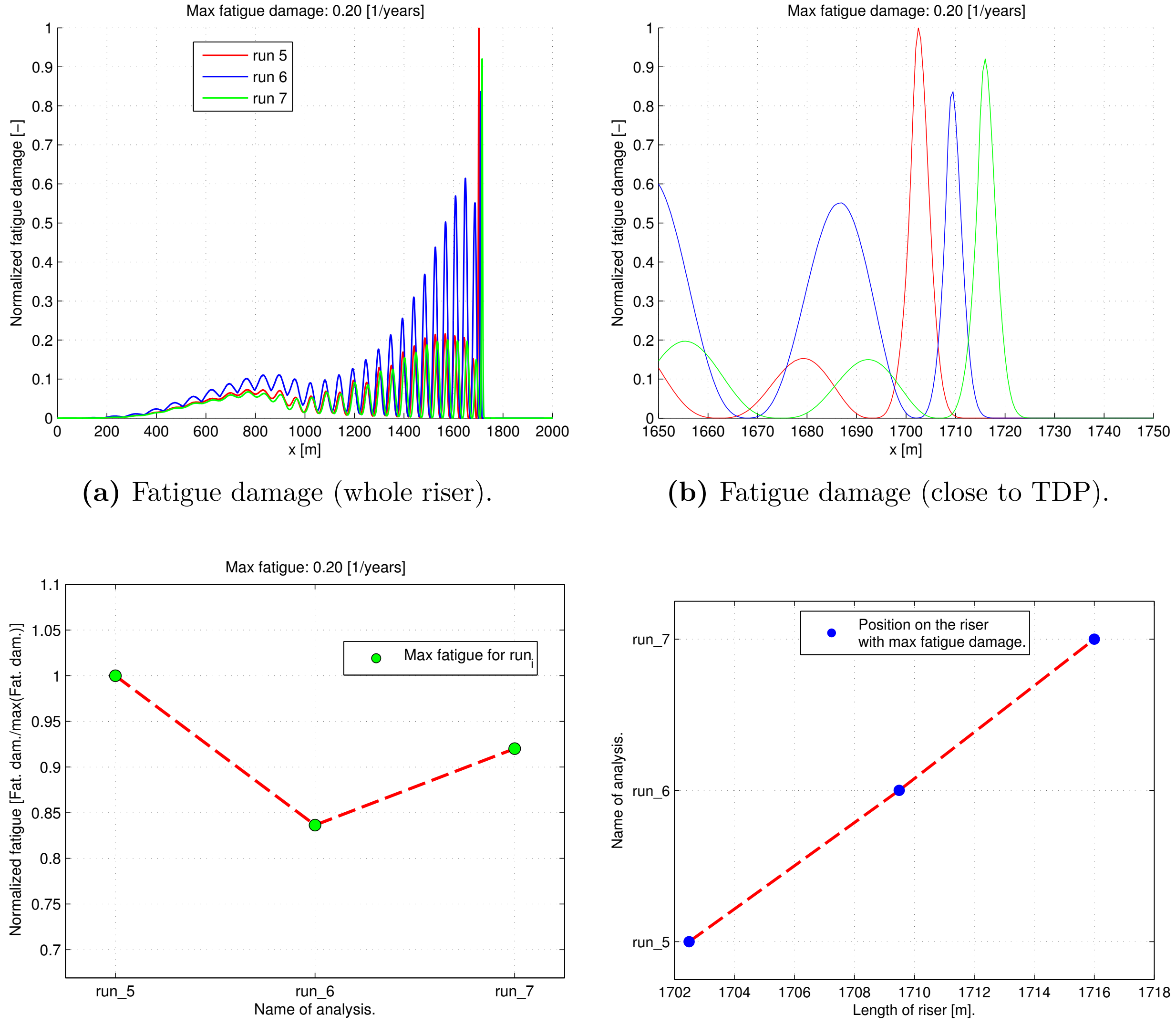


Figure 7.4: Excitation zones of Run 4.

7.8.2 Run 5-7



(a) Fatigue damage (whole riser).

(b) Fatigue damage (close to TDP).

(c) Max fatigue damage (Run 5-7).

(d) Riser point with max fatigue.

Figure 7.5: Results from parameter run 5-7.

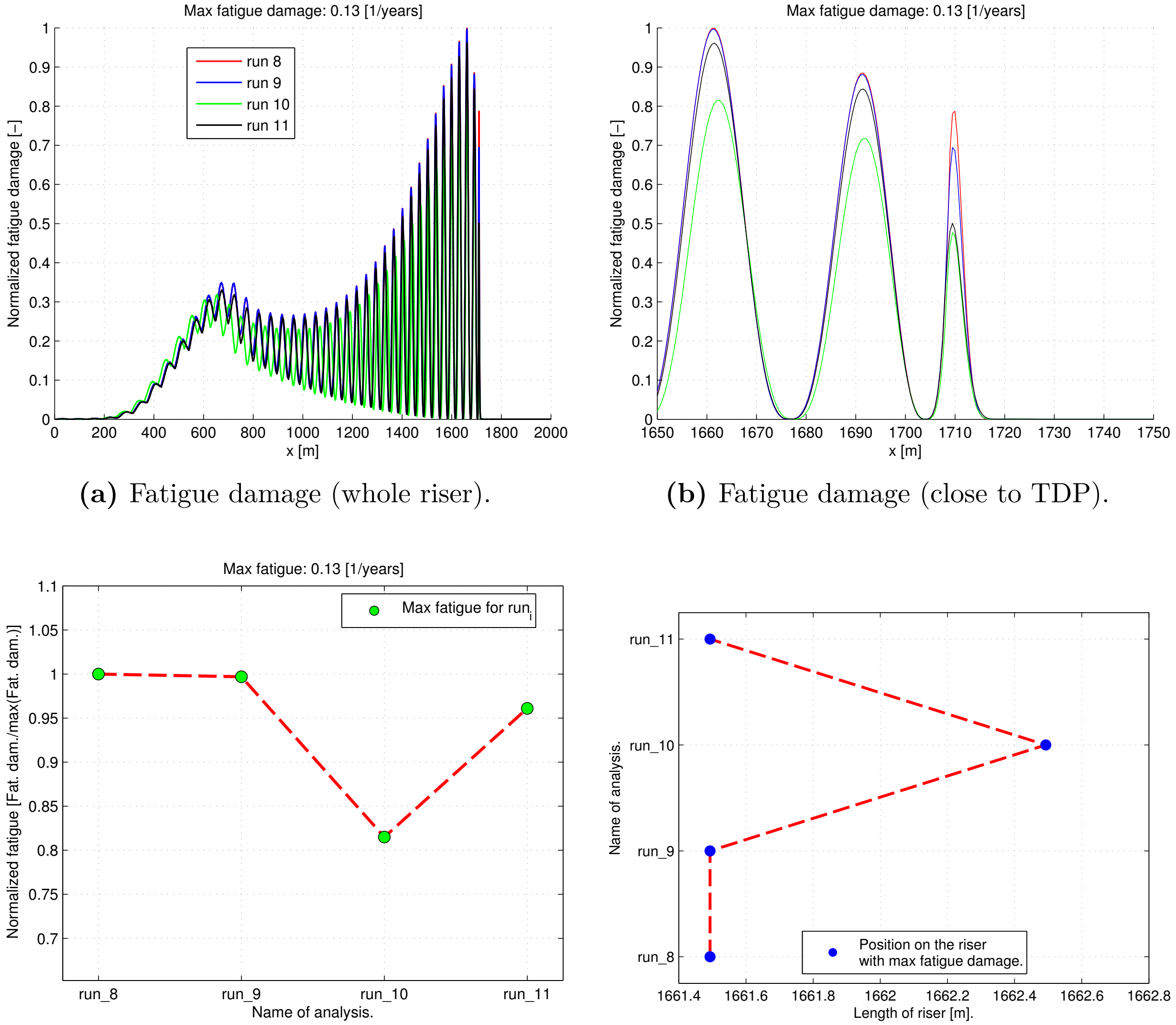
In Run 5-7 the current direction is tested. The current magnitude is the same.

Run:	Direction [deg]:
5	0.0
6	90.0
7	270.0

Table 7.2: Current directions used in parameter study.

It is clearly illustrated that the fatigue damage differs depending on which direction the current comes from. This is why it is important to improve existing VIV software to be able to handle all current directions. The difference in fatigue damage is relatively small in the beginning. This is because the upper part of the riser is quite straight, and the fatigue damage is similar as for a straight, top tensioned riser. However, the fatigue increases as the curvature of the riser increases. We can see from the figures above that the current coming in the orthogonal riser plane gives largest fatigue almost over the entire riser, except near the TDP. There, the largest fatigue is seen for Run 5. Run 6 and Run 7 have a maximum fatigue that is around 84 % and 93 % of Run 5. The point with maximum damage is also different for every run in this section. It is also interesting to see that the increase in fatigue damage is quite large for Run 5 and 7 when reaching the touchdown area.

7.8.3 Run 8-11



(a) Fatigue damage (whole riser).

(b) Fatigue damage (close to TDP).

(c) Max fatigue damage.

(d) Riser point with max fatigue.

Figure 7.6: Results from parameter run 8-11.

In Run 8-11 the bottom properties of the sea floor are examined.

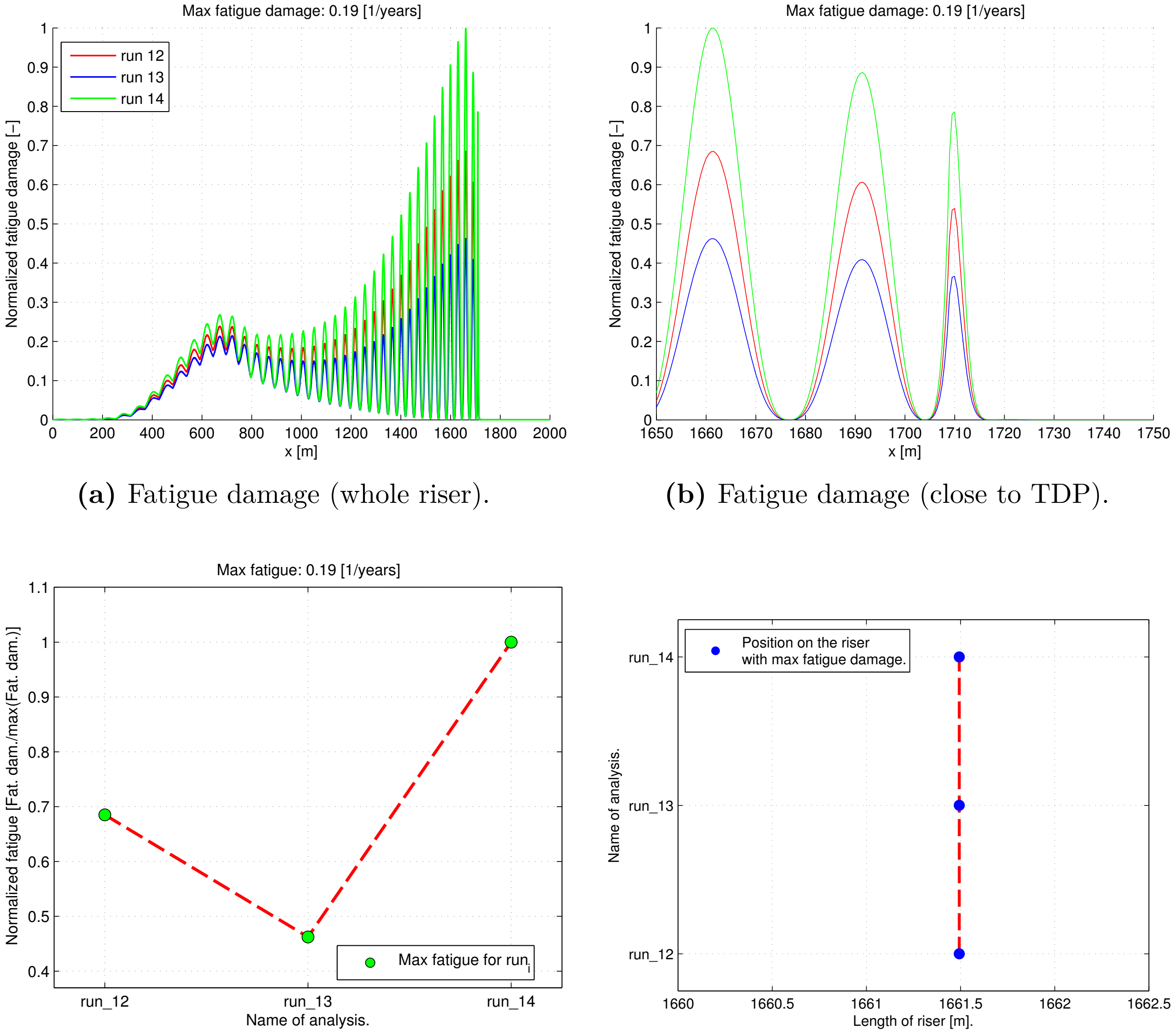
Run	Bottom stiff.	Axial stiff.	Lateral stiff.
8	300.0	200.0	200.0
9	250.0	150.0	150.0
10	200.0	100.0	100.0
11	150.0	50.0	50.0

Table 7.3: Bottom stiffness values used in the parameter analysis.

All stiffness values are in [kN/m].

The bottom stiffnesses have been minimized with 50 [kN/m] for each run. The results show that run 8 gave the largest fatigue damage. This was expected as well. There are some differences in the fatigue damage before the TDP, however, they are negligible. From Figure 7.6a one can notice that Run 10 has a fatigue damage that is not in phase with the other runs between an arclength of around 600 [m] until somewhere around 1100 [m]. After that the damages are in phase with each other. The greatest difference is seen near the touchdown point. The run with greatest bottom stiffness gave the worst maximum damage. From the results, it is seen that the point on the riser with greatest damage does not move considerably with different bottom properties.

7.8.4 Run 12-14



(a) Fatigue damage (whole riser).

(b) Fatigue damage (close to TDP).

(c) Max fatigue damage.

(d) Riser point with max fatigue.

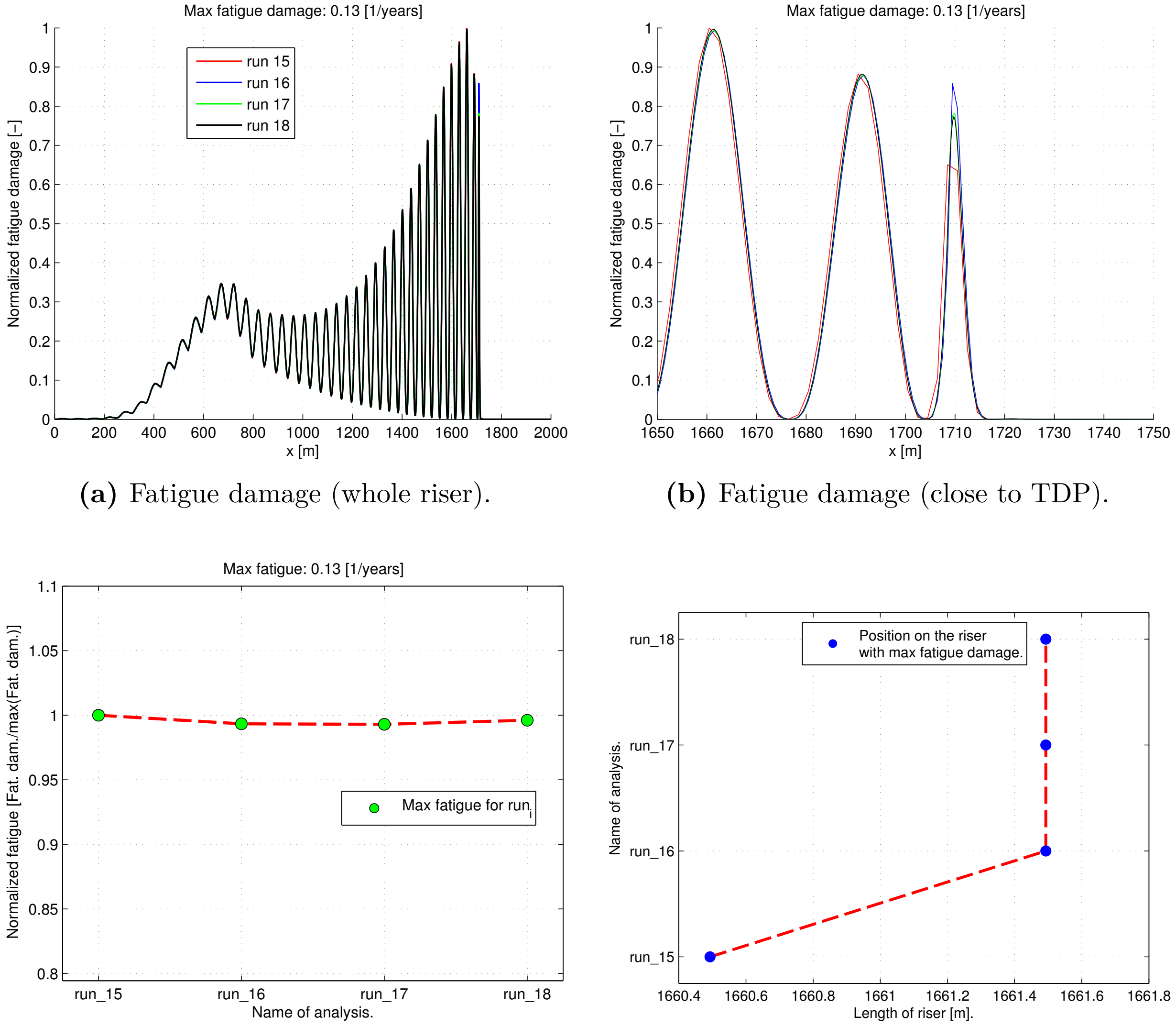
Figure 7.7: Results from parameter run 12-14.

Run	Relative damping	Variation
12	0.01	Original value
13	0.012	+20 % of original value
14	0.008	−20 % of original value

Table 7.4: Values of relative damping used in the parameter study.

The results from Run 12-14 were as expected. The riser with the largest relative damping has the smallest fatigue damage, and vice versa. However, the results reveal that if the structural damping is increased with 20 %, then the maximum damage is decreased with a value of around 30 %. If the structural damage is decreased with 20 %, then the maximum fatigue damage is increased with around 45 %. There is no change of where the maximum fatigue is on the riser. This is satisfactory. The fatigue damages are also in phase, which seems reasonable.

7.8.5 Run 15-18



(a) Fatigue damage (whole riser).

(b) Fatigue damage (close to TDP).

(c) Max fatigue damage.

(d) Riser point with max fatigue.

Figure 7.8: Results from parameter run 15-18.

