Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Engineering
Department of Structural Engineering

Marthe Elise Leirvåg
Alida Malén Trondsen

Estimation of Aerodynamic Admittance Functions for a Twin-box Suspension Bridge Crossing the Halsafjord

Wind Tunnel Testing and Pressure Tap Measurements

Master's thesis in Civil and Environmental Engineering
Supervisor: Ole Andre Øiseth
Co-supervisor: Øyvind Wiig Petersen and Oddbjørn Kildal
June 2022

NTNU
Norwegian University of
Science and Technology

Marthe Elise Leirvåg
Alida Malén Trondsen

Estimation of Aerodynamic Admittance Functions for a Twin-box Suspension Bridge Crossing the Halsafjord

Wind Tunnel Testing and Pressure Tap Measurements

Master's thesis in Civil and Environmental Engineering
Supervisor: Ole Andre Øiseth
Co-supervisor: Øyvind Wiig Petersen and Oddbjørn Kildal
June 2022

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

NTNU
Norwegian University of
Science and Technology

Department of Structural Engineering
Faculty of Engineering
NTNU – Norwegian University of Science and Technology

ACCESSIBILITY

MASTER'S THESIS 2022

SUBJECT AREA:	DATE:	NO. OF PAGES:
Structural Dynamics	10.06.2022	194 (18 + 119 + 57)

TITLE:

Estimation of the Aerodynamic Admittance Functions for a Twin-box Suspension Bridge Crossing the Halsafjord

Estimering av aerodynamiske frekvensresponsfunksjoner for en hengebru med dobbelt kassetverrsnitt over Halsafjorden

BY:

Marthe Elise Leirvåg

Alida Malén Trondsen

SUMMARY:

This thesis aims to study the pressure distribution on a twin-box bridge and estimate the aerodynamic admittance functions. A section model of a twin-box bridge was constructed and tested in the wind tunnel at the Department of Energy and Process Engineering at NTNU Trondheim for different wind velocities, angles of attack, and active grid-generated turbulence. The pressure distribution was measured with 256 pressure tubes connected to four MSP4264 pressure scanners. The tubes were distributed along six strips on each box to investigate the coherence.

The aerodynamic admittance functions were estimated with three different methods; the general, the auto-spectral, and the cross-spectral. The estimated admittance functions were compared to each other, the Sears function, and previous research to validate the results. All methods displayed a peak at approximately 50 Hz, which was the same frequency as the peaks observed in the force spectra. These peaks were mainly caused by the downstream-box due to vortex shedding. Moreover, the Sears function deviated significantly from the identified admittance functions and is not considered applicable for the twin-box bridge. The Sears function overestimated the admittance function for drag, while the admittance functions for lift and moment were highly underestimated. The cross-spectral admittance functions showed some deviations compared to the auto-spectral method, which could indicate that the auto-spectral method produces inaccurate results. Nevertheless, based on the results obtained in this master's thesis, as the force spectra and static coefficients, the estimated admittance functions seem reasonable and are a valid representation of the buffeting forces acting on the twin-box.

RESPONSIBLE TEACHER: Professor Ole Andre Øiseth

SUPERVISOR(S): Professor Ole Andre Øiseth, Associate Professor Øyvind Wiig Petersen and PhD Candidate Oddbjørn Kildal

CARRIED OUT AT: Department of Structural Engineering

Preface

The research presented in this report is the final product of our master's thesis at the Norwegian University of Science and Technology (NTNU), Department of Structural Engineering. A great motivation for implementing this report was the variation of practical work, including the building process and wind tunnel testing, combined with the theoretical aspect of the wind-induced response. The process has been challenging but also incredibly interesting and instructive. We appreciate our unique opportunity to participate in the research within such an important and exciting area.

We want to express our gratitude to our supervisor Professor Ole Andre Øiseth, for his expertise, engagement, guidance, and always being available when needed. We also want to thank Associate Professor Øyvind Wiig Petersen for his commitment, competence and help with data processing. In addition, we would extend our great gratitude to PhD Candidate Oddbjørn Kildal for providing his help with the wind tunnel testing, data processing, and good conversations. Further, we want to thank Marius Østnor Døllner, Pål Erik Endrerud and Gøran Loraas for their great assistance with the laboratory work and interest in our project.