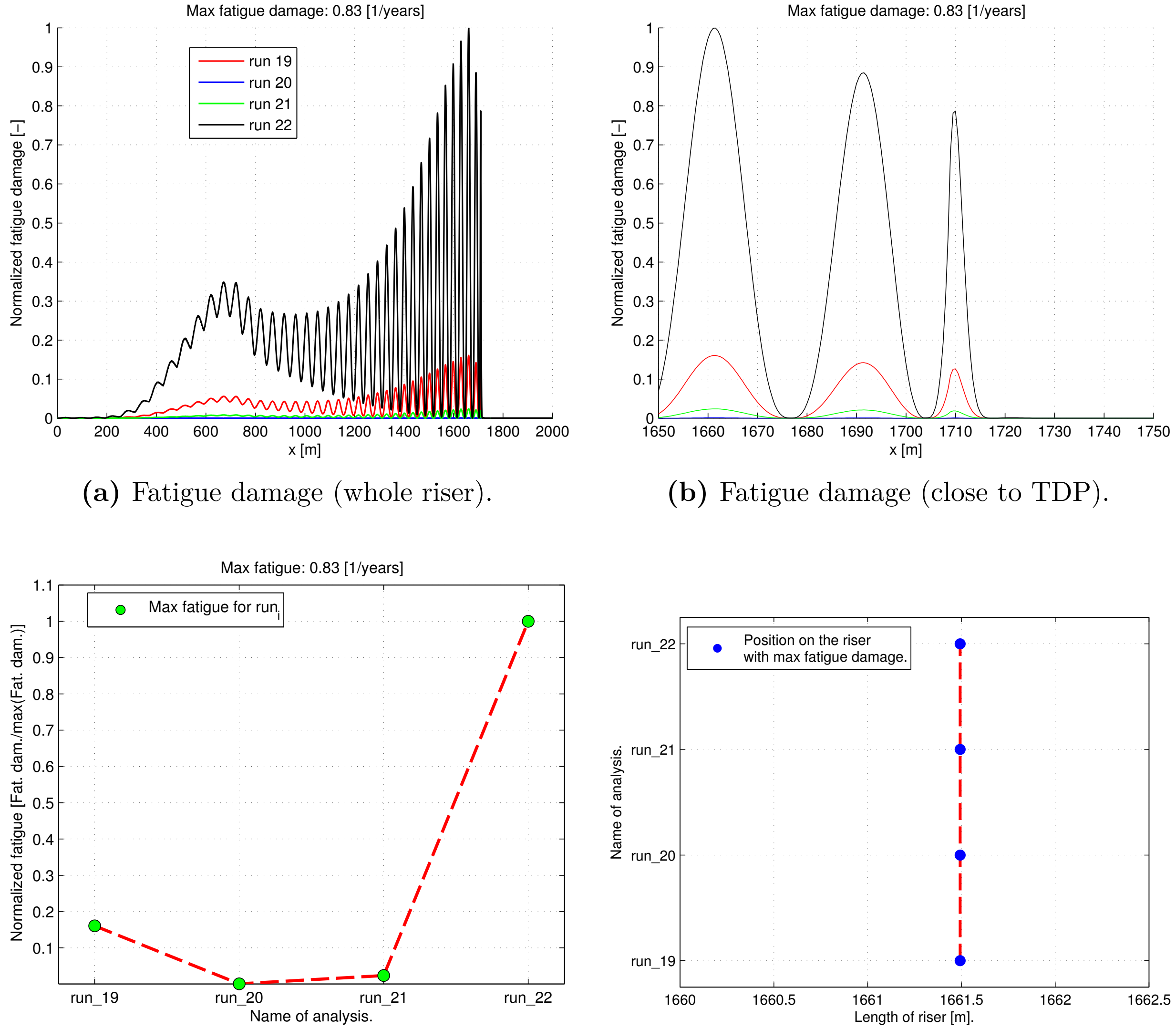
Here are the parameters varied in Run 15-18.

Run	No. of elements [-]	Element length [m]
15	400	2.0
16	800	1.0
17	1600	0.5
18	3200	0.25

Table 7.5: Number of elements used in the parameter study.

The riser is modelled with three segments as illustrated in Figure 7.1. The second segment is covering the part of the riser which has the greatest curvature and is at the touchdown area. The total length of the segment is 800 [m]. This is the reason for looking into how fine the elements need to be in this area in order to get reliable results. By looking at Figure 7.8a it is not easy to see any differences in the results. However, more details are revealed as the fatigue results are magnified. The maximum fatigue damage is relatively equal for all cases, with only couple of percent difference between each run. The greatest effects of the element discretisation is seen at the fatigue top at 1710 [m] in Figure 7.8b. The curve is not as smooth in the area because of the small number of elements. The two last runs are smooth and the results satisfactory. In this test case, VIVANA gave quite similar results even though the element length was altered between 2.0–0.25 [m].

7.8.6 Run 19-22



(a) Fatigue damage (whole riser).

(b) Fatigue damage (close to TDP).

(c) Max fatigue damage.

(d) Riser point with max fatigue.

Figure 7.9: Results from parameter run 19-22.

The last variable changed in the parameter variation study was different S-N curves, in order to get an idea of how they effect the final fatigue results. The S-N curves are taken from [14]. As mentioned more in detail in Chapter 4, the basic design of a S-N curve is given by the equation

$$log(N) = log(\bar{a}) - m \cdot log\left[\Delta\sigma\left(\frac{t}{t_{ref}}\right)^k\right] \tag{7.4}$$

where, the $log(\bar{a})$ is the intercept of log N-axis by the S-N curve, and m is the negative inverse slope of the S-N curve.

In Run 19-22 the parameters $log(\bar{a})$ and m that were changed as found in the recommended practice (RP-C203). The values used in this study were:

Run	m	$log(\bar{a})$
19	3	11.764
20	4	15.117
21	3	12.592
22	3	10.970

Table 7.6: S-N curves used in the parameter study.

The results were as expected. When there is a requirement of many cycles to fatigue, as in Run 19 and Run 21, the fatigue is small. This is especially seen for Run 20 where the S-N curve is moved up to an intercept of the log N-axis to 15.117. It is easy to notice that the fatigue damage is almost eliminated in this case. On the other hand, the fatigue damage is extremely magnified for Run 22. Here a S-N curve with a low interception is chosen, which means that the number of cycles to fatigue is lowered. This is pointed out in the plots above, where we can see an increase of a great scale in the fatigue damage. The results show that S-N curves are extremely important to the outcome of the fatigue results.

Chapter 8

Variation of vessel offset analysis.

8.1 Introduction

The reason for doing this analysis is because of the strong increase in fatigue damage near the TDP of the SCR. A key issue in the design of a SCR is to control stresses and fatigue damage in the touchdown area. Dynamic bending stresses will vary along the riser, and a large peak will often be seen near the TDP. The peak is caused by the restrictions on the riser displacement from the presence of the sea floor, and the local bending stresses will be influenced by stiffness and damping properties of the bottom.

The analysis model in VIVANA represents the interaction between the riser and the sea floor with discrete springs. In linear frequency domain analysis the spring properties remain constant independent of the displacements. This type of model may give an over-prediction of the bending stresses at the touchdown point, since a linear spring will induce tensile forces instead of being released from the sea bottom when the riser is lifted. The TDP will also move with time because of the vessel or floater motion, which will lead to a more distributed fatigue damage (engineer P.E. Voie, pers. comm.).

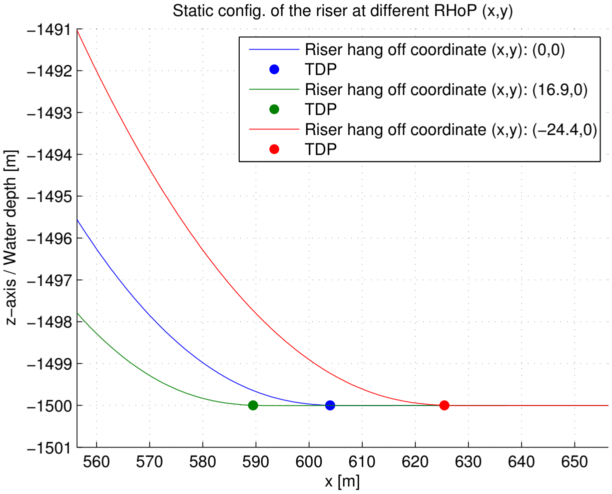


Figure 8.1: Example of changing TDP.

8.2 Analysis steps

The analysis of the varying coordinate for the RHoP is going to be explained in the following pages. Here is a list displaying the overview of the procedure. A flow chart illustrating this procedure can be displayed on page 40 as well.

1. Read the vessel motion files with Matlab.

2. Find the largest coordinate values in order to create a transformed coordinate system which only has positive values.

3. Transform the motion coordinates (x, y) into transformed coordinates (X_T, Y_T).

4. Multiply the transformed coordinates enough such that all coordinates are integers, and then can easily be placed into a matrix.

5. Create a large matrix where the index (i, j) represents the multiplied transformed coordinates. This is necessary to count the number of occurrences of a original (x, y) coordinate of the vessel motion.

6. Count the occurrences of the transformed coordinates (X_T, Y_T) into the large matrix.

7. Transform the large coordinates back to the original ones. $(X_T, Y_T) \Rightarrow (x_i, y_i)$

8. Having the number of occurrences of each (x_i, y_i) and the total number of all occurrences, the probability is possible to find by $p(x_i, y_i) = n(x_i, y_i)/N_{tot}$.

9. There is now a probability $p(x_i, y_i)$ for each coordinate (x_i, y_i). Since there are many coordinates, it is easier to sum the probability of the coordinates into defined discrete points in sectors that cover all the occurrences.

10. A statistical representation of the vessel motion is described as probability at specified discrete points.

11. An interpolation of the discrete points is carried out in Matlab. This is useful to create a surface and a contour plot of the probability. **N.B.** The probability is only correct on the surface/contour points corresponding to the discrete points.

12. A RIFLEX and VIVANA analysis is then carried out at each of these discrete points.

13. The fatigue results are extracted into a fatigue.txt file in its respective analysis folder.

14. A Matlab script will sum all the fatigue results with their respective probability in order to get the final outcome.

15. The final step is comparing the the results from the steps above with the original analysis, where the RHoP is placed in origin, and not moved.

8.3 Vessel motion

8.3.1 Raw data from SIMO analysis

The analysis started with an analysis done by Per Erlend Voie, the co-supervisor from DNV. He did a motion analysis with SIMO of a vessel/platform. The analysis was done using a scatter diagram of the environmental conditions at a specific area. The details are not important in this work. The intention was to get an idea how the fatigue results would be affected by varying the (x, y) position of the riser hang of point (RHoP). The data that were analysed in this work consisted of 556 different folders. Each folder had a specific weather condition and consisted of two files representing the vessel data. An example of a folder name was dir0_hs10_tp11.4.

Explanation of the folder name.

dir	[degrees]	Direction of the weather.
h_s	[m]	Significant wave height.
t_p	[s]	Peak period.

In each folder there were files as a result of the motion analysis in SIMO. Two of the files were Xpos.asc and Ypos.asc. Each of these files consisted of x- or y-position at a certain time. Every analysis was done for 3599.75 [s]. The files containing the motion information were imported into Matlab for further processing. For every folder that was run through, there was created a text-file that consisted of data showing time, x-pos and y-pos of the motion.

Here is an example of such a text-file and a figure showing the visualization of the motion.

time [s]	**x-pos [m]**	**y-pos [m]**
0.00	-3.3264	5.6444
0.25	-3.3264	5.6444
0.50	-3.3265	5.6444
0.75	-3.3265	5.6444
1.00	-3.3264	5.6443
.	.	.
.	.	.
.	.	.
3599.25	-1.4527	2.9018
3599.50	-1.3978	2.7995
3599.75	-1.3667	2.7375

Table 8.1: Example of (x, y) motion file.

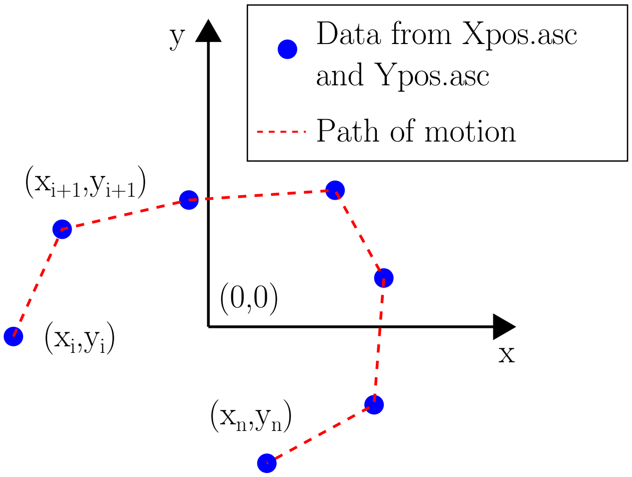


Figure 8.2: Example of vessel motion.

Figure 8.3 is an example of the vessel motion paths when the weather direction, significant wave height and peak period changes. The floater is also displayed in the origin, to show the starting point before the analyses were performed. The weather forces made the floater drift away from the origin, and then it oscillated around a mean value which was not $(0, 0)$. Figure 8.4 is the vessel motion path considering only one weather direction.

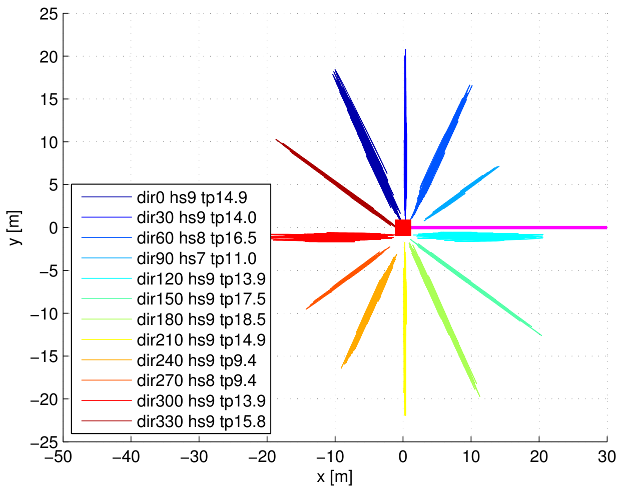


Figure 8.3: Vessel motion of 0-360°

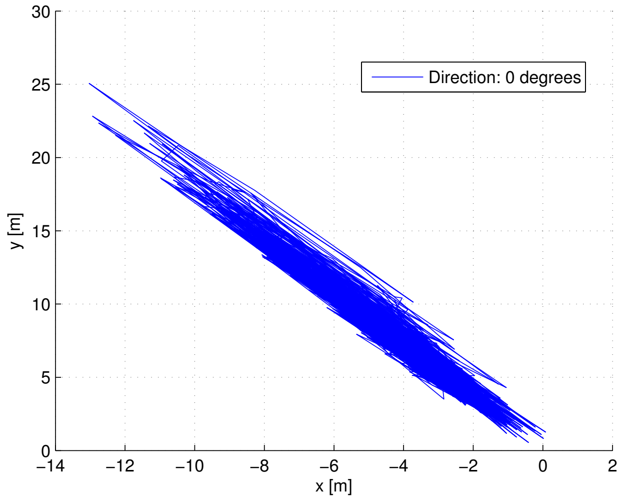


Figure 8.4: Vessel motion of 0°

By looking at the figures above, it is possible to see how the floater moves through the grid lines in the plot. The idea is to count the number of occurrences at specific points around the coordinates system. By having this information, it is possible to create a statistical representation of the vessel motion.

8.4 Transforming the coordinates

A transformation of the vessel motion coordinates $(x_i, y_i) \Rightarrow (X_T, Y_T)$ was done in order to simplify the way of counting the data. The idea was by transforming the data such that all coordinates were positive, it would be easier inserting the occurrences of a certain position in a matrix.

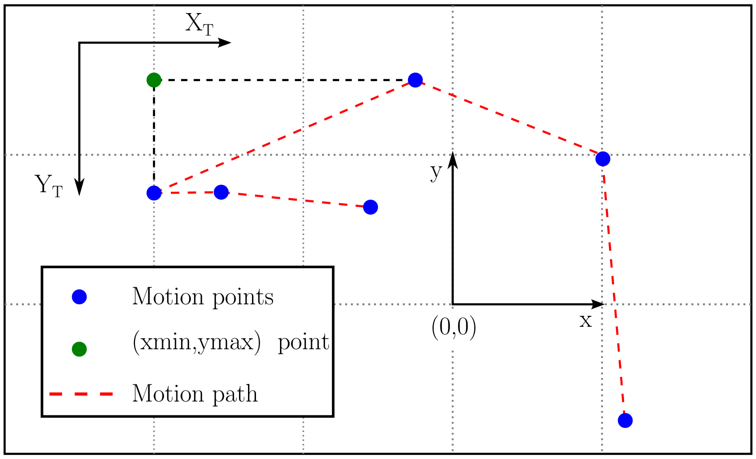


Figure 8.5: Original and transformed coordinate system.

Figure 8.5 shows where the transformed coordinate system is placed. By having the new coordinate system this way, all possible motion data points would become positive by setting the transformed origin to $(x_{min} - \epsilon, y_{max} + \epsilon)$. Where ϵ is a small value. In practice this was done by flooring the x_{min} value and ceiling the y_{max} value. An example is that a (x_{min}, y_{max}) value of $(-3.12, 5.66)$ would give a transformed origin $(-4, 6)$. By having the new, transformed origin, the new coordinates are transformed using

$$X_{T,i} = x_i - x_{min} \tag{8.1}$$

$$Y_{T,i} = y_{max} - y_i \tag{8.2}$$

This was done by the Matlab script *transform_xy_data.m*. All the new coordinates are positive and are floating numbers with only one decimal (ex. 4.2, 5.3, 45.1).

To enable fast counting of the transformed coordinates and hence, be able to find the probability, they are multiplied with a factor of 100. This is to remove all decimal digits. The numbers in the example above are then 42, 53 and 451. A large matrix was therefore created where the number of rows was equal to the largest transformed y-coordinate that was multiplied, while the number of columns was the largest transformed x-coordinate multiplied. A script was created that ran through all the coordinates and added their count into the respective indices into the large matrix.

8.4.1 Vessel motion probability

When all coordinates were gone through and counted, the transformed coordinates were calculated back to the original ones. $(100 \cdot X_{T,i}, 100 \cdot Y_{T,i}) \Rightarrow (X_{T,i}, Y_{T,i}) \Rightarrow (x_i, y_i)$. The probability of each coordinates was found by

$$p(x_i, y_i) = \frac{n(x_i, y_i)}{N_{tot}} \tag{8.3}$$

where

$p(x_i, y_i)$	[–]	The probability of the vessel being at (x_i, y_i).
$n(x_i, y_i)$	[–]	The counted occurrences at (x_i, y_i).
N_{tot}	[–]	The total number of occurrences.

8.5 Summary of the counting procedure

1. Find the x_{min} and y_{max} of the vessel motion coordinates. Example: $(-20.1, 15.3)$. The new origin is then $(-21, 16)$, with the y-axis pointing downwards like shown in Figure 8.5.

2. Transform all the vessel motion coordinates $(x_i, y_i) \Rightarrow (X_{T,i}, Y_{T,i})$. This was done as shown in equations 8.1 and 8.2.

3. The transformed coordinates were then multiplied with a factor of 100 to remove all decimal digits. The coordinates were so run through and the count was added into a large matrix **A** of dimension $n \times m$, where m was the largest $100 \cdot Y_T$ coordinate and n was the largest $100 \cdot X_T$ coordinate.

$$\mathbf{A}_{m,n} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix}$$

i	x	y	X_T	Y_T	$100 \cdot X_T$	$100 \cdot Y_T$	Index in **A**
1	3.145	1.572	7.1	2.4	710	240	$\mathbf{A}(240, 710)$
2	4.345	2.172	8.3	1.8	830	180	$\mathbf{A}(180, 830)$
3	6.897	3.448	10.9	0.6	1090	60	$\mathbf{A}(60, 1090)$
4	-3.456	-1.728	0.5	5.7	50	570	$\mathbf{A}(570, 50)$

Table 8.2: Example of transformed data.

4. The number of occurrences and the probability was found at each coordinate by using equation 8.3.

Where $(x_{min}, y_{max}) = (-3.456, 3.448) \Rightarrow (-4, 4)$.

8.6 Reducing the motion information

When doing an analysis in VIVANA, it was necessary to specify where the riser was connected to the sea floor, and where it was connected to the vessel. The latter meant that the computer program needed an input that was a coordinate of the point where the riser was connected to the vessel. It meant that one analysis with VIVANA was with a certain (x_i, y_i) point. In real life, the vessel or platform moves with time and it was necessary to try and represent this in the fatigue analysis. After running the counting and finding the probability of the vessel motion in Matlab, there were many (x_i, y_i) coordinates which all have their respective probability $p(x_i, y_i)$. If the number of points equals to N_{points}, then N_{points} analyses have to be done with VIVANA. This was not suitable due the total time all the analysis would need.