Last, we want to thank our families and loved ones for their cheering and support throughout our education.

We build too many walls and not enough bridges.

- Isaac Newton

Marthe E. Leirvåg

Marthe Elise Leirvåg

Alida M. Trondsen

Alida Malén Trondsen

Trondheim, 10th June 2022

Abstract

Numerous long-span suspension bridges are planned as part of Norway's "Ferry-free E39" project, including a bridge crossing the Halsafjord. The fjord is approximately 2 kilometres wide, and the long span and rough weather conditions complicate the bridge design. As bridges are built longer and longer, the dynamic effects of wind dominate in terms of structural loading effects and become the main issue in their overall design. Twin-box bridges have been shown to improve aerodynamic stability by changing the surface pressure distribution around the bridge deck. Nonetheless, there is limited experience with twin-box bridges. Wind tunnel testing has shown that the loads on twin-box bridges are far more complicated than on single-box bridges. Vibration in terms of buffeting loads induced by the fluctuating wind and vortex-induced vibrations may be encountered due to the gap between the bridge decks.

This thesis aims to study the pressure distribution on a twin-box bridge and estimate the aerodynamic admittance functions. A section model of a twin-box bridge was constructed and tested in the wind tunnel at the Department of Energy and Process Engineering at NTNU Trondheim for different wind velocities, angles of attack and active grid-generated turbulence. The pressure distribution was measured with 256 pressure tubes connected to four MSP4264 pressure scanners. The tubes were distributed along six strips on each box to investigate the coherence.

The study consists of four main phases. The first phase focused on theoretical studies of wind effects, wind-tunnel testing, aerodynamic admittance functions, and the development and design of a 3D-printed mid-section. The second phase was the building process. A 1:50 scale twin-box bridge was built, and pressure tubes were fastened to the 3D-printed model and the four pressure scanners. Further, the third phase consisted of additional knowledge of the experimental setup and processing, pretests of the pressure scanners, and the final tests conducted in the wind tunnel. The last phase consisted of data processing and interpretation of the final results. The aerodynamic admittance functions were estimated with three different methods; the general, the auto-spectral, and the cross-spectral. The calculated pressure distribution, static load coefficients and the estimated admittance functions were evaluated and compared with previous research.

The estimated admittance functions obtained from the different methods were compared to each other and the Sears function. Previous research was also used to compare and validate the results. All methods displayed a peak at approximately 50Hz, which was the same frequency as the peaks observed in the force spectra. These peaks were mainly caused by the downstream-box due to vortex shedding. Moreover, the Sears function deviated significantly from the identified admittance functions and is not considered applicable for the twin-box bridge. The Sears function overestimated the admittance function for drag, while the admittance functions for lift and moment were highly underestimated. The cross-spectral admittance functions showed some deviations compared to the auto-spectral method, which could indicate that the auto-spectral method produces inaccurate results. Nevertheless, based on the results obtained in this master's thesis, as the force spectra and static coefficients, the estimated admittance functions seem reasonable and are a valid representation of the buffeting forces acting on the twin-box.

Sammendrag

Flere hengebruer med lange spenn er planlagt som en del av Norges "Fergefri E39"- prosjekt, inkludert en bru over Halsafjorden. Fjorden er omtrent 2 kilometer bred, og det lange spennet og de røffe værforholdene gjør brudesignet mer komplisert. Ettersom bruene bygges lengre og lengre, dominerer de dynamiske vindeffektene når det kommer til de strukturelle belastningseffektene. Disse vindeffektene er hovedutfordringen når det kommer til brudesignet. Dobbel-kasse bruer har vist seg å forbedre den aerodynamiske stabiliteten ved å endre trykkfordelingen på overflaten rundt brutverrsnittet. Likevel er det begrenset med erfaring med slike bruer. Gjennom vindtunneltesting har det bitt vist at bruer med dobbel-kasse tverrsnitt er langt mer kompliserte enn de tradisjonelle enkelt-kasse bruene. Vibrasjoner på grunn av buffeting krefter fra vinden og virvel-indusert vibrasjoner kan oppstå på grunn av avstanden mellom brudekkene.