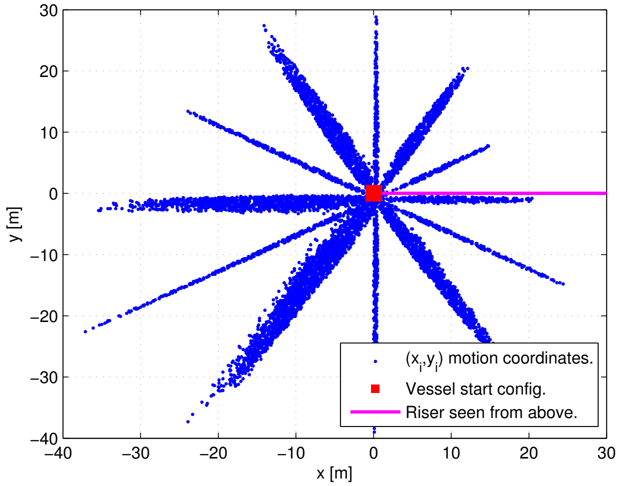


Figure 8.6: Scatter representation of the vessel motion.

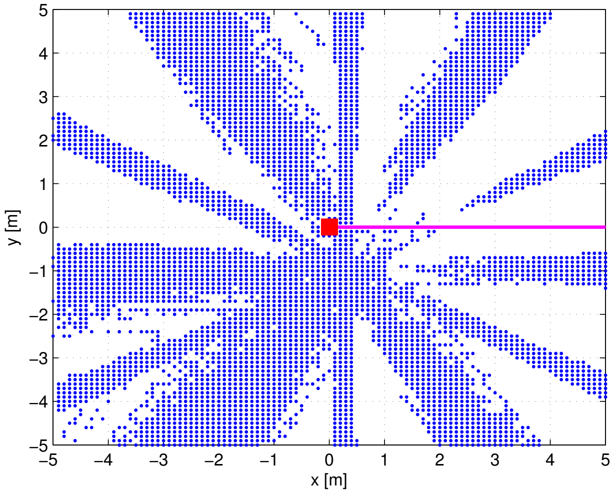


Figure 8.7: Scatter representation of the vessel motion. Close to the origin

Figure 8.6 illustrates all the N_{points} that represent the vessel motion. A probability was found for each point. When running a VIVANA analysis, each point will have its own fatigue results. The idea was to run VIVANA at different coordinates and sum the fatigue with the respective probability such as

$$D_{tot} = \sum_{k=1}^{N} D_i \cdot p(x_i, y_i) \tag{8.4}$$

where

D_{tot}	[1/years]	Summed fatigue damage for all vessel motion coordinates.
D_i	[1/years]	Fatigue damage from VIVANA after a run at RHoP at (x_i, y_i).
$p(x_i, y_i)$	[–]	Probability for at RHoP at (x_i, y_i).

As seen from the figures 8.6 and 8.7 there are many points in the calculations. The total vessel motion consisted of $N_{points} = 34273$ points. To go through all the data, the same amount of VIVANA analyses should have been performed. As mentioned before, this is not realistic due to time limitations. A way of doing the analysis with the data possessed is to sum up the vessel motion data into defined, discrete points.

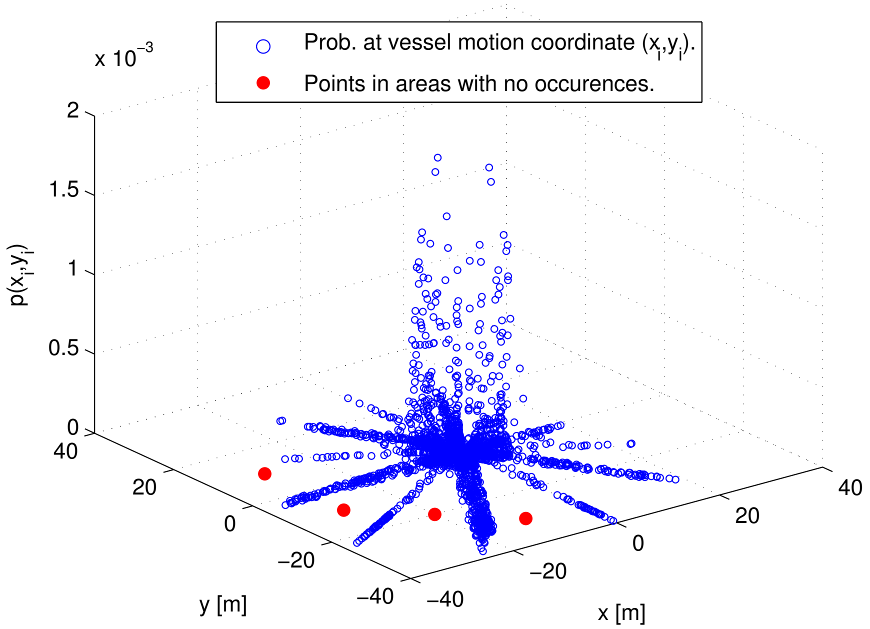


Figure 8.8: Probability of motion coordinates.

The above figure illustrates the probability at all motion coordinates, (x_i, y_i). Some values have not been taken into the plot due to issues of figure size. The coordinates have been reduced from a total number of 34274 to around 2500. However, the plot describes well the probability of the motion. The main idea with the plot was to show how many motion points there are and that doing an analysis for every point would taken too much time. That is the reason to add the probability of the blue points in Figure 8.8 to the discrete points shown in Figure 8.14. The red points in the figure above are examples of areas where the SIMO analysis had no occurrences.

8.6.1 Discrete coordinates

In order to do practical VIAVANA analyses with the vessel motion data, there was a need of reducing the number of vessel motion points, without loosing the probability of the movement. The idea of the discrete points was to sum the probability of the coordinates into discrete points which were defined in a specific way. The discrete points were created in sectors that went 360° around in order to include all the data. Every sector was divided into a specified number of parts, and each part in the sector had a discrete point. The theory is that if a vessel motion coordinate is within a specific sector and its respective part, the probability is counted to the discrete point representing this area. Figures 8.9 and 8.10 illustrates how a sector could be defined. In Figure 8.11 an example of four sectors with an angle of 90° each is illustrated.

The arrows in the figure demonstrate to which discrete point the probability of the vessel motion coordinates is summed to.

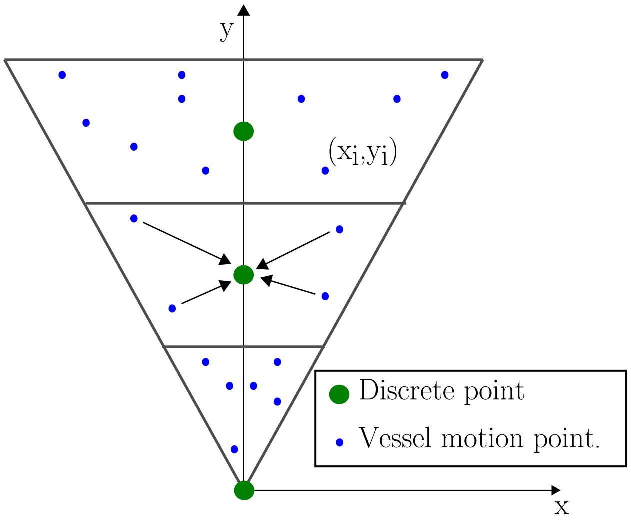


Figure 8.9: Sector with three part.

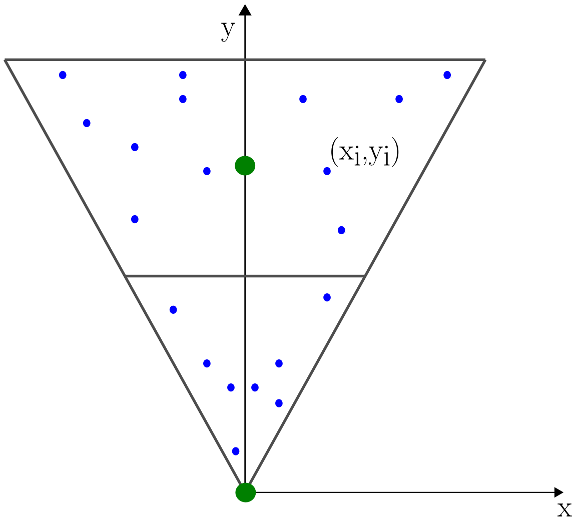


Figure 8.10: Sector with two parts

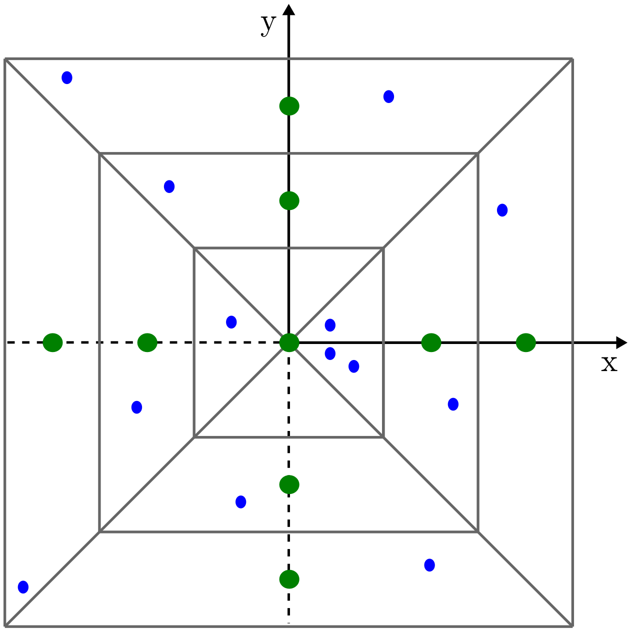


Figure 8.11: Example of four sectors of 90° each that cover all the vessel motion points. Each sector consists of three sector parts.

A Matlab script called *search_xy_in_sectors.m* was constructed to create the discrete points, and count the probability of the vessel motion coordinates into them. The function is programmed such that the user can choose the angle of each sector, and how many sector parts there can be in one sector. The examples on page 67 show one sector with three parts while the other only has two parts. This is something that is important to discuss about, because the number of parts might alter the statistical data in the discrete points. This is further discussed in Section 8.10.

Every first part of the sectors has a discrete point in the origin. The next parts have a discrete coordinate at the midpoint in their respective part. The sectors are designed such that the probability of the vessel motion coordinate which are within the first sector part are counted into the discrete point in the origin. The vessel motion coordinates which are inside the other sector parts will be counted into the respective discrete coordinate.

The analysis that was done in this work consisted of 30° sectors which gave a total of $360°/30° = 12$ sectors in order to capture all of the vessel motion coordinates. The radius of the sector was set to be equal the largest, floored radius of a vessel motion coordinate. An example is that if the largest radius was 32.9 [m] then the radius of the sectors was set to 33.0 [m].

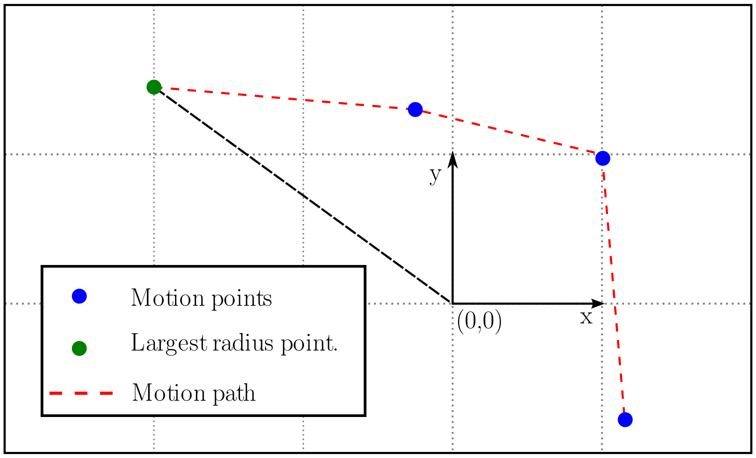


Figure 8.12: Largest radius of a vessel motion point.

The number of sector parts was set to 12. The largest radius of the sector was 45 [m], which means that each sector part was 3.75 [m]. The discrete points looked like in Figure 8.13. This is the coordinates (x_i, y_i) which were used to run the VIVANA analysis. The total number of discrete point was 133. However, since the radius of each sector was based on the largest radius found from the vessel motion coordinates, it was not certain that all points had a probability. This will is further discussed in the following pages.

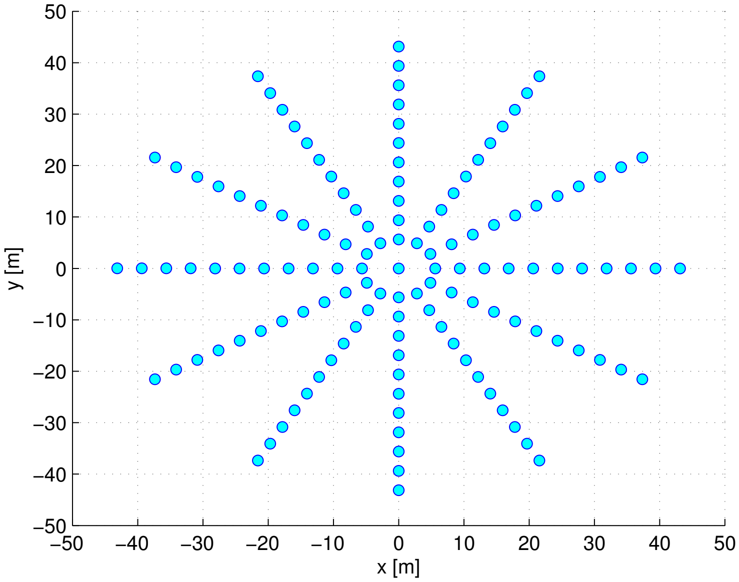


Figure 8.13: Discrete (x, y) points.

The probability of the vessel motion coordinates was then summed to respective discrete points. Figure 8.14 illustrates how the probability is distributed around at the discrete coordinates.

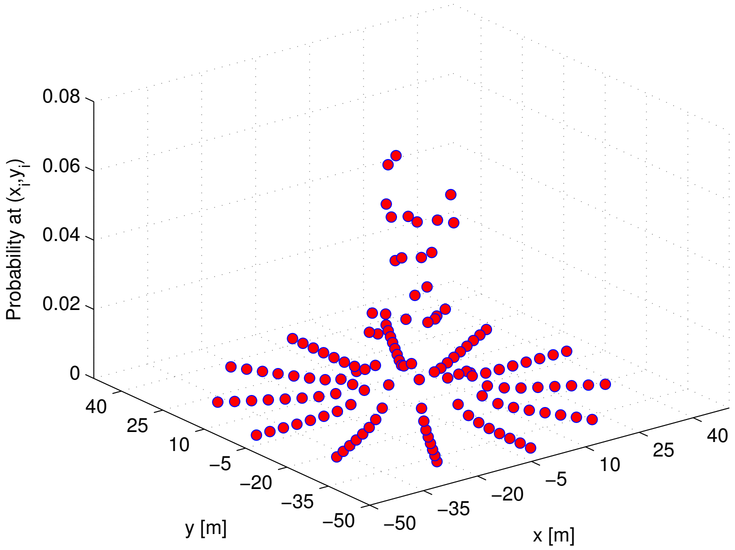


Figure 8.14: The probability of the discrete coordinates.

It is not easy to see how the probability is distributed along the discrete coordinates by looking at the figure above. A way of having a better understanding is to interpolate these points and create a surface and contour plot of with the probability at the discrete points. It is important to note that the surface and contour plots have only the correct probability at the discrete points. The surface is hence not a probability distribution of the vessel motion, but only a good way of illustrating how the probability at the discrete points is distributed.

The contour plot shows the contours of the surface plot.

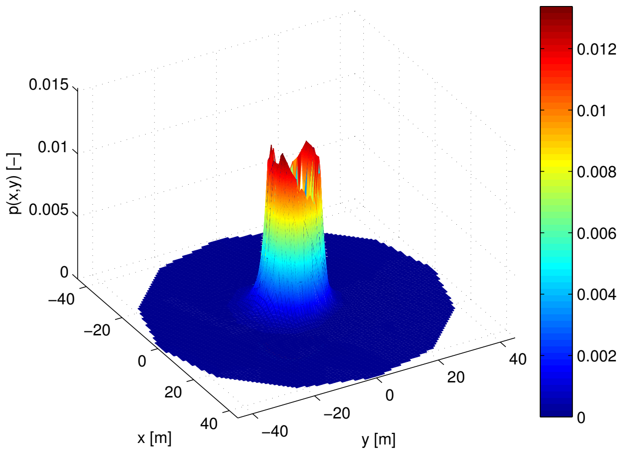


Figure 8.15: Surface plot.

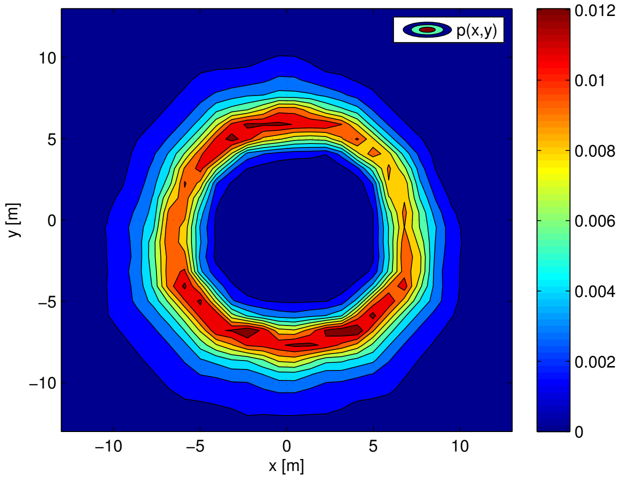


Figure 8.16: Contour plot.

By looking at the contour and surface plot, it is seen that the largest peak is Northwest to the origin. This was explained by the way the SIMO analysis was done and what scatter diagram was used. In this thesis the details about the weather are not important, it is the effect of the vessel coordinate variation that is looked upon.

Viewing at the discrete points probability from Figure 8.14 and looking at the contour, it is possible to see that there is a large difference between the probability of the outer points and the points that are within a certain radius near the origin. It looks as the vessel is mostly located at a position that is within a radius of 5-10 [m]. Since VIVANA performed a calculation one discrete point at the time, it was smart not to use all the discrete points because of their relative low probability.

By setting a probability limit to $p(x_i, y_i) \geq 0.00009$ some points were not taken into account without having a great effect on the results. The points that were used in the analysis are illustrated as blue dots in Figure 8.17. Quite many points were left out of the analysis. Some points also had zero probability. That is because of the way the sector counting technique was made. More about this is discussed in Section 8.10.

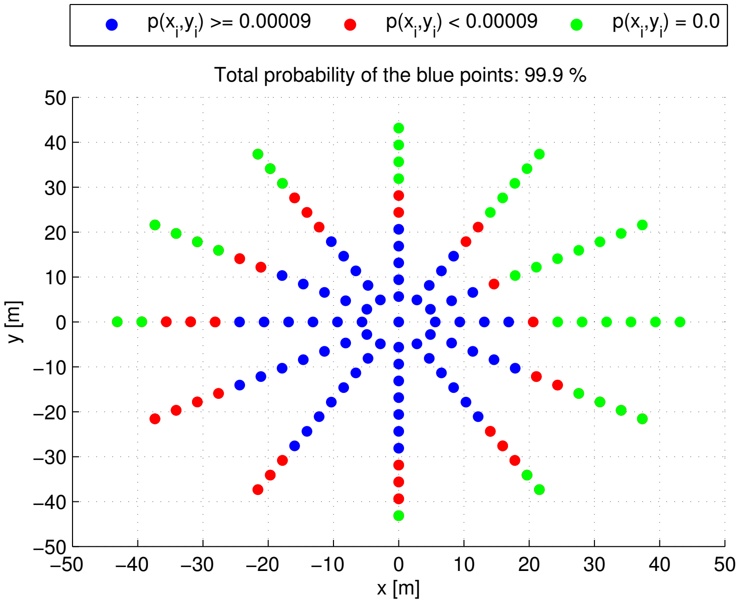


Figure 8.17: Discrete points used in analysis.

Only using the discrete points with a probability larger than 0.00009, the number of analyses performed were reduced from 133 to 66.

8.7 Calculation steps.

To get the results for the coordinate variation analysis, a RIFLEX/VIVANA analysis was done for each blue coordinate as seen in Figure 8.17. For this to be done in an efficient way, a BATCH-script was used to make the analysis go automatically. The BATCH-script used several Matlab functions in order to insert necessary values in the input files files needed in RIFLEX and VIVANA. The flow chart showing this procedure is illustrated in Figure 5.4. A variations.txt file with inputs needed to perform the analysis was used. The file had information about the coordinate of the RHoP and the probability for that coordinate. The coordinates were inserted into the respective RIFLEX input file, and a VIVANA analysis was performed in the end. When all the RHoP coordinates were analysed, a Matlab-script summed up all the fatigue damages multiplied by the probability in order to get the final fatigue result.

A part of the variations.txt file is displayed in the following.

#Id	$CURDIR	$XPOS	$YPOS	$PROB	$XNAME	$YNAME	$PROBNAME
Run_1	90.0	0.0	0.0	8.559E-03	0_0	0_0	0_008559
Run_2	90.0	5.6	0.0	3.910E-02	5_6	0_0	0_039098
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
Run_65	90.0	14.6	-8.4	1.114E-03	14_6	-8_4	0_001114
Run_66	90.0	17.9	-10.3	2.911E-04	17_9	-10_3	0_000291

Table 8.3: variations.txt file for the coordinate variation.

The parameters are explained below.

#Id	[–]	Run identifier. Used to order the results numerically.
$CURDIR	[°]	The direction of the ocean current. The definition is shown in Figure 1.4.
$XPOS	[m]	The x coordinate of the RHoP.
$YPOS	[m]	The y coordinate of the RHoP.
$PROB	[–]	The probability of RHoP at ($XPOS,$YPOS).
$XNAME	[m]	$XPOS in another format. Used for folder naming.
$YNAME	[m]	$YPOS in another format. Used for folder naming.
$PROBNAME	[m]	$PROB in another format. Used for folder naming.

When the BATCH-script is run, it will extract one line of the parameters for each loop. The parameters are then sent to a Matlab-script called *change_str_in_file.m*, which will insert them into the correct input file needed by RIFLEX (inpmod.inp) or VIVANA (vivana.inp). From Table 8.3 the Matlab-script will insert $XPOS and $YPOS into the input files. The id, xname, yname and probname are used to create a folder where the fatigue results are.

A typical folder name would be: Run_1_xpos0_0ypos0_0prob0_008559.

In every folder there were result files (.res) from the RIFLEX and VIVANA analyses. The matrix plot files (.mpf) are deleted in order to save space on the hard-drive. The most important results, the fatigue results, were saved in the matrix plot file. A Matlab-script was therefore created to extract the fatigue results from the vivana.mpf file (which could be around 20-30 Mb) before they were deleted. The new file was called fatigue.txt which looked like
The arclength [m] is the riser length from the RHoP. The fatigue damage unit is [1/years] which is the annual fatigue at the worst point on the circumference at the respective arclength. A typical plot of the fatigue damage on the riser is shown in Figure 8.18.
From the figure above it is seen that the largest fatigue is the proximity of the TDP. The fatigue goes then to zero somewhere after the touchdown point. The idea behind varying the (x_i, y_i) coordinate of the RHoP, is that each run with a different coordinate would have a different fatigue damage due to the change in the TDP. Hopefully the results would show a fatigue damage that is distributed differently and with a smaller peak.

```
#Extracted data from .mpf file.
#Max accumulated damage.
#Arclength [m]   Fatigue damage [1/years]
      0.0               4.4218E-06
      1.0               5.7204E-06
      2.0               7.8668E-06
      3.0               1.0516E-05
       ⋮                    ⋮
   1997.5               6.2072E-10
   1998.5               5.7709E-10
   1999.5               5.3116E-10
   2000.5               4.8875E-10
```

Table 8.4: Example of a fatigue.txt file.

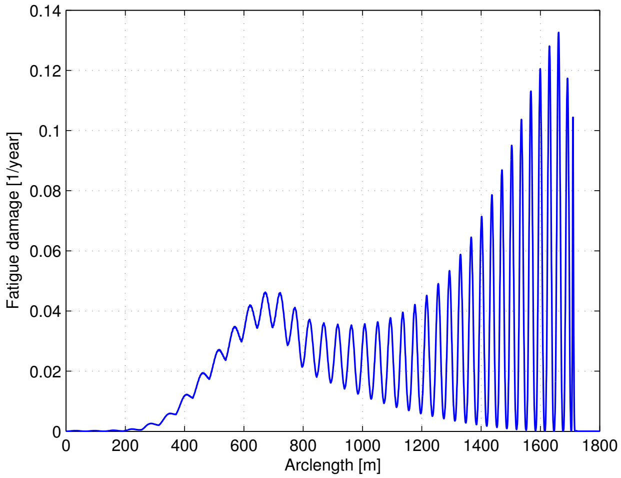


Figure 8.18: Example of fatigue damage on riser.

8.8 Current profile probability

When calculating the fatigue damage that is inflicted on a slender structure over several years it is important to represent the statistics of the loading in the most realistic way. The probability of a current profile decides the annual damage to the structure caused by the profile. There has been discussions this semester if the current probability was to be taken into account when running the variation of vessel motion probability. This would most probably given different fatigue results, because different modes would have been used excited for different current profiles. The total annual fatigue damage per year would have been

$$D_{annual} = \sum_{i=1}^{M} p(x_i, y_i) \sum_{j=1}^{N} p_{j,current} \cdot D_j(x_i, y_i) \tag{8.5}$$

where

$p(x_i, y_i)$	[–]	Probability for floater coordinate at (x_i, y_i).
$p_{j,current}$	[–]	Annual probability for current profile j.
$D_j(x_i, y_i)$	[1/year]	Annual fatigue damage when floater at (x_i, y_i), when current j is acting.

The idea was never realized in this master thesis. In the work, the final fatigue damage was found by

$$D_{annual} = \sum_{i=1}^{M} p(x_i, y_i) \cdot D(x_i, y_i) \tag{8.6}$$

Due to lack of time and current profile data, there has not been a chance to do fatigue calculations with VIVANA where the current probability was taken into account. The results in Chapter 8 are based on just one current acting. This is of course conservative, especially since the current profile used in the most analyses has large velocity values. The current profiles used in the analyses can be found in Table 7.1. In the future work with risers and VIV fatigue, analyses with different current profiles and vessel motion will be investigated.

8.9 Effective tension

8.9.1 Effects on fatigue damage

The effective tension of the riser will vary as the floater moves in the XY-plane. This is seen by the results from VIVANA. When the vessel is located at a position close to the TDP, the effective tension in the riser will be smaller compared when the riser is further away from the TDP. This is explained easily, because when the vessel moves further away from the point where the riser touches the sea bed, a larger part of the riser will be lifted from the sea floor. This will increase the weight needed to be lifted by the floater. If the riser is closer the TDP, more of the riser will touch the sea floor, and the effective tension will decrease. The effective tension will have an effect on the riser response induced by the ocean current. If the riser is stretched, the effective tension will increase and the structure will need more energy to be deformed. The VIV response will thus decrease with increased effective tension and vice versa.

The effective tension of the riser is illustrated by the following figures.

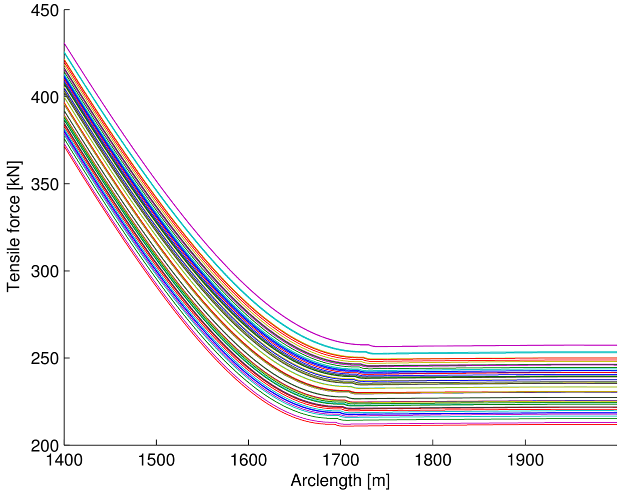


Figure 8.19: Effective tension of the lower riser part.

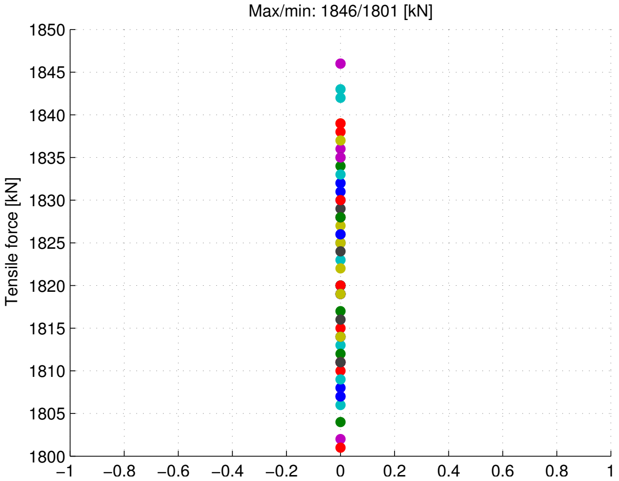


Figure 8.20: The effective tension at the RHoP for all the runs.

The effective tension is greatest the RHoP and decrease linearly towards the sea bottom. It is of great importance to have a positive effective tension in order to avoid collapse of the riser. Figure 8.20 shows the difference in effective tension at the RHoP for the different runs. The difference between the maximum and minimum effective tension at the RHoP is 45 [kN].

8.10 Sector discretisation issues

When the probability is found for every coordinate that the vessel motion goes through, the result is messy and it not easy to run a VIVANA analysis because of the great number of coordinates (look at Figure 8.8). The way of dealing with this is to create discrete points that cover all possible vessel motion points, and then add the probability to them. However, it noticed that changing the way of discretisation, alters the probability at the discrete coordinates.

The discretisation used in the analysis consisted of the sector data displayed in the following table.

Sector angle:	[deg]	30.0
Number of sectors:	[–]	12
Sector radius:	[m]	45.0
Sector parts:	[–]	12
Length of part:	[m]	3.75

Table 8.5: Add caption

If the sector data used in the analysis consisted of different parameters the Matlab-function created to count the probability of the vessel motion coordinates, could give a different representation of the probability at the discrete points. This is easily seen looking at sectors with different number of parts.

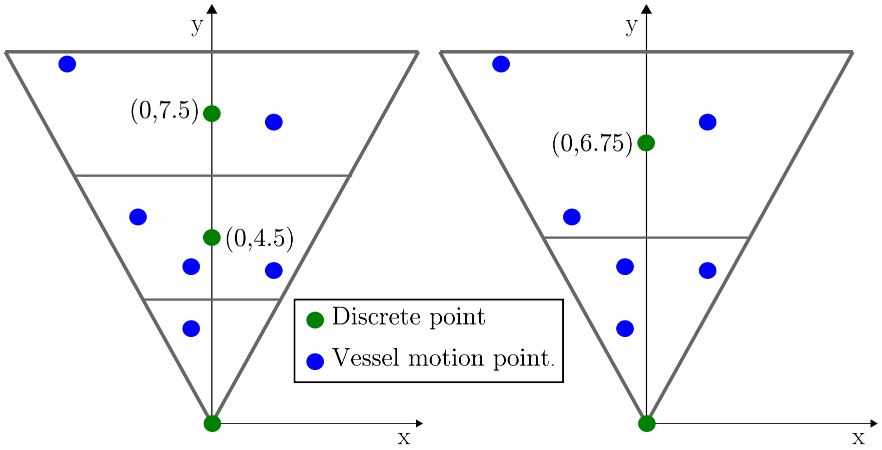


Figure 8.21: Difference of sector parts.

The figure above gives an example of how the probability at the discrete points might change, depending on the sectors used. In both sectors, the vessel motion points are the same. There is a total of 6 blue points that define the motion. The left sector has the same radius and angle as the right hand one. The difference is in the sector parts. There is a discrete point at the origin of every sector. The sector on the right has only one additional discrete point at coordinate $(0, 6.75)$. The left hand sector has two. They are placed at coordinates $(0, 4.5)$ and $(0, 7.5)$. The probabilities at the discrete points (green) are found by summing all the probabilities of the vessel motion on points (blue) inside the respective sector.

If all the blue points in the figure above have the same probability, $p(x,y_i) = 1/6$ the probability of the discrete points in the two sectors would be different.

Discrete point	Left sector	Right sector
$(0.0, 0.0)$	0.17	0.50
$(0.0, 4.5)$	0.50	–
$(0.0, 6.75)$	–	0.50
$(0.0, 7.5)$	0.33	–
$\sum$ prob.	1.00	1.00

Table 8.6: Probability at the discrete points.

From the table above it is clear that the probability is different at the discrete coordinates. Looking at the probability at the discrete point in the origin, it is noticed that the probability changed with a factor of 2.94. Increasing the number of sector parts the probability of the discrete points would go towards the probability found at each and every vessel motion point. This is illustrated in Figure 8.23. Thus, it is important to use the probability results with caution.

8.10.1 Probability variation

In order to show how the discretisation effects the probability, differences in the number of sector parts have been used to illustrate this. The angle value has not been changed since the analytical results from the SIMO analysis had discrete degree values of the motion with 30° intervals. The following figures demonstrates how the probability at the discrete coordinates alters as changes in the number of sector parts are varied.

Figure 8.22 and 8.23 reveal how the coordinate of the greatest probability changes with the number of sector parts. The x-coordinate is quite stable, except that it has a jump to around 4.2 [m] when the sectors used have 30 parts. This is quite interesting. The discretisation with 30 parts must have been unique and captured a specific area with probability that changed the x-coordinate to an special point. This demonstrates one of many probable differences that alterations in discretisation may give. The coordinate is stable around $x = 0$ [m] elsewhere, which is satisfying. The plot the y-coordinate reveals something noteworthy. It is stable around $y = 0$ [m] until 8 parts. From 9 parts and further the y-coordinate varies between a value of around 7 to (-7.5) [m]. This is not satisfactory. However, as mentioned in Section 8.9.1, the fatigue damage is highly influenced by the riser tension, but the tension does not vary considerable in the y-direction as seen from Figure 8.29. This means that the result from the variation of vessel motion coordinates will not be influenced much due to these alterations in y-direction.

The last noteworthy remark is how the maximum probability changes as the number of sector parts alters. This is displayed in Figure 8.24. One can see that the maximum probability converges relatively quickly. When the number of sector parts is lower than 9, the maximum probability will be in the origin an the contour plot will be quite symmetric as illustrated in Figure 8.26a. The reason why the maximum probability is smaller for an increased number of sector parts, is because there is an increased number of discrete points in the sector. The probability will hence be distributed over a greater number of discrete coordinates. The maximum probability will be less, but the $\sum p(x_i, y_i)$ will be equal.

The straight, magenta lines in the three following figures displays the x-coordinate and y-coordinate of maximum probability from the raw data, and the value of it. The probability distribution of the raw data is illustrated in Figure 8.8. The table on page 79 displays the results from Figure 8.24 as numbers.

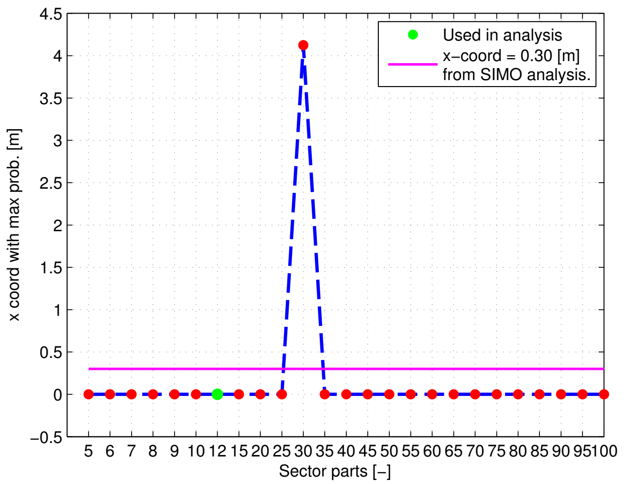


Figure 8.22: x-coordinate of maximum probability.

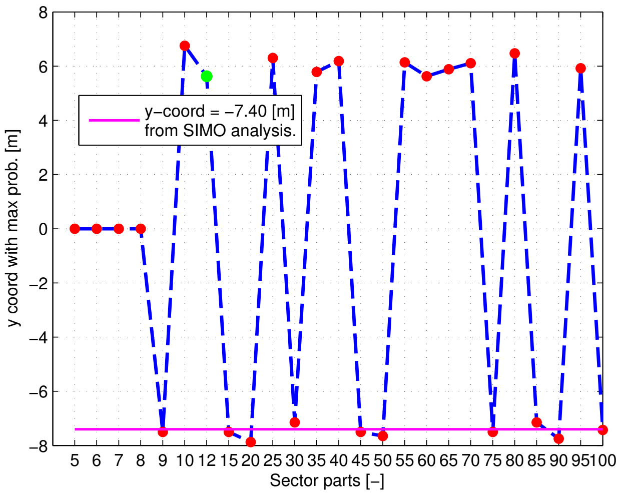


Figure 8.23: y-coordinate of maximum probability.

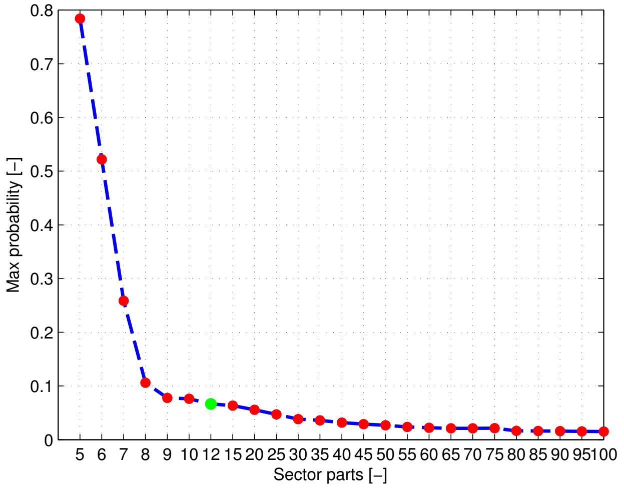


Figure 8.24: Maximum probability convergence.