Målet med denne oppgaven er å studere trykkfordelingen på dobbel-kasse bruen og estimere frekvensresponsfunksjonene. En modell av en dobbel-kasse bru ble bygget og testet i vindtunnelen ved Institutt for energi- og prosessteknikk ved NTNU Trondheim. Tester med ulike vindhastigheter, angrepsvinkler og aktivt gittergenerert turbulens ble utført. Trykkfordelingen ble målt med 256 plastrør som var koblet til fire MSP4264 trykkskannere. Disse plastrørene ble fordelt på seks linjer i spenn retningen for å undersøke korrelasjonen.

Oppgaven besto i hovedsak av fire faser. Den første fasen besto av det teoretiske grunnlaget av blant annet vind effektene, vindtunnel testing og frekvensresponsfunksjonene. I tillegg ble det designet og utviklet en 3D-printet modell for plassering av plastrørene. Den andre fasen var byggeprosessen der en dobbel-kasse bru i skala 1:50 ble bygget. Plastrørene ble festet til den 3D-printede modellen og de fire trykkskannerne. Videre besto den tredje fasen av å sikre oversikt over det eksperimentelle oppsettet, samt en test av trykkskannerne før de faktiske testene i vindtunnelen ble utført. Siste fase av oppgaven besto av å prosessere data og tolking av de endelige resultatene. Frekvensresponsfunksjonene ble estimert ved bruk av tre forskjellige metoder; den generelle, den auto spektrale og den kryss spektrale metoden. Trykkfordelingen, de statiske koeffisientene og de estimerte frekvensresponsfunksjonene ble evaluert og sammenlignet med tidligere forskning.

De estimerte frekvensresponsfunksjonene fra de ulike metodene ble sammenlignet med hverandre, Sears funksjonen og tidligere forskning for å validere resultatene. De estimerte frekvensresponsfunksjonene viste en topp på omtrent 50 Hz, noe som også ble observert i lastspektrene. Disse toppene er i hovedsak forårsaket av nedstrøms-kassen på grunn av virvelavgivelse fra oppstrøms-kassen. Det ble observert at Sears funksjonen avvek betydelig fra de identifiserte frekvensresponsfunksjonene og ansees derfor ikke som gjeldende for bruer med dobbelt-kasse tverrsnitt. Sears funksjonen overestimerte frekvensresponsfunksjonen for drakreftene, mens den underestimerte for løft og moment. De kryss spektrale frekvensresponsfunksjonene viste noen avvik sammenlignet med den auto spektrale metoden, noe som kan indikere at den auto spektrale metoden estimerer unøyaktige resultater. Ut fra resultatene i denne oppgaven, som blant annet lastspektrene og de statiske koeffisientene, ansees de estimerte frekvensresponsfunksjonene rimelig og kan brukes for å forstå buffeting kreftene på en bru med dobbelt-kasse tverrsnitt.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Suspension bridges are subjected to enormous wind forces, resulting in extreme loads. To build suspension bridges that can withstand these loads, detailed knowledge of the effects on the structures is required. Understanding the wind-induced behavior of bridges has come a long way throughout history. One event that was a turning point in the history of bridge design was the Tacoma Narrows Bridge disaster in 1940. Following the Tacoma Narrows, aeronautical engineers made a significant contribution to the development of aerodynamics in civil engineering. Among other things, it became common practice to conduct tests in a conventional aeronautical wind tunnel in uniform, smooth airflow rather than with simulated natural winds [1].

As bridges are built longer and longer, the dynamic effects of wind dominate in terms of structural loading effects and become the main issue in their overall design. Bridges with a main span of about 1 km or more are often referred to as "long span" bridges. As span length and slenderness increase, the flexibility of these bridges becomes high, and the first natural frequency is typically 0.1 Hz or below [2]. As a result, long-span bridges become more sensitive to the wind, increasing the aerodynamic stability requirements.

For wind-induced vibration of long-span bridges, buffeting is a central research area. It is a random force vibration, which is generated by the structure under the action of natural wind fluctuation components. Low wind speed causes buffeting, and the effects can result in fatigue or serviceability issues and are therefore important with the increasing span of bridges. Up to now, the buffeting response has mainly been obtained by theoretical calculation and wind tunnel tests. The theoretical model is based on pioneering work achieved by Davenport and has later been enhanced in several studies. Nonetheless, due to the complexity and diversity of bridge cross-sections and the characteristics of atmospheric turbulence, it is difficult to estimate the aerodynamic forces and wind-induced repose entirely through theoretical analysis. Wind tunnel tests are therefore an important and essential method for calculating the buffeting response.