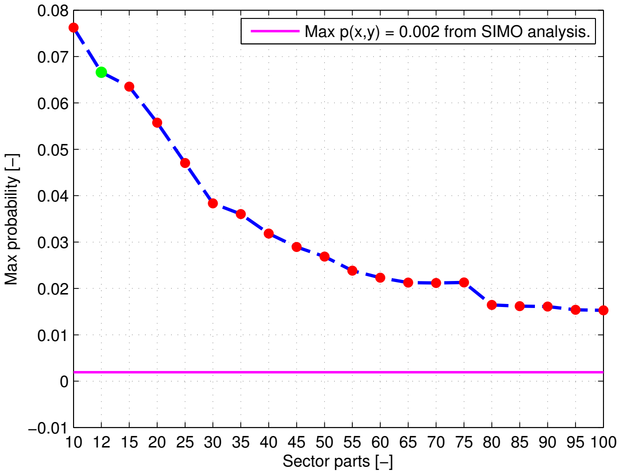


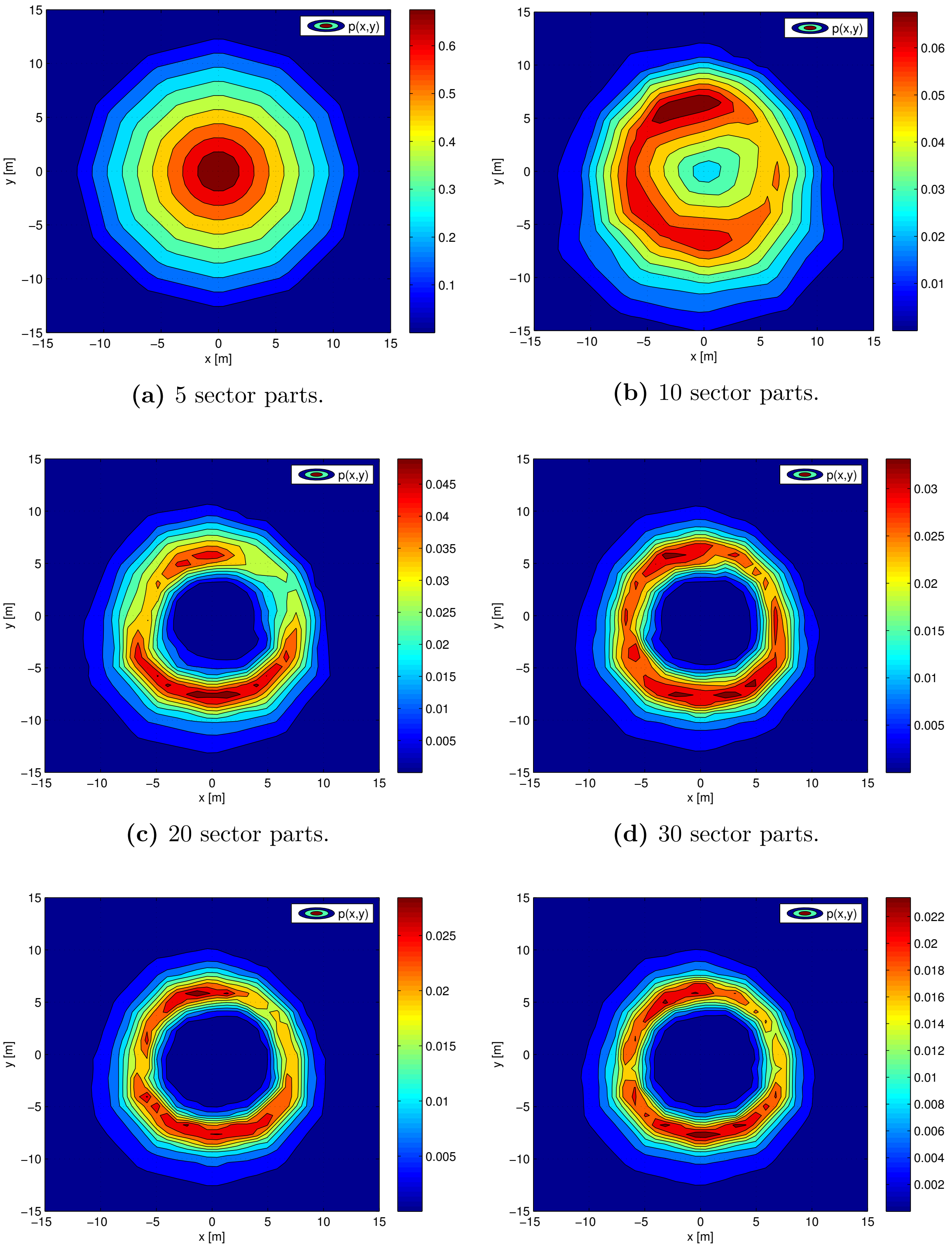
Figure 8.25: Maximum probability convergence (closer).

The way the probability alters at the discrete points when the number sector parts is changed is also viewable as contour plots on page 80. The contour are shown for 5, 10, 20, 30, 40, and 50 sector parts. The figures illustrate that in the beginning, when there is only 5 parts, the maximum frequency is in the origin. The surface looks like a symmetric cone. This is reasonable since the most probable values are within the borders of the first sector parts. The probability has a similar appearance up to sectors with eight parts. From eight parts and further, the main probability is assembled in a ring around the origin. This is a more accurate representation of the results given by DNV (SIMO analysis). However, it is important to point out that this ring of probability changes for every time the number of sector parts is increased, but that the differences are small.

There was not enough time to do analyses with sectors having more than 12 parts. This could have demonstrated alterations in the fatigue results.

Sector parts	x	y	Max $p(x,y)$
5	0.00	0.00	0.78
6	0.00	0.00	0.52
7	0.00	0.00	0.26
8	0.00	0.00	0.11
9	0.00	-7.50	0.08
10	0.00	6.75	0.08
12	0.00	5.63	0.07
15	0.00	-7.50	0.06
20	0.00	-7.88	0.06
25	0.00	6.30	0.05
30	4.13	-7.14	0.04
35	0.00	5.79	0.04
40	0.00	6.19	0.03
45	0.00	-7.50	0.03
50	0.00	-7.65	0.03
55	0.00	6.14	0.02
60	0.00	5.63	0.02
65	0.00	5.88	0.02
70	0.00	6.11	0.02
75	0.00	-7.50	0.02
80	0.00	6.47	0.02
85	0.00	-7.15	0.02
90	0.00	-7.75	0.02
95	0.00	5.92	0.02
100	0.00	-7.43	0.02

Table 8.7: Number of sectors, maximum probability and its coordinates.



(a) 5 sector parts.

(b) 10 sector parts.

(c) 20 sector parts.

(d) 30 sector parts.

(e) 40 sector parts.

(f) 50 sector parts.

Figure 8.26: Alterations of the probability contour.

8.11 Results

66 analyses were performed with VIVANA. Each analysis had its own probability $p(x_i, y_i)$ and respective coordinate (x_i, y_i) for the RHoP. This means that the total fatigue damage was found by multiplying the fatigue from each run with its respective probability and then getting the final fatigue damage by summing all the runs as shown with Equation 8.4. The final results are illustrated in figures 8.27 and 8.28.

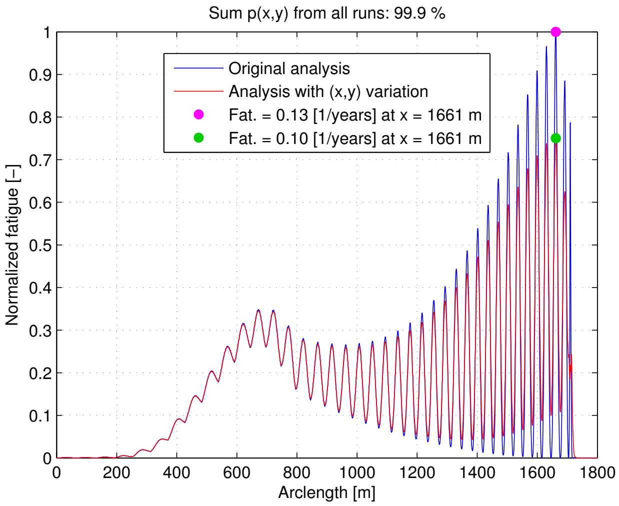


Figure 8.27: Normalized fatigue. Variation analyis vs. original.

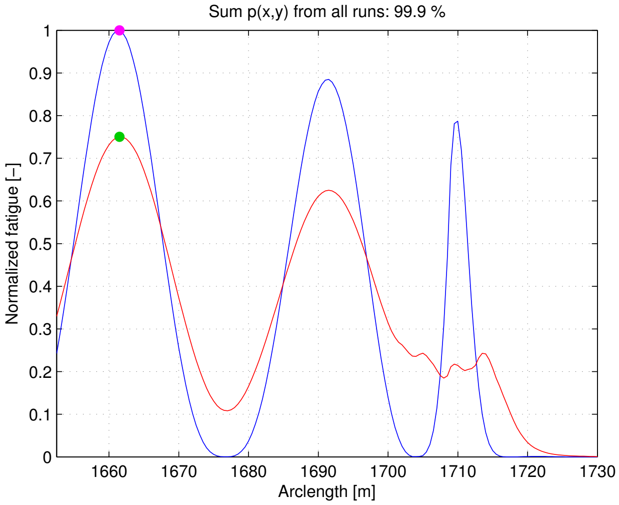


Figure 8.28: Closer look at TDP.

The results demonstrate that there is a difference in the fatigue damage between the two analyses. The original VIVANA analysis was with the RHoP at the origin only, while the variation analysis was performed for many different offsets. Around a length of 1000 [m] it is possible to notice a difference in the results. The maximum fatigue damage from the variation of coordinates (x_i, y_i) was around 75 % of the original maximum fatigue. Figure 8.28 shows a closer look at what happens with the fatigue damage near the TDP. It shows how the variation of the vessel coordinates has distributed the fatigue close to the TDP, such that there are several smaller peaks between 1705-1720 [m]. The difference in the fatigue damage is because of the difference in the tension of the riser. As the vessel moves further away from the TDP, more of the riser will be lifted from the sea floor, and the tension will increase. The opposite will be the case if the vessel travels towards the termination point of the SCR.

Analysing the effective tension at the different coordinates, it showed that the effective tension does not change significantly with the y-coordinate. The greatest effect is seen when the vessel moves towards and away from the TDP in x-direction. This is pointed out in the following figure.

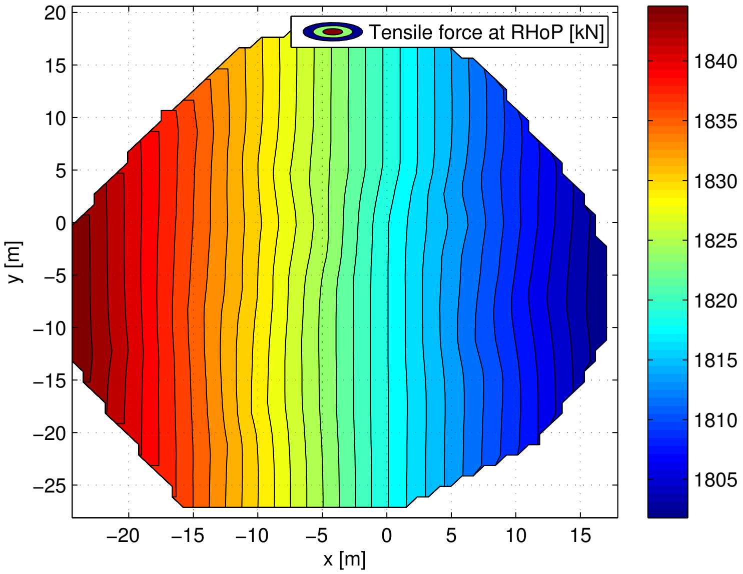


Figure 8.29: Contour of the effective tension when RHoP is at (x_i, y_i)

The contour lines are almost parallel with the y-axis, which means that there is no great change in the y-direction.

To figure out why the fatigue damage is smaller in the variation of RHoP coordinate analysis, it was necessary to find out how much of the total time the vessel was at a position that was $x > 0$, $x < 0$ or $x = 0$.

Analysing the data, following probabilities were found.

Postion [m]	**Prob. of pos. [-]**
x > 0	35 %
x < 0	45 %
x = 0	20 %

Table 8.8: Probability of x-position of the vessel.

Tension at RHoP [kN]	**Prob. of tension [–]**
$\mathrm{T_{RHoP} > T_{orig}}$	54 %
$\mathrm{T_{RHoP} < T_{orig}}$	35 %
$\mathrm{T_{RHoP} = T_{orig}}$	11 %

Table 8.9: Probabilty of tension at RHoP.

$\mathrm{T_{orig}}$ is the effective tension from the original analysis where the RHoP is located at $(0, 0)$. $\mathrm{T_{orig}} = 1819$ [kN].

Table 8.8 describes that the vessel is most likely to be at a position which is $x \leq 0.0$. This implies that the effective tension in the variation of RHoP coordinates analysis has an effective tension at RHoP which was greater, most of the time, compared to the original analysis. This is pointed out in Table 8.9. The tension is below $\mathrm{T_{orig}}$ only 35 % of the time. This is satisfactory with the fatigue results from Figure 8.27 and 8.28 which illustrate smaller fatigue damage with the variation of RHoP coordinate.

It is also interesting to note that the fatigue results have the similar mode acting. This can be explained by Equation 8.7 (prof. C.M Larsen, pers. comm).

$$\omega_i = \frac{i\pi}{l}\sqrt{\frac{T}{m}} \quad , i = 1, 2, 3..., n \tag{8.7}$$

where,

ω_i	[1/rad]	$\mathrm{i^{th}}$ eigenfrequency.
i	[–]	Mode number.
l	[m]	Length of riser.
T	[N]	Top tension.
m	[kg/m]	Mass per unit length.

When the length of the riser not touching the sea bed increases, which happens when the vessel moves away from the TDP, the tension will also increase. These two factors seem to cancel each other out such that the dominating eigenfrequency ω_{orig} and $\omega_{variation}$ stay similar.

Chapter 9
Conclusions

The work done in this master thesis was mainly to assess what affects the fatigue damage of a steel catenary riser near the touchdown point. The motivation for this work is the fact that more and more activity in the oil and gas industry developments is moving into deeper waters. This means that new riser designs are needed to tackle this development. As the industry is going to deeper areas, the ocean currents will play an important role in riser design. There are also issues doing research on this subject because of the scale that is needed to simulate ultra-deep waters. A consequence of this is that current models need to be predicted in a better way, which includes variability both in direction and magnitude as a function of time. And in order to be able to prevent environmental disasters, there is a necessity to be capable to design risers that can withstand extreme.

The key studies in this work have been.

- A parametric variables study of the fatigue on a SCR.
- A fatigue analysis of the a SCR considering floater motion statistics.

9.1 Parametric study

The parametric study was done with a total of 22 runs, where six different variables were altered. In the four first runs, the current magnitude was looked upon to see what effects it had on the fatigue. It was noticed that the fatigue damage increased for larger current velocities, except for the last and greatest current. Here, the fatigue damage went down, and the point on the riser with this fatigue moved as well. This was explained by which modes were excited. It seems that they may alter the point with maximum fatigue in such a way that the touchdown point does not experience the largest damage. In the three next runs, the current direction was varied while the magnitude was kept constant. For a straight riser this would not have had any effect since it is symmetrical in all planes. A SCR on the other hand is not, and this is demonstrated in the results. The greatest damage was found when the current came perpendicular on the riser plane in the positive x-direction (0°). There is also a difference between 0° and 180°, which is explained by how the current is decomposed in the different directions. VIVANA will use a current velocity which is decomposed such that the current velocity is orthogonal on the respective riser element. The current coming from 90°, which is orthogonal to the riser plane, gave the largest fatigue damage on the greater part of the riser. The only exception was at the touchdown point. The parameter that gave greatest variations between the largest and the smallest fatigue was the choice of S-N curve used as input in VIVANA. The S-N curve used in Run 20 gave almost no fatigue damage due to its relatively high interception of the log N-axis. On the other side, the S-N curve in the last run gave the largest fatigue damage registered in all of the parametric variables checked.

9.2 Variation of vessel position

The variation of the vessel coordinates was done by performing many VIVANA analyses, each run having its respective coordinates (x_i, y_i) and probability $p(x_i, y_i)$. The coordinates and the respective probabilities were found by processing a vessel motion analysis done in SIMO. The final fatigue damage was then found by summation of all the individual VIVANA analyses multiplied with their probability. The results illustrated that the fatigue damage was less compared to simply doing one VIVANA analysis where the riser does not move. The maximum damage was seen to be around 77 % of the original analysis. The peaks in the fatigue results nearest to the touchdown point were distributed in the vicinity. It was also noticed that if the floater was moved to different coordinates in the XY-plane, the mode did not change for the final result. By viewing the results, the modes looked similar. The reason was found to be that a change of tension in the riser would counteract the change in length of the riser above the sea floor. These two factors seemed to counteract each other in a way giving a similar response.

Chapter 10

Discussion

The parametric study where different variables were changed was done in order to understand how the mathematical model responds when being exposed to changing key variables. Doing so, it was possible to learn which variables had a great impact, and where this impact was. When a fatigue analysis is performed, one of the most important results is where and how large the maximum fatigue damage is. The mathematical models of today have many shortcomings with describing the fatigue realistically. As an example, the current model used in this work was unidirectional, meaning that the fatigue damage was concentrated around a small area of the riser circumference. The interaction between the sea floor and the riser was modelled with linear springs in VIVANA, which gave conservative results in the touchdown zone. In the work it has not been enough time to do analysis with changing current directions and magnitudes. Also, when carrying out VIV analyses on steel catenary risers, it is possible to use only four current directions. The empirical computer program VIVANA was created by doing experiments on a symmetric pipe. The results are therefore difficult to use for a general riser configuration. The reason for this is that if the ocean current would come from another direction than the four mentioned, the excitation coefficients would be coupled. As of today, there does not exist a good understanding of this problem.

One of the main subjects investigated in the work was to look at how the fatigue results would respond by moving the floater based on floater statistics. This model was created by use of motion data that consisted of time series with coordinates (x_i, y_i) at many time steps. Thie model was created by the co-supervisor from DNV, and the computer program SIMO was used for the task. The results in this thesis are hence not from a real floater motion. By looking on the statistic of the motion, there was no motion in some areas. The reason is because the SIMO analysis was done in such a way that the produced motions were only concentrated in small direction intervals. A real motion would most probably differ from the SIMO analysis. It is therefore crucial to look more into this issue. Another question was how should one analysis be performed? The original analysis, where the floater was located in the origin and restrained from moving, was done in VIVANA without considering other offsets. The statistical results show that the vessel was most likely to be at a coordinate which was not in the origin. Should therefore the original analysis have been with the floater at the coordinate with the greatest probability? And how would the results have been changed? This is a task that needs to be more examined.

A great part of the work in this thesis is mostly a study on how the fatigue damage is affected by moving the floater. The question in the end was if the results produced in this work were realistic. It is difficult to answer, but it indicates that the challenge with fatigue damage near the touchdown zone needs more attention.

Chapter 11

Recommendation for further work

I recommend that further work in this area is concentrated around three different areas. Firstly, there is a need of doing a considerable amount of work to establish well formulated and relevant design current conditions for offshore slender structures. It is necessary to be able to represent the nature in the best way as possible, such that engineers can design solutions that can operate in an environment with variable current conditions. The design currents need to be based on measured data, and simulations need to be done which can give good predictions.

Secondly, there needs to be a good understanding of how to build up mathematical models to represent structures such as steel catenary risers. Which parameters are of crucial importance in such a way that there might be large differences in the results? Work needs also to be done predicting the vortex induced vibrations in a more realistic way. The research effort needed to learn more about this phenomenon and how it affects the underwater structure is of highly importance when the industry is going into greater depths. This is of particular importance for non-symmetrical risers.

Finally, there is a need of performing vortex induced fatigue analyses where the floater motion is taken into account. The statistical models used in these types of analyses need to represent the motion of a floater in the most realistic way. In order to have any knowledge if these types of analyses give reliable results, there needs to be ways to verify the outcome. Does the variation of vessel motion coordinate give realistic fatigue damage? Will the change in the TDP position distribute the damage as found in this work?

Bibliography

[1] Mekha, B. New frontiers in the design of steel catenary risers for floating production systems. *Journal of Offshore Mechanics and Arctic Engineering*, 2001.

[2] Lie, H., Larsen, C. M., and Tveit, Ø. Vortex induced vibration analysis of caternary risers. Offshore Technology Conference in Houston, 2001.

[3] Larsen, C. M and Passano, E. Time and frequency domain analysis of catenary risers subjected to vortex induced vibrations. Hamburg, Germany, 2006. Proceedings of 25th international conference on offshore mechanics and arctic engineering,(OMAE).

[4] Larsen, C. M., Wu, J, and Lie, H. On the understanding of non-stationary viv of slender beams. Shanghai, 2010. In proceedings of the 29th International Conference on Ocean, Offshore and Arctic Engineering (OMAE).

[5] Yttervik, R. *Ocean current variability in relation to offshore engineering.* PhD thesis, NTNU, 2004.

[6] Larsen, C. M. VIV - A short and incomplete introduction to fundamental concepts, 2011.

[7] Langen, I. and Sigbjørnson, R. *Compendium in Dynamic Analysis of Constructions.*

[8] Gopalkrishnan, R. *Vortex-induced forces on oscillating bluff cylinders.* PhD thesis, Department of Ocean Engineering, MIT, Boston, 1993.

[9] Vikestad, K. *Multi-frequency response of a cylinder subjected to vortex shedding and support motions.* PhD thesis, Department of Marine Technology, NTNU, Trondheim, 1998.

[10] Larsen, C. M. and Passano, E. Time and frequency domain analysis of catenary risers subjected to vortex induced vibrations. Hamburg, Germany, June 2006. OMAE. Proceedings of the 29th international conference on ocean, offshore and arctic engineering.

[11] Hodder, M.S. and Byrne, B.W. 3d experiments investigating the interaction of a model scr with the seabed. *Applied Ocean Research*, 32(2):146 – 157, 2010.

[12] Katifeoglou, S. A. and Chatjigeorgiou, I. K. Dynamic interaction of catenary risers with the seafloor. *Applied Ocean Research*, 38:1 – 15, 2012.

[13] Jing, C. Fatigue analysis techniques of deepwater steel catenary risers. Shanghai, 2010. In proceedings of the 29th International Conference on Ocean, Offshore and Arctic Engineering (OMAE).

[14] Det Norske Veritas. *Recommended Practice C203 – Fatigue Design of Offshore Steel Structures*, October 2012.

[15] Det Norske Veritas. *Recommended Practice F204 – Riser Fatigue*, October 2010.

[16] Li, F. Z. and Low, Y. M. Fatigue reliability analysis of a steel catenary riser at the touchdown point incorporating soil model uncertainties. *Applied Ocean Research*, 38:100 – 110, 2012.

[17] *RIFLEX User's Manual.*

[18] *VIVANA User's Manual.*

[19] Barholm, G. S., Larsen, C. M., and Lie, H. On fatigue damage accumulation from in-line and cross-flow vortex-induced vibrations on risers. *Journal of Fluids and Structures*, (22):109–127, 2006.

[20] Faltinsen, O. M. *Sea Loads on Ships and Offshore Structures.* Cambridge University Press, 1990.

[21] Larsen, C. M, Lie, H., Passano, E., Yttervik, R., Wu, J., and Baarholm, G. Vivana theory manual. version 3.7, June 2009.

Appendix A
Data files

All necessary data files, scripts and functions have been uploaded to a zip-file on the web, except the data containing the vessel motion due to the large size (around 600 MB). The data files containing the vessel motion have been uploaded to a USB-disc that is given to the supervisor.

A.1 Input files

In this work, three type of input files have been used in RIFLEX and VIVANA.

sima_inpmod.inp Input file containing necessary data about the riser structure. Needed to run IMPMOD.

sima_stamod.inp Input file needed for the STAMOD calculation.

sima_vivana.inp Input file for VIVANA.

These files are not attached in the text, however, they have been uploaded to DAIM as a zip-file, and on an USB-disc given to the supervisor.

A.2 Variation text files

There was created two variations.txt files. One for the parameter variation and one for the vessel coordinate variation. These files contain all the variables that are passed to the input files by a Matlab-script called *change_str_in_file.m*. The text-files containing the variables used for the parameter variation study are attached such that the reader might look up the different variables when reading Chapter 7. The text-file containing the variables for the variation of vessel coordinate analysis is uploaded to DAIM and to the USB-disc.

A.2.1 variations.txt file for the parameter variation study

#RUNID	$ICURIN	$CURDIR	$STFBOT	$STFAXI	$STFLAT	$NELSEG	$CQX	$CQY	$RELDAM	$CHID	$RM1	$RC1	What:
run_1	1	90.0	300.0	200.0	200.0	1600	0.0315	1.05	0.01	DNV_T	3.0	11.764	Current
run_2	2	90.0	300.0	200.0	200.0	1600	0.0315	1.05	0.01	DNV_T	3.0	11.764	
run_3	3	90.0	300.0	200.0	200.0	1600	0.0315	1.05	0.01	DNV_T	3.0	11.764	
run_4	4	90.0	300.0	200.0	200.0	1600	0.0315	1.05	0.01	DNV_T	3.0	11.764	
run_5	3	0.0	300.0	200.0	200.0	1600	0.0315	1.05	0.01	DNV_T	3.0	11.764	Current heading
run_6	3	90.0	300.0	200.0	200.0	1600	0.0315	1.05	0.01	DNV_T	3.0	11.764	
run_7	3	270.0	300.0	200.0	200.0	1600	0.0315	1.05	0.01	DNV_T	3.0	11.764	
run_8	4	90.0	300.0	200.0	200.0	1600	0.0315	1.05	0.01	DNV_T	3.0	11.764	Bottom properties
run_9	4	90.0	250.0	150.0	150.0	1600	0.0315	1.05	0.01	DNV_T	3.0	11.764	
run_10	4	90.0	200.0	100.0	100.0	1600	0.0315	1.05	0.01	DNV_T	3.0	11.764	
run_11	4	90.0	150.0	50.0	50.0	1600	0.0315	1.05	0.01	DNV_T	3.0	11.764	
run_12	4	90.0	300.0	200.0	200.0	1600	0.03150	1.05000	0.01	DNV_T	3.0	11.764	Relative damping.
run_13	4	90.0	300.0	200.0	200.0	1600	0.03150	1.05000	0.01200	DNV_T	3.0	11.764	
run_14	4	90.0	300.0	200.0	200.0	1600	0.03150	1.05000	0.00800	DNV_T	3.0	11.764	
run_15	4	90.0	300.0	200.0	200.0	400	0.03150	1.05000	0.01000	DNV_T	3.0	11.764	No. of elements of second segment.
run_16	4	90.0	300.0	200.0	200.0	800	0.03150	1.05000	0.01000	DNV_T	3.0	11.764	
run_17	4	90.0	300.0	200.0	200.0	1600	0.03150	1.05000	0.01000	DNV_T	3.0	11.764	
run_18	4	90.0	300.0	200.0	200.0	3200	0.03150	1.05000	0.01000	DNV_T	3.0	11.764	
run_19	4	90.0	300.0	200.0	200.0	1600	0.0315	1.05	0.01	DNV_T	3.0	11.764	SN curve choice
run_20	4	90.0	300.0	200.0	200.0	1600	0.0315	1.05	0.01	DNV_B1	4.0	15.117	
run_21	4	90.0	300.0	200.0	200.0	1600	0.0315	1.05	0.01	DNV_C	3.0	12.592	
run_22	4	90.0	300.0	200.0	200.0	1600	0.0315	1.05	0.01	DNV_W3	3.0	10.97	

A.3 Files from the SIMO analysis

The analysis from the SIMO consisted of many folders, every with different weather conditions. Inside every folder there were two files needed for this work. They were

- **Xpos.asc:** x-coordinate for every time step.
- **Ypos.asc:** y-coordinate for every time step.

These files contained the vessel motion information. The total size of all files were around 600 MB, and they are only uploaded to the USB-disc.

A.4 BATCH-scripts

The BATCH-scripts were created to run RIFLEX and VIVANA with the opportunity to change different parameters. This was crucial in order to do many calculations with the computer programs. The BATCH-scripts were also used to call different Matlab-scripts that were designed to do special tasks necessary in the work. The two BATCH-scripts are attached in the thesis, so that the reader may see how a BATCH-script is constructed.

A.4.1 BATCH-script used for parameter variation study.

```
@ECHO Off
SETLOCAL ENABLEDELAYEDEXPANSION

rem ** PATHS
SET SIMDIR=%CD%
SET RIFLEX_HOME=C:\Users\lejlic\master_thesis\riflex_simo_academic_4.0.2\
    Riflex\bin
SET INP=%SIMDIR%\input
SET MATROOT=C:\programs\MATLAB\R2012a\bin
SET MAT_SCRIPTS=%SIMDIR%\mat

REM ** CONTROL WHICH RESULTS FILES THAT WILL BE DELETED
SET CLEAN=*.ffi *.sam *.gnd *.raf *.plt *.plo *.t *.ts *.key
SET CLEAN_MATLAB=*.m

rem ** CONTROL STAMOD AND DYNMOD MEMORY
SET RIFLEX_STAMOD_MEM=800
SET RIFLEX_DYNMOD_MEM=800
SET RIFLEX_MAXDYN_IFNDYN=800000

rem ** PROMPT SYSTEM IDENTIFIER
SET /P SYSID="Please enter system identifier (ID):"
SET /P CASES="Please enter case list file:"
SET RINP=%SYSID%_inpmod.inp
SET RSTA=%SYSID%_stamod.inp
SET RVIV=%SYSID%_vivana.inp

rem ** LOOP CASES
FOR /F "eol=# tokens=1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18" %%a IN (%CASES%) DO (

    rem ** Assign case variables
    SET RUNID=%%a
    SET ICURIN=%%b
    SET CURDIR=%%c
    SET STFBOT=%%d
    SET STFAXI=%%e
    SET STFLAT=%%f
    SET NELSEG=%%g
    SET CQX=%%h
    SET CQY=%%i
    SET RELDAM=%%j
    SET CHID=%%k
    SET RM1=%%l
    SET RC1=%%m

    rem ** Create result folders
    TITLE Analysis running - !RUNID!
    SET RESDIR=%SIMDIR%\results\%SYSID%\!RUNID!
    IF NOT EXIST !RESDIR! (MD !RESDIR!)
    CD !RESDIR!

    rem ** Copy master input files (remember that in your case *_inpmod.inp also contain the RAOs)
    COPY %INP%\%SYSID%_*.inp . /Y

    rem ** copying the matlab scripts to the result map in order to run the script from there.
    COPY %MAT_SCRIPTS%\*.m . /Y

    rem ** Inserting values for wildcards in inpmod.inp
    CALL %MATROOT%\matlab -nodesktop -nosplash -wait -nojvm -r "change_str_in_file !RESDIR!\%SYSID%_inpmod.inp $CURDIR !CURDIR! $STFBOT !STFBOT! $STFAXI !STFAXI! $STFLAT !STFLAT! $NELSEG !NELSEG! $CQX !CQX! $CQY !CQY!"

    rem ** Inserting values for wildcards in stamod.inp
    CALL %MATROOT%\matlab -nodesktop -nosplash -wait -nojvm -r "change_str_in_file !RESDIR!\%SYSID%_stamod.inp $ICURIN !ICURIN!"

    rem ** Inserting values for wildcards in vivana.inp
    CALL %MATROOT%\matlab -nodesktop -nosplash -wait -nojvm -r "change_str_in_file !RESDIR!\%SYSID%_vivana.inp $RELDAM !RELDAM! $CHID !CHID! $RM1 !RM1! $RC1 !RC1!"

    rem ** Riflex (INPMOD, STAMOD, VIVANA)
    FOR %%z IN ( i s v ) DO (CALL %RIFLEX_HOME%\riflex %%z %SYSID%)

    rem ** Clean
    DEL %CLEAN%

    rem ** Plotting VIVANA results from the mpf file:
    FOR %%z IN ( stamod vivana ) DO (CALL %MATROOT%\matlab -nodesktop -wait
       -nosplash -nojvm -r "postprocess_mpf %SYSID%_%%z.mpf"  )

    DEL %CLEAN_MATLAB%

    CD %SIMDIR%
  )

TITLE %CASES% COMPLETE
PAUSE
```

A.4.2 BATCH-script used for variation of vessel coordinate analysis

```
@ECHO Off
SETLOCAL ENABLEDELAYEDEXPANSION

rem ** PATHS
SET SIMDIR=%CD%
SET RIFLEX_HOME=C:\Users\lejlic\master_thesis\riflex_simo_academic_4.0.2\
   Riflex\bin
SET INP=%SIMDIR%\input
SET MATROOT=C:\programs\MATLAB\R2012a\bin
SET MAT_SCRIPTS=%SIMDIR%\mat


REM ** CONTROL WHICH RESULTS FILES THAT WILL BE DELETED
SET CLEAN=*.ffi *.sam *.gnd *.raf *.plt *.plo *.t *.ts *.key
SET CLEAN_MATLAB=*.m

rem ** CONTROL STAMOD AND DYNMOD MEMORY
SET RIFLEX_STAMOD_MEM=800
SET RIFLEX_DYNMOD_MEM=800
SET RIFLEX_MAXDYN_IFNDYN=800000

rem ** PROMPT SYSTEM IDENTIFIER
SET /P SYSID="Please enter system identifier (ID):"
SET /P CASES="Please enter case list file:"
SET RINP=%SYSID%_inpmod.inp
SET RSTA=%SYSID%_stamod.inp
SET RVIV=%SYSID%_vivana.inp
SET FATDAM=fatigue_damage.txt

rem ** LOOP CASES
FOR /F "eol=# tokens=1,2,3,4,5,6,7,8" %%a IN (%CASES%) DO (

    rem ** Assign case variables
    SET RUNID=%%a
    SET CURDIR=%%b
    SET XPOS=%%c
    SET YPOS=%%d
    SET PROB=%%e
    SET XNAME=%%f
    SET YNAME=%%g
    SET PROBNAME=%%h

    rem ** Create result folders
    TITLE Analysis running - !RUNID!
    SET RESDIR=%SIMDIR%\results\%SYSID%\!RUNID!_xpos!XNAME!ypos!YNAME!prob!PROBNAME!
    IF NOT EXIST !RESDIR! (MD !RESDIR!)
    CD !RESDIR!

    rem ** Copy master input files (remember that in your case *_inpmod.inp also contain the RAOs)
    COPY %INP%\%SYSID%_*.inp . /Y

    rem ** copying the matlab scripts to the result map in order to run the script from there.
    COPY %MAT_SCRIPTS%\*.m . /Y

    rem ** Inserting values for wildcards in inpmod.inp
    CALL %MATROOT%\matlab -nodesktop -nosplash -wait -nojvm -r "change_str_in_file !RESDIR!\%SYSID%_inpmod.inp $CURDIR !CURDIR! $XPOS !XPOS! $YPOS !YPOS!"

    rem ** Riflex (INPMOD, STAMOD, VIVANA)
    FOR %%z IN ( i s v ) DO (CALL %RIFLEX_HOME%\riflex %%z %SYSID%)

    rem ** Clean
    DEL %CLEAN%

    rem ** Plotting VIVANA results from the mpf file:
    rem ** FOR %%z IN ( stamod vivana ) DO (CALL %MATROOT%\matlab -nodesktop -wait -nosplash -nojvm -r "postprocess_mpf %SYSID%_%%z.mpf"  )

    rem ** Extracting the fatigue damage from the .mpf file and deletning and creating
    rem ** a new .txt file containing the fatigue results in order to take up less space.
    CALL %MATROOT%\matlab -nodesktop -nosplash -wait -nojvm -r "extract_fatigue_from_mpf %SYSID%_vivana.mpf %FATDAM%"

    DEL %CLEAN_MATLAB%
    rem ** Deleting the .mpf files which take up a lot of space...
    DEL *.mpf

    CD %SIMDIR%
  )

TITLE %CASES% COMPLETE
PAUSE
```

A.5 Matlab files

During the semester many scripts have been made in Matlab in order to do necessary calculations. The most important scripts and functions have been attached together with the thesis. This may give the reader a better understanding how the different functions and scripts are designed for this work. All scripts and functions have also been uploaded to DAIM and to the USB-disc.

A.5.1 Run file for processing of the motion file from SIMO, process_x_y_pos.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%process_x_y_pos.m
%
%This script gets information about a vessels (x,y) position and counts the
%number of occurences. By doing this, it is possible to get the probability
%of the vessel's postion, p(x,y). This probability will then be used in
%fatigue analysis of a SCR, to see what the effect on the fatigue at the
%TDP.
%
%
%Name: Emir Lejlic
%Date: 16.04.13
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Reading the input-files:

%NB: It is very important to have the folder containing all the x and y
%info in the same directory as this script!

%The input files
x_y_folder = 'X_Y_pos';
x_pos_file = 'Xpos.asc';
y_pos_file = 'Ypos.asc';

%The columns 3-4 are the transformed coord, while columns 5-6 are the
%transformed are mulitplied in order to reference them into a matrix.
x_y_motion_cols = 6;

%Saving the input into a struct called x_y_motion.

%x_data_rows and x_pos_files give information on how many files there are
%and how large they are. By knowing that it will be easy to loop through
%them.
x_y_motion = read_xy_data(x_y_folder, x_pos_file, y_pos_file, x_y_motion_cols)
    ;

%% Finding the max and min values.
[x_pos_files, ~] = size(x_y_motion);
```

```matlab

[x_data_rows, ~] = size(x_y_motion(1,1).x_y_data);

%By knowing the largest values, it will give a good indicator where to
%place my transformed coord-system:

%Want to get the max/min values from col 1 and 2;
index = 1;

[x_max, x_min, y_max, y_min] = ...
 find_extreme_coord(x_y_motion, x_pos_files, index);

fprintf('Max/min values:\n');
fprintf('x_max: %4.2f and x_min: %4.2f\n',x_max,x_min);
fprintf('y_max: %4.2f and y_min: %4.2f\n',y_max,y_min);

fprintf('\nCeiled max and floored min values:\n');
fprintf('x_max: %4.2f and x_min: %4.2f\n',ceil(x_max),floor(x_min));
fprintf('y_max: %4.2f and y_min: %4.2f\n',ceil(y_max),floor(y_min));

%Saving the original max/min into variables:
x_max_orig = x_max;
x_min_orig = x_min;
y_max_orig = y_max;
y_min_orig = y_min;

%Ceiling and flooring the max/min values!
x_max = ceil(x_max);
x_min = floor(x_min);
y_max = ceil(y_max);
y_min = floor(y_min);

%Saving the ceiled/floored max/min values into an array.
rounded_orig_max_values = [x_min x_max y_min y_max];

% Running this function to transform the data.
[x_y_motion] = transform_xy_data(x_y_motion, x_pos_files, x_data_rows,...
    x_min, y_max);

%Want to get the max/min values from the transformed data in col. 3 and 4.
index = 3;
[x_max_trans, x_min_trans, y_max_trans, y_min_trans] = ...
 find_extreme_coord(x_y_motion, x_pos_files, index);

fprintf('\nTransformed max/min values:\n');
fprintf('x_max: %4.2f and x_min: %4.2f\n',x_max_trans,x_min_trans);
fprintf('y_max: %4.2f and y_min: %4.2f\n',y_max_trans,y_min_trans);
fprintf('\nx_min is 0.0 as default. This means that the');
fprintf(' find_extreme_coord function\nwill use this if a lower value is not found!\n');

%The largest values from the transformed
index = 5;
[x_max_trans_100, x_min_trans_100, y_max_trans_100, y_min_trans_100] = ...
```

```
 find_extreme_coord(x_y_motion, x_pos_files, index);

%Ceiling and flooring the max/min values!
x_max_trans = ceil(x_max_trans_100);
x_min_trans = floor(x_min_trans_100);
y_max_trans = ceil(y_max_trans_100);
y_min_trans = floor(y_min_trans_100);

fprintf('\nCeiled and floored transformed*100 max/min values:\n');
fprintf('x_max_100: %4.2f and x_min_100: %4.2f\n',...
    x_max_trans_100,x_min_trans_100);
fprintf('y_max_100: %4.2f and y_min_100: %4.2f\n',...
    y_max_trans_100,y_min_trans_100);

% Creating the big coord matrix and counting the occurences of the motion.