The aerodynamic admittance function (AAF) is an essential function for estimating the buffeting response. It is a transfer function that transfers the turbulent wind fluctuations to buffeting forces. Sears (1941) was the first to propose a theoretical approach for the AAF on streamlined bodies in the frequency domain. Later, Davenport (1962) extended the theory to buffeting analysis of bridges based on the quasi-steady theory. However, for bluff bodies, separation and reattachment of the flow make the spatial characteristics of the aerodynamic forces far more complex. Hence, the cross-sectional admittance functions may be determined from wind tunnel experiments with time series of drag, lift, and moment forces or by pressure tap measurements around the periphery of the cross-section. The pressure distribution can be calculated and indirectly described by the

AAF by pressure measurements. This is, however, "one step further" from standard tests with force measurements. There are few previous studies, and the experience is limited.

A better understanding of the pressure distribution and the flow around the bridge cross-section leads to a better description of the response, which has several benefits. More knowledge of wind-induced behavior provides opportunities for longer bridges and may affect material consumption. The correct choice and the right amount of material are important to ensure that the bridge withstands the wind effects. In addition, it may lead to reduced costs and climate footprint. However, safety is the first priority in civil engineering, and underestimating the response can have catastrophic consequences.

Today, numerous long-span suspension bridges are planned as part of Norway's "Ferry-free E39" project. The design of single box bridges with a span length of over 1700 meters that satisfy the aerodynamic requirements is troublesome. However, twin-box bridges have been shown to improve aerodynamic stability by changing the surface pressure distribution around the bridge deck. In addition, it may result in lighter structures, and they are therefore more financially appealing. Thus, twin-box bridges have received much attention and have been used in super long-span suspension bridges, such as the 1915 Çanakkale Bridge with a center span of 2023 m, the Ti Sun-sin Bridge with a center span of 1545 m, and the Xihoumen Bridge with a center span of 1650 m [3].

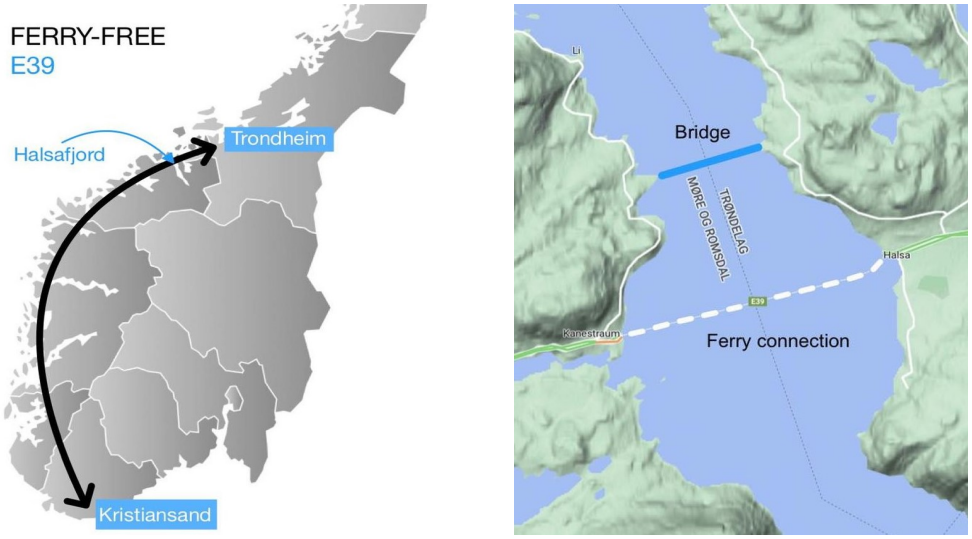
Nonetheless, there is limited experience with twin-box bridges. Wind tunnel testing has shown that a twin-box bridge's wind loads are far more complicated than for a single-box bridge [4]. Vibration in terms of buffeting loads induced by the fluctuating wind and vortex-induced vibrations may be encountered due to the gap between the bridge decks. There are few previous studies focusing on the aerodynamic admittance of a twin-box bridge. Therefore, in this thesis, the aim is to estimate the AAFs of a twin-box bridge. A section model of a twin-box bridge shall be built and tested in a wind tunnel for different wind velocities and active grid-generated turbulence. To study the pressure distribution and the aerodynamic forces, 256 pressure tubes shall be used to measure the surface pressure around the periphery of the cross-section. The pressure tubes will be separated and distributed along six strips for the opportunity to investigate the correlation between the buffeting forces. Further, the AAFs will be estimated.

Chapter 2

Bridge Crossing the Halsafjord

The Coastal Highway Route E39 goes from Kristiansand in the south to Trondheim in the north, as seen in Figure 2.1(a). The road is approximately 1100 kilometers along the west coast of Norway and passes the citis; Stavanger, Stord, Bergen, Førde, Ålesund, and Molde. Today, the travel time is about 21 hours with seven ferry connections. The Norwegian Public Roads Administration (NPRA) aims to improve E39 and make it ferry-free to halve the travel time on the entire stretch and also between the cities. Improving the current E39 will link large business regions, housing, labor, and service markets more closely and contribute to developing Norway's largest export region. This shall be done by making the stretch almost 50 kilometers shorter, replacing ferries with fixed connections or more frequent ferry departures, as well as improving the road between the fjords along the stretch [5].

A central part of the improvements of E39 is the project "Ferry-free E39", where the plan is to replace the ferry connections with bridges and sub-sea tunnels. A bridge shall replace the ferry connection across the Halsafjord. The fjord is approximately 2 kilometers wide and has a depth of 500 meters [6]. The proposed location of the Halsafjord bridge is illustrated with a blue line in Figure 2.1(b), while the current ferry connection is outlined in white.



(a) The "Ferry-free E39" project.

(b) The proposed location of the bridge across the Halsafjord

Figure 2.1: An overview of the project "Ferry-free E39" and the Halsafjord.

The long span and the rough weather conditions make the bridge design complicated. Several different variants of bridge concepts have been considered, such as the concept of a twin-box bridge. After three years of obtaining necessary data about the fjord, it was announced a preliminary project. It involved a study of three different bridge concepts for the Halsafjord, which should provide the Norwegian Public Roads Administration ground for decision-making. Bridge concepts of a floating bridge and a suspension bridge with one span were delivered in April this year by the work community involving Norconsult and Dr. techn. Olav Olsen. In addition, the work community consisting of Aas-Jakobsen, Multiconsult, and COWI delivered a concept of a suspension bridge with two spans with tension leg platform [6].

Chapter 3

Literature Review

This chapter presents relevant theory for the work done in this master's thesis. It gives a brief introduction to modal analysis, wind induced response, scaling laws, wind tunnel effects and aerodynamic admittance functions, among other things.

3.1 Modal Analysis

Modal analysis is the process of specifying the dynamic characteristics of a system and using them to formulate a mathematical model of its dynamic behaviour. It is based upon the fact that the structural displacements $\mathbf{r}$(x,t) can be expressed by the sum of the products between the natural eigenmodes, $\boldsymbol{\phi}_i(x)$, and the generalized coordinates, $\eta_i(t)$, i.e. [7]:

$$\mathbf{r}(x,t) = \sum_{i}^{N_{mod}} \boldsymbol{\phi}_i(x) \cdot \eta_i(t) = \boldsymbol{\phi}(x) \cdot \boldsymbol{\eta}(t) \tag{3.1}$$

where

$$\boldsymbol{r}(x,t) = \begin{bmatrix} r_y & r_z & r_\theta \end{bmatrix}^T \tag{3.2}$$

$$\boldsymbol{\phi}(x) = \begin{bmatrix} \boldsymbol{\phi}_i(x) & ... & \boldsymbol{\phi}_i(x) & ... & \boldsymbol{\phi}_{N_{mod}}(x) \end{bmatrix} \tag{3.3}$$

$$\boldsymbol{\eta}(t) = \begin{bmatrix} \eta_i(t) & ... & \eta_i(t) & ... & \eta_{N_{mod}}(t) \end{bmatrix}^T \tag{3.4}$$

$\boldsymbol{\phi}_i(x) = \begin{bmatrix} \phi_y & \phi_z & \