[counted_x_y_data, large_transformed_coord_matrix, total_count] = ...
    count_in_large_matrix(x_y_motion, x_data_rows,...
    x_pos_files, y_max_trans, x_max_trans, x_min, y_max);

%% Inserting the occurences and probability of all x and y coordinates.

xy_coord_prob = xy_count_and_prob(counted_x_y_data, ...
    large_transformed_coord_matrix, total_count, x_min, y_max);

%Checking if all possibilities are in the calc:
clc;
sum_prob = sum(xy_coord_prob(:,4));
sum_count = sum(xy_coord_prob(:,3));

fprintf('\nTotal p(x,y): %4.1f %%.\n',sum_prob*100);
fprintf('Total count: %i occurences.\n',sum_count);

%% Converting the coordinates to polar coordinates.

xy_polar_cord_prob = convert_xy_to_polar_xy(xy_coord_prob);
%% Counting the position, number of occurences and probability to discrete points

clc;
%The largest radius creates the boundary.
largest_radius = ceil(max(xy_polar_cord_prob(:,3)));
%The angle size of each sector.
sector_degree = 30;
%How many parts each sector has.
sector_parts = 12;

xy_counted_in_sectors = ...
search_xy_in_sectors(xy_polar_cord_prob,sector_degree,largest_radius, sector_parts);

%Using the discrete scattered data to create a function Z which will create a surface plot.
%The surface plot is just to have a better picture of how the points are
```

```
%connected.
x_min = min(xy_counted_in_sectors(:,1));
x_max = max(xy_counted_in_sectors(:,1));

y_min = min(xy_counted_in_sectors(:,2));
y_max = max(xy_counted_in_sectors(:,2));

%The largest probability.
z_max = max(xy_counted_in_sectors(:,7));

%Creating this for the meshgrid.
xlin = linspace(x_min,x_max,100);
ylin = linspace(y_min,y_max,100);

%Making the grid.
[X,Y] = meshgrid(xlin,ylin);

%The important parameters from the xy_counted_in_sectors.
x = xy_counted_in_sectors(:,1);
y = xy_counted_in_sectors(:,2);
prob = xy_counted_in_sectors(:,7);

%Interpolating the probability at the discrete points.
f = scatteredInterpolant(x,y,prob,'linear','none');

%Z is now the surface function.
Z = f(X,Y);

%% Creating the variations file:

%Creating a variations file for the (x,y) pos of the vessel. Only the (x,y)
%points with p(x,y) ~= 0.0 are taken into consideration.
counter = 0;
counter_2 = 0;
counter_3 = 0;

prob_limit = 0.00009;

fid = fopen('variations.txt','w');

fprintf(fid,'%s %s %s %s\n','#Id','$x_pos','$y_pos','$prob');

for i = 1:size(xy_counted_in_sectors,1)

    if (xy_counted_in_sectors(i,7) >= prob_limit)

        fprintf(fid,'%s %3.1f %3.1f %e\n',strcat('Run_',num2str(counter)),...
                xy_counted_in_sectors(i,1),xy_counted_in_sectors(i,2),...
                xy_counted_in_sectors(i,7));

        counter = counter + 1;

    elseif (xy_counted_in_sectors(i,7) < prob_limit) && (xy_counted_in_sectors
        (i,7) > 0)

        %Counting between 0 and prob_limit
        counter_2 = counter_2 + 1;

    else
        %Counting which have 0 prob.
        counter_3 = counter_3 + 1;

    end

end

fclose(fid);

num_points_considered = counter;

points_into_consid = zeros(num_points_considered,3);
points_not_into_consid = zeros(counter_2,3);
points_with_zero_prob = zeros(counter_3,3);
```

A.5.2 read_xy_data.m

```
function [x_y_motion, x_data_rows, x_pos_files] =...
    read_xy_data(x_y_folder,x_pos_file,y_pos_file,x_y_motion_cols)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This matlab script is supposed to read through the Xpos and Ypos files,
%and save them in a struct for further processing.
%
%Name: Emir Lejlic
%Date: 09.04.13
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Reading the data:

%Going inside the x_y_folder
cd(x_y_folder);
%Saving the dir info where the x_y folders are
X_Y_directory = dir;
%Going out of the folder
cd ..

%Need the size to loop through the folders!
[X_Y_dir_row, ~] = size(X_Y_directory);

%—2 because of . and .. in the directory.
x_pos_files = X_Y_dir_row — 2;

%Unecessary variable:
clear X_Y_dir_col;

%Creating a struct containing all necessary info. The —2 is bebause of .
%and .. in the directory.
```

```
x_y_motion(X_Y_dir_row-2,1) = struct();

%Creating paths were the x_y_pos files are:
    for i = 3:X_Y_dir_row

        x_y_motion(i-2,1).folder = fullfile(cd,x_y_folder,X_Y_directory(i,1).name);

    end
fprintf('\nReading the .asc files\n');
% Creating data:

    for i = 1:x_pos_files

        %The path to the .asc files
        x_file = fullfile(x_y_motion(i,1).folder,x_pos_file);
        y_file = fullfile(x_y_motion(i,1).folder,y_pos_file);
        %Importing them into variables as a string.
        x_text = file2str(x_file);
        y_text = file2str(y_file);

        %The last hashtag indicates when the data starts.
        hashtags = strfind(x_text,'#');

        last_hashtag = hashtags(1,length(hashtags));

        x_str_data = x_text(last_hashtag+1:end);
        y_str_data = y_text(last_hashtag+1:end);

        x_data = str2num(x_str_data);
        y_data = str2num(y_str_data);

        [x_data_rows, ~] = size(x_data);

        clear x_data_cols;

        %Preallocating the x_y_data matrixes:
        x_y_motion(i,1).x_y_data = zeros(x_data_rows,x_y_motion_cols);

        %Inserting the x and y data:
        x_y_motion(i,1).x_y_data(:,1) = x_data(:,2);
        x_y_motion(i,1).x_y_data(:,2) = y_data(:,2);

        fprintf('%i of %i files scanned.\n',i,x_pos_files);


        %Deleting unecessary variables:
        clear x_file y_file x_text y_text x_str_data y_str_data;

    end

end
```

A.5.3 find_extreme_coord.m

```
function [x_max, x_min, y_max, y_min] = find_extreme_coord(x_y_motion, x_pos_files, index)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Find the largest extreme coordinates from the data.
%
%Name: Emir Lejlic.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Need to set a start value!
x_max = 0;
x_min = x_max;
y_max = x_max;
y_min = x_max;

%Going through the x and y pos files and finding the largest values:
    for i = 1:x_pos_files

        %The temp values are the max from each file, while x_max is the largest
        %of all x_max_tmp.

        %The index refers to which x—value you want to get. Index + 1 will
        %find the y—values.
        x_max_tmp = max(x_y_motion(i,1).x_y_data(:,index));

        if x_max_tmp >= x_max
            x_max = x_max_tmp;
        end

        x_min_tmp = min(x_y_motion(i,1).x_y_data(:,index));

        if x_min_tmp <= x_min
            x_min = x_min_tmp;
        end

        y_max_tmp = max(x_y_motion(i,1).x_y_data(:,index+1));

        if y_max_tmp >= y_max
            y_max = y_max_tmp;
        end

        y_min_tmp = min(x_y_motion(i,1).x_y_data(:,index+1));

        if y_min_tmp <= y_min
            y_min = y_min_tmp;
        end

    end

end
```

A.5.4 transform_xy_data.m

```
function [x_y_motion] = ...
transform_xy_data(x_y_motion, x_pos_files, x_data_rows, x_min, y_max)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%The function takes in the original (x,y) data, and transforms them according to
%the new transformed coordinate system.
%
%Created by: Emir Lejlic
%Date:       18.04.13
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%These values are the origin of the transformed coord. system.
x_start = x_min;
y_start = y_max;

fprintf('\nTransforming the (x,y) data...\n');

    for i = 1:x_pos_files

        fprintf('%i of %i files transformed.\n',i,x_pos_files);

        for j = 1:x_data_rows

            %The transformed x-coord:
            x_y_motion(i,1).x_y_data(j,3) = round(10* (x_y_motion(i,1).x_y_data(j,1) - x_start))/10;
            %The transformed y-coord:
            x_y_motion(i,1).x_y_data(j,4) = round(10* (y_start - x_y_motion(i,1).x_y_data(j,2)))/10;

            %Multiplying the transformed data with 100 to be able to place
            %them in a matrix.
            x_y_motion(i,1).x_y_data(j,5) = round(x_y_motion(i,1).x_y_data(j,3) * 100);
            x_y_motion(i,1).x_y_data(j,6) = round(x_y_motion(i,1).x_y_data(j,4) * 100);

        end

    end

end
```

A.5.5 count_in_large_matrix.m

```
function [counted_x_y_data, large_transformed_coord_matrix, total_count] ...
    = count_in_large_matrix(x_y_motion, x_data_rows, x_pos_files,...
                            y_max_trans, x_max_trans, x_min, y_max)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function goes through all the transformed, multiplied coordinates,
%and inserts the number of occurences into a large matrix.
%
%Date: 23.04.13
%Name: Emir Lejlic.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Since the coord. system is transformed, the y_max will give how many rows
%that will be necessary, and how many

%These values are needed to transform back to the original coordinates:
x_start = x_min;
y_start = y_max;

large_transformed_coord_matrix = zeros(y_max_trans, x_max_trans);

%Creating a 100000 x 2 matrix to store the results:
%| x | y |

%The matrix has to be large enough to store all the results. To store the
%number of counts, the total_counter is used.
counted_x_y_data = zeros(100000,2);

%The total count will give the final count of all occurences.
total_count = 0;

%This counter will help store the results in the counted_x_y_data matrix.
counter = 1;

fprintf('\nInserting the occurences into the large matrix:\n');

    for i = 1:x_pos_files

        fprintf('%i out of %i files finished.\n',i,x_pos_files);

        for j = 1:x_data_rows

            total_count = total_count + 1;

            %The transformed x-value corresponds to the m-index in the large
            %matrix. The same is with the n-index and the transformed y-value.
            m = x_y_motion(i,1).x_y_data(j,5);
            n = x_y_motion(i,1).x_y_data(j,6);

            large_transformed_coord_matrix(n,m) =
               large_transformed_coord_matrix(n,m) + 1;

               %Counting which coordinates are being registered.
               if large_transformed_coord_matrix(n,m) == 1

                  %Dividing by 100 to get the transformed coords.
                   x_transformed = m/100;
                   y_transformed = n/100;

                   %Transforming the coord back to the original ones.
                   x = x_transformed + x_start;
                   y = y_start - y_transformed;

                   counted_x_y_data(counter,1) = x;
                   counted_x_y_data(counter,2) = y;

                   counter = counter + 1;

               end
         end
    end

    %Deleting the unecessary empty cells:
    counted_x_y_data(counter:end,:) = [];

end
```

A.5.6 xy_count_and_prob.m

```
function xy_coord_prob = ...
xy_count_and_prob(counted_x_y_data, large_transformed_coord_matrix,...
total_count, x_min, y_max)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%The function goes through all the coordinates in the counted_x_y_data
%matrix and finds the number of occurences in the
%large_transformed_coord_matrix. By knowing the total number of occurences
%it is possible to find the probability of the motion p(x,y).
%
%Date: 24.04.13
%Name: Emir Lejlic
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%These are needed to convert to the transformed coordinates.
x_start = x_min;
y_start = y_max;

%Needed to loop through the matrices.
[rows, ~] = size(counted_x_y_data);

%The cols 1 and 2 will have the x and y coordinate, while cols 3 and 4 will
%have the number of occurences and probability.
xy_coord_prob = zeros(rows, 4);

    for i = 1:rows

        x = counted_x_y_data(i,1);
        y = counted_x_y_data(i,2);

        xy_coord_prob(i,1) = x;
        xy_coord_prob(i,2) = y;

        y_transformed = y_start - y;
        x_transformed = x - x_start;

        m = round(x_transformed * 100);
        n = round(y_transformed * 100);

        counted_occurences = round(large_transformed_coord_matrix(n,m));

        %The count of each respective x and y coordinate.
        xy_coord_prob(i,3) = counted_occurences;

        %The probability of each the respective x and y coordinate.
        xy_coord_prob(i,4) = counted_occurences/total_count;


    end



end
```

A.5.7 convert_xy_to_polar_xy.m

```
function xy_polar_cord_prob = convert_xy_to_polar_xy(xy_coord_prob)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function takes the coordinates, their counts and prob, and
%converts the coordinates to polar coordinates, for further processing.
%
%Date: 25.04.13
%Name: Emir Lejlic.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Creating the matrix containing the coordinates. It has to have:
%
%  | x | y | radius | theta_rad | theta_deg | counts | prob. |
%
%It means that it should be a rows(xy_coord_prob) x 7

[rows, ~] = size(xy_coord_prob);

xy_polar_cord_prob = zeros(rows, 7);

    fprintf('Converting to polar coordinates...\n');

    %Inserting the values.
    for i = 1:rows

        fprintf('%i of %i coordinates converted.\n',i,rows);

        %Inserting x values:
        xy_polar_cord_prob(i,1) = xy_coord_prob(i,1);
```

```matlab
        %Inserting y values:
        xy_polar_cord_prob(i,2) = xy_coord_prob(i,2);
        %Calculating the radius:
        xy_polar_cord_prob(i,3) = sqrt(xy_coord_prob(i,1).^2 + xy_coord_prob(i,2).^2);

        %Finding the theta angle in radians:
        xy_polar_cord_prob(i,4) = atan2(xy_polar_cord_prob(i,2), xy_polar_cord_prob(i,1));

            %If the coordinate is in the 3. or 4. quadrant, this is
            %necessary.
            if xy_polar_cord_prob(i,4) < 0
                xy_polar_cord_prob(i,4) = xy_polar_cord_prob(i,4) + (2*pi);
            end

        %Theta angle in degrees:
        xy_polar_cord_prob(i,5) = xy_polar_cord_prob(i,4) .* (180/pi);


        %Number of counts:
        xy_polar_cord_prob(i,6) = xy_coord_prob(i,3);
        %The probabilit:
        xy_polar_cord_prob(i,7) = xy_coord_prob(i,4);

    end

end
```

A.5.8 search_xy_in_sectors.m

```matlab
function xy_counted_in_sectors = ...
search_xy_in_sectors(xy_polar_cord_prob,sector_degree,largest_coord_pos, sector_parts)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function will go through the sectors and discrete points defined in
%the input, and count the occurences of the (x,y) postion of the vessel.
%
%Date: 28.04.13
%Name: Emir Lejlic.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Number of sectors.
no_sectors = 360/sector_degree;

%These theta (angle) values are used in the for loop.
theta =(sector_degree/2):sector_degree:(360-sector_degree/2);

%Sector parts define how many intervals there is inside a sector. This will
%create the borders which will be borders for the counting of the
%positions.
delta_radius = (largest_coord_pos/sector_parts);
```

```
sector_radius = 0:delta_radius:largest_coord_pos;

%Finding the discrete points and creating a matrix for it.
discrete_points = create_discrete_point(sector_degree,largest_coord_pos,
   sector_parts);

%Creating the matrix for the final results.
%| x | y | r | sector_start | sector_end | counts | prob |
xy_counted_in_sectors = zeros((no_sectors * (sector_parts-1)) + 1,7);

%Starting the counter for the while loops.
i = 1;
j = 1;

%Need this counter to jump to new sector coords.
counter = 1;

%Used to check when xy_rows > 1.
xy_counter = 1;

    %Setting the points inside to sector_parts.
    while j <= size(xy_counted_in_sectors,1)

        if i == 1

            xy_counted_in_sectors(j,1) = discrete_points(i,1);
            xy_counted_in_sectors(j,2) = discrete_points(i,2);
            %Inserting the radius.
            xy_counted_in_sectors(j,3) = discrete_points(i+1,3);
            %Inserting the degrees:
            xy_counted_in_sectors(j,4) = discrete_points(i,4);
            xy_counted_in_sectors(j,5) = discrete_points(i,5);

            i = i + 1;
            j = j + 1;

        elseif (i > 1) && (i < size(discrete_points,1)-1)

            xy_counted_in_sectors(j,1) = (discrete_points(i,1) +
               discrete_points(i+1,1))/2;
            xy_counted_in_sectors(j,2) = (discrete_points(i,2) +
               discrete_points(i+1,2))/2;
            xy_counted_in_sectors(j,3) =  discrete_points(i+1,3);

            %Inserting the degrees:
            xy_counted_in_sectors(j,4) = discrete_points(i,4);
            xy_counted_in_sectors(j,5) = discrete_points(i,5);

            i = i + 1;
            j = j + 1;

            counter = counter + 1;

                %This is needed when all the points are checked in one
                %sector.
                if counter == length(sector_radius)-1

                    %Extra jump.
                    i = i + 1;
                    %Resetting the counter.
                    counter = 1;

                end

        else

            xy_counted_in_sectors(j,1) = (discrete_points(i,1) + discrete_points(i+1,1))/2;
            xy_counted_in_sectors(j,2) = (discrete_points(i,2) + discrete_points(i+1,2))/2;
            xy_counted_in_sectors(j,3) =  discrete_points(i+1,3);

            %Inserting the degrees:
            xy_counted_in_sectors(j,4) = discrete_points(i,4);
            xy_counted_in_sectors(j,5) = discrete_points(i,5);

            i = i + 1;
            j = j + 1;
        end

    end

clear i j;

    fprintf('Looping through the sectors...\n');
    %This for loop will find the probability in the discrete points.
    for i = 1:no_sectors

         fprintf('%i of %i sectors checked.\n',i,no_sectors);

         theta_end = theta(i);
         theta_start = theta_end - sector_degree;
         theta_tmp = theta_start;

         theta_discrete_deg = (theta_start + theta_end)/2;
         theta_discrete_rad = theta_discrete_deg *(pi/180);

         %Looping through the borders
         for j = 1:length(sector_radius)-1

             %Need to restart this after each j-iteration. Look at line
             %167.
             theta_start = theta_end - sector_degree;

             %The local borders in each sector.
             x_loc_start = sector_radius(j);
             x_loc_end = sector_radius(j+1);

```

```
%The local x distance in each sector.
x_local = sqrt((xy_polar_cord_prob(:,1).^2 + xy_polar_cord_prob(:,2).^2))...
    .* cos(abs(xy_polar_cord_prob(:,4)-theta_discrete_rad));

   %To account for negative values in the find_search below.
   if theta_tmp < 0

       %Saving the column with the info about the degree to
       %the coordinate.
       xy_dir_row_temp = xy_polar_cord_prob(:,5);

       %Looping through the degree column to change the
       %largest degrees. This is to make the find function to
       %work
       for k = 1:size(xy_polar_cord_prob,1)

           if xy_polar_cord_prob(k,5) > theta(end)

               xy_polar_cord_prob(k,5) = xy_polar_cord_prob(k,5) - 360;

           end

       end

       %Rows that match the criteria.
       res_rows = find(xy_polar_cord_prob(:,5) >= theta_start ...
                   & xy_polar_cord_prob(:,5) < theta_end ...
                   & x_local(:) <= x_loc_end ...
                   & x_local(:) > x_loc_start);

       %Inserting the original values back again.
       xy_polar_cord_prob(:,5) = xy_dir_row_temp;

   else

       %Rows that match the criteria.
       res_rows = find(xy_polar_cord_prob(:,5) >= theta_start ...
                   & xy_polar_cord_prob(:,5) < theta_end ...
                   & x_local(:) <= x_loc_end ...
                   & x_local(:) > x_loc_start);
   end



   if ~isempty(res_rows)

       %To account for negative values in the find_search below.
       if theta_tmp < 0
           theta_start = 360 - theta_end;
       end


```

```
%Looking if it is close to the origin.
if x_loc_end <= sector_radius(2)

   xy_row = 1;

%If the point is at any other place.
else

    %This if statement is important for finding the
    %exact row. Small numerical differences can be
    %crucial...
    if x_loc_end == sector_radius(end)

        %The delta_radius/20 will help the
        %FIND—function to find only one xy_row.
        xy_row = find(xy_counted_in_sectors(:,3) > (x_loc_start + delta_radius/20) ...
                & xy_counted_in_sectors(:,4) == theta_start ...
                & xy_counted_in_sectors(:,5) == theta_end);

                if_sentece = 1;


    else
        xy_row = find(xy_counted_in_sectors(:,3) > (x_loc_start + delta_radius/20) ...
                & xy_counted_in_sectors(:,3) < (x_loc_end + delta_radius/20)   ...
                & xy_counted_in_sectors(:,4) == theta_start ...
                & xy_counted_in_sectors(:,5) == theta_end);

                if_sentece = 2;

    end

    %This is useful for checking where the error is if
    %the FIND—function finds more than one xy_row.
    if length(xy_row) > 1

        fprintf('\nTime no: %i\n',xy_counter);
        fprintf('Rows found: %i\n',xy_row');
        fprintf('Sector no: %i\n',i);
        fprintf('Radius no: %i\n',j);
        fprintf('If_sentence no: %i\n',if_sentece);
        fprintf('theta_start: %f\n',theta_start);
        fprintf('theta_end: %f\n',theta_end);
        fprintf('x_loc_start: %f\n',x_loc_start);
        fprintf('x_loc_end + delta_radius/2: %f\n',(x_loc_start + delta_radius/20));
        fprintf('x_loc_end: %f\n',x_loc_end);
```

```
                             fprintf('x_loc_end + delta_radius/2: %f\n',(x_loc_end + delta_radius/20));

                             xy_counter = xy_counter + 1;

                         end

                     end

                     %This if-sentence shows where the error is the xy_row
                     %is empty.
                     if isempty(xy_row)

                         fprintf('\nx: %f\n',xy_polar_cord_prob(res_rows,1));
                         fprintf('y: %f\n',xy_polar_cord_prob(res_rows,2));
                         fprintf('x_loc_start: %f\n',x_loc_start);
                         fprintf('x_loc_end: %f\n',x_loc_end);
                         fprintf('theta_start: %f\n',theta_start);
                         fprintf('theta_end: %f\n\n',theta_end);

                     end

                     %The for-loop inserts the counts and the prob. at the
                     %discrete points.
                     for k = 1:length(res_rows)

                        %The counts.
                        xy_counted_in_sectors(xy_row,6) = xy_counted_in_sectors(xy_row,6) + xy_polar_cord_prob(res_rows(k),6);

                        %The probability.
                        xy_counted_in_sectors(xy_row,7) = xy_counted_in_sectors(xy_row,7) + xy_polar_cord_prob(res_rows(k),7);


                     end

                end

        end

    end

%Some key numbers.
fprintf('\nsum_cord_count = %i\n',sum(xy_polar_cord_prob(:,6)));
fprintf('sum_xy_sector_count = %i\n',sum(xy_counted_in_sectors(:,6)));
fprintf('Diff in count: %i\n',abs(sum(xy_counted_in_sectors(:,6)) - sum(xy_polar_cord_prob(:,6))));
fprintf('Sum_orig_prob = %f\n',sum(xy_polar_cord_prob(:,7)));
fprintf('Sum_counted_prob = %f\n',sum(xy_counted_in_sectors(:,7)));
fprintf('Diff in prob: %f\n',abs(sum(xy_counted_in_sectors(:,7)) - sum(xy_polar_cord_prob(:,7))));
```

```

end
```

A.5.9 Run file for summing the fatigue damages at different coordinates, run_variation.m

```
%% Run sum_fatigue_damage() function...
clc;
clear all;

run_folder = 'C:\Users\Emir Lejlic\Documents\MATLAB\master_thesis\fatigue_damage_xy_variation\variation_coord_result_folders';

%Creating a struct with the important info:
run_data = save_run_xy_info_in_struct(run_folder);

%% Adding the fatigue damage to the run_data struct.
fatigue_file = 'fatigue_damage.txt';
run_data = add_fatigue_damage_to_struct(run_data, run_folder,fatigue_file);


%% Summing the fatigue

[final_fat_dam, sum_prob] = sum_fatigue_damage_from_xy_variation(run_data);

fprintf('Summed p(x,y): %6.3f %%.\n',sum_prob *100);

%% Some key values...
clc;
fat_max_origin = max(run_data(1,1).fatigue_damage(:,2));
indx_origin = find(run_data(1,1).fatigue_damage(:,2) == fat_max_origin);
fat_max_variation = max(final_fat_dam(:,2));
indx_variation = find(final_fat_dam(:,2) == fat_max_variation);

format shortg
x_max_origin = run_data(1,1).fatigue_damage(indx_origin,1);
x_max_variation = final_fat_dam(indx_variation,1);
format short

%This is the difference in x values...
dx = abs(x_max_origin-x_max_variation);

%If the difference is small, I set dx = 3. Look at the xlim in the figures.
    if dx < 3

        dx = 3;

    end

%% Finding the largest probable position
clc;
prob_values = zeros(length(run_data),1);
```

```

for i = 1:length(run_data)

    prob_values(i,1) = run_data(i,1).prob;

end

max_prob_indx = find(prob_values == max(prob_values));
run_data(max_prob_indx,1).axial_force(1:5,2)

%% Creating a matric with coords, axial force at RHoP and prob.
clc;
%Finding the max and min values:

%Preallocating the memory
vessel_coords = zeros(length(run_data),4);

for i = 1:length(run_data)
    vessel_coords(i,1) = run_data(i,1).xpos;
    vessel_coords(i,2) = run_data(i,1).ypos;
    vessel_coords(i,3) = run_data(i,1).axial_force(1,2);
    vessel_coords(i,4) = run_data(i,1).prob;
end

max_tensile = max(vessel_coords(:,3));
min_tensile = min(vessel_coords(:,3));

%% Statistics of x-values

%Checking the probability for the vessel behind at x = 0, x > 0 and x < 0.
xprob_ahead_origin = 0;
xprob_behind_origin = 0;
xprob_origin = 0;

%Some valus may be really small, like 1.5e-14. That is larger than 0.
orig_lim = 0.1;

for i = 1:length(run_data)

    if vessel_coords(i,1) > orig_lim

        %Counting the prob for x > origin.
        xprob_ahead_origin = xprob_ahead_origin + vessel_coords(i,4);

    elseif vessel_coords(i,1) < -orig_lim

        %Counting the prob for x < origin.
        xprob_behind_origin = xprob_behind_origin + vessel_coords(i,4);

    else

        %Counting the prob. for x = origin.
        xprob_origin = xprob_origin + vessel_coords(i,4);

    end

end
clc;
fprintf('The probabilities are:\n');
fprintf('x-coord > 0: %3.2f\n',xprob_ahead_origin);
fprintf('x-coord = 0: %3.2f\n',xprob_origin);
fprintf('x-coord < 0: %3.2f\n\n',xprob_behind_origin);
fprintf('Sum prob: %3.2f\n',xprob_ahead_origin+xprob_behind_origin+
    xprob_origin);

%% Statistics of tensile forces.
%How many times are the tensile forces larger than at the origin.

%The tensile forces at the origin.
tensile_origin_rhop = vessel_coords(1,3);

%Finding if the probability is larger for greater tensile forces.
tensile_larger = 0;
tensile_origin = 0;
tensile_lower = 0;

for i = 1:length(run_data)

    %Tensile forces larger than in origin
    if vessel_coords(i,3) > tensile_origin_rhop

        tensile_larger = tensile_larger + vessel_coords(i,4);

    %Tensile forces lower than in origin.
    elseif vessel_coords(i,3) < tensile_origin_rhop

        tensile_lower = tensile_lower + vessel_coords(i,4);

    %The tensile forces are the same as the origin.
    else

        tensile_origin = tensile_origin + vessel_coords(i,4);

    end

end

fprintf('The probabilities are:\n');
fprintf('T_RHoP > T_orig: %3.2f\n',tensile_larger);
fprintf('T_RHoP < T_orig: %3.2f\n',tensile_lower);
fprintf('T_RHoP = T_orig: %3.2f\n\n',tensile_origin);
fprintf('Sum prob: %3.2f\n',tensile_larger+tensile_lower+tensile_origin);
%% Surface plot of the tensile forces

x_min = min(vessel_coords(:,1));
x_max = max(vessel_coords(:,1));

y_min = min(vessel_coords(:,2));
```

```
y_max = max(vessel_coords(:,2));

z_min = min(vessel_coords(:,3));
z_max = max(vessel_coords(:,3));

%Creating this for the meshgrid.
xlin = linspace(x_min,x_max,50);
ylin = linspace(y_min,y_max,50);

%Making the grid.
[X,Y] = meshgrid(xlin,ylin);

%The important parameters from the xy_counted_in_sectors.
x = vessel_coords(:,1);
y = vessel_coords(:,2);
tensile_forces = vessel_coords(:,3);

%Interpolating the probability at the discrete points.
f = scatteredInterpolant(x,y,tensile_forces,'natural','none');

%Z is now the surface function.
Z = f(X,Y);

tensile_forces = tensile_forces + 1;
%% Finding important parameters from the stamod.res file

%From the stamod.res file:
res_filename = 'sima_stamod.res';
run_data = find_important_parameters_resfile(run_data, run_folder,
    res_filename);

%% Finding the interval of the TDP
TDP_x_values = zeros(1,length(run_data));
Prob_values = zeros(1,length(run_data));
for i = 1:length(run_data)

    TDP_x_values(i) = run_data(i,1).TDP_x_value;
    Prob_values(i) = run_data(i,1).prob;

end

min_TDP = min(TDP_x_values);
max_TDP = max(TDP_x_values);

plot_TDP_y = 0.15;
plot_TDP = [min_TDP plot_TDP_y; max_TDP plot_TDP_y];

%% Comparing the axial force of the riser...
close all;

%The axial matrix has one row less than the static config matrix.
axial_rows = size(run_data(1,1).axial_force,1);

%Preallocating memory.
```

```
axial_force_origin = zeros(axial_rows,2);
axial_force_variation = zeros(axial_rows,2);
axial_force_variation_tmp = zeros(axial_rows,1);

%The line length [m]. OBS: 1:end-1 because of one less row!!
axial_force_variation(:,1) = run_data(1,1).fatigue_damage(1:end-1,1);
axial_force_origin(:,1) = axial_force_variation(:,1);

%Important that the first row is (0,0).
axial_force_origin(:,2) = run_data(1,1).axial_force(:,2);

%Adding together all the axial forces with their respective probabilities.
for i = 1:length(run_data)

   prob = run_data(i,1).prob;
   axial_force_variation_tmp = prob .* run_data(i,1).axial_force(:,2);

   axial_force_variation(:,2) = axial_force_variation(:,2) + axial_force_variation_tmp(:,1);

end

%Calculating the deviation between the axial forces.
deviation = zeros(1,axial_rows);

for i = 1:axial_rows
     deviation(i)  = abs(axial_force_origin(i,2)-axial_force_variation(i,2));
end

%Max deviation.
max_dev = max(deviation);
```

A.5.10 save_run_xy_info_in_struct.m

```
function run_data = save_run_xy_info_in_struct(run_folder)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%This function goes through all the folders containing the info of the
%(x,y) coordinate variation runs and saves it in a struct for further
%processing.
%
%Name: Emir Lejlic.
%Date: 08.05.13
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%These are needed to extract the run info needed.
list = dir(run_folder);

%Saving all the names
name = {list.name};
%Creating a string of the names, with a # between each folder.
str = sprintf('%s#', name{:});

```

```
%Finding the important strings to be able to sort the names.
find_run = strfind(str,'Run');
find_xpos = strfind(str,'xpos');

%Preallocating the for the numbers...
num = zeros(1,length(find_run));

    %Saving the run numbers
    for i = 1:length(find_run)
        num(i) = sscanf(str(find_run(i):find_xpos(i)-1), 'Run_%d');
    end

%This is the way to find the index of the numbers. The run is the sorted
%numbers.
[~, index] = sort(num);

%Creating a new struct which will only contain the run info.
run_data(size(list,1)-2,1) = struct();

    %Adding the sorted folder names to the struct.
    for i = 1:length(index)

        run_data(i,1).name = list(index(i)+2,1).name;

    end

    for i = 1:length(run_data)

    %Finding these strings in the name.
    str_xpos = strfind(run_data(i,1).name,'xpos');
    str_ypos = strfind(run_data(i,1).name,'ypos');
    str_prob = strfind(run_data(i,1).name,'prob');

    %Extracting the values.
    x_str = run_data(i,1).name(str_xpos+length('xpos'):str_ypos-1);
    y_str = run_data(i,1).name(str_ypos+length('ypos'):str_prob-1);
    prob_str = run_data(i,1).name(str_prob+length('prob'):end);

    %Replacing the underscores (_) with dots (.)(.)
    x_str = strrep(x_str,'_','.');
    y_str = strrep(y_str,'_','.');
    prob_str = strrep(prob_str,'_','.');

    %Now, converting the string values to numbers.
    run_data(i,1).xpos = str2double(x_str);
    run_data(i,1).ypos = str2double(y_str);
    run_data(i,1).prob = str2double(prob_str);

    end


end
```

A.5.11 add_fatigue_damage_to_struct.m

```
function run_data = add_fatigue_damage_to_struct(run_data, run_folder,
    fatigue_file)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%The function goes to the respective folder, and extracts the fatigue
%damage saved in the fatigue_damage.txt file. It then saves this info in
%the struct for further processing.
%
%Name: Emir Lejlic.
%Date: 08.05.13
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Creating a waitbar
h = waitbar(0,'Adding the fatigue damage...');

%Finding the fatigue damage.
    for i = 1:length(run_data)

        waitbar(i / length(run_data))

        %The folder where the fatigue.txt file is.
        fatigue_file_path = fullfile(run_folder,run_data(i,1).name,
            fatigue_file);
        %Going to the folder.

        %Reading the .txt file containing the damage.
        fatigue_text = file2str(fatigue_file_path);

        %[1/years] are the last chars before the numbers.
        find_str = strfind(fatigue_text,'[1/years]');

        run_data(i,1).fatigue_damage = str2num(fatigue_text(find_str+length('
            [1/years]')+1:end));

    end

    close(h)

end
```

A.5.12 sum_fatigue_damage_from_xy_variation.m

```
function [final_fat_dam, sum_prob] = sum_fatigue_damage_from_xy_variation(
    run_data)
%The function goes through the run_data struct and sums the fatigue damage.

%Preallocating the memory
final_fat_dam = zeros(size(run_data(1,1).fatigue_damage,1),2);
fat_dam_tmp = zeros(size(run_data(1,1).fatigue_damage,1),1);
```

```

%The arclength of the riser.
final_fat_dam(:,1) = run_data(1,1).fatigue_damage(:,1);

%Summing the probability to check if all are taken into account.
sum_prob = 0;


    for i = 1:length(run_data)

        %Adding the prob. each row in the struct.
        prob = run_data(i,1).prob;
        %Multiplying the fatigue for each (x,y) with its respective p(x,y).
        fat_dam_tmp(:,1) = prob .* run_data(i,1).fatigue_damage(:,2);
        %Adding it to the final fatigue.
        final_fat_dam(:,2) = final_fat_dam(:,2) + fat_dam_tmp(:,1);

        %Summing the probability.
        sum_prob = sum_prob + prob;

    end


end
```

A.5.13 find_important_parameters_resfile.m

```
function run_data = ...
    find_important_parameters_resfile(run_data, run_folder, res_filename)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%THIS FUNCTION FINDS IMPORTANT PARAMETERS FROM THE STAMOD.RES FILE. THE
%IMPORTANT PARAMTERS ARE:
%
%   1. TOUCH DOWN POINT
%   2. STATIC CONFIG OF THE RISER
%   3. AXIAL TENSION
%
%Name: Emir Lejlic.
%Date: 10.05.13
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Creating a waitbar
h = waitbar(0,'Finding important paramters from stamod.res file...');

    for i = 1:length(run_data)

        waitbar(i / length(run_data))

        %------------------------------------------------------------------
        %SEARCHING FOR THE TDP IN THE STAMOD.RES FILE.
        %------------------------------------------------------------------

        %The folder where the stamod.res file is.
        res_file_path = fullfile(run_folder,run_data(i,1).name,res_filename);
        %Going to the folder.

        %Reading the .res file containing the TDPs.
        stamod_res_text = file2str(res_file_path);

        %Finding the Seafloor spring / friction status in the text
        str_seafloor = strfind(stamod_res_text,'Seafloor spring / friction status');

        %This is the string after the
        str_you_may = strfind(stamod_res_text,'You may');

        %This is now the only
        seafloor_spring_text = stamod_res_text(str_seafloor:str_you_may-1);

        %Finding the last (-) in the text. It is possible to find the TDP
        %element after that.
        str_L = strfind(seafloor_spring_text,'(L)');

        %Extracting the element no.
        element_no = seafloor_spring_text(str_L(end)+13:str_L(end)+16);

        %Adding the TDP and its respective x value point on the riser element to the struct.
        run_data(i,1).TDP_element = str2double(element_no);
        run_data(i,1).TDP_x_value = run_data(i,1).fatigue_damage(str2double(element_no),1);

        %-------------------------------------------------------------------------
        %SEARCHING FOR THE STATIC CONFIG IN THE STAMOD.RES FILE.
        %-------------------------------------------------------------------------

        %Finding the X-coord. in the text.
        str_xcoord = strfind(stamod_res_text,'X-coord.');

        %This is the string after the the static results.
        str_static_results = strfind(stamod_res_text,'STATIC RESULTS WRITTEN TO FILE');

        %This is the important text.
        config_text_tmp = stamod_res_text(str_xcoord:str_static_results-1);

        %Finding 'number' in the text.
        str_number = strfind(config_text_tmp,'number');

        %The config results are now.
        config_text = config_text_tmp(str_number+length('number')+1:end);

        %Making the text into an array with the results.
        %Under it is possible to see the headers.
        %| Node | X-coord. | Y-coord. | Z-coord. | Y-moment | Y-curv. | Z-curv. |
        run_data(i,1).static_config = str2num(config_text);

        %---------------------------------------------------------------------
        %SEARCHING FOR THE AXIAL IN THE STAMOD.RES FILE.
        %---------------------------------------------------------------------

        %Finding the ELEMENT AXIAL AXIAL in the .res file
        str_elem_axial = strfind(stamod_res_text,'Element     Axial     Axial');

        %Finding the en (Node X-coord.
        str_node_xcoord = strfind(stamod_res_text,'Node        X-coord.');

        %This is the tmp text.
        axial_text_tmp = stamod_res_text(str_elem_axial:str_node_xcoord-1);

        %Finding the last 'angle' in the text.
        str_angle = strfind(axial_text_tmp,'angle');

        %The important text.
        axial_text = axial_text_tmp(str_angle(end)+length('angle')+1:end);

        %Inserting into a matrix in the struct.
        run_data(i,1).axial_force = str2num(axial_text);

        %Removing the unnecessary cols.
        run_data(i,1).axial_force(:,3:end) = [];


    end

    close(h)

end
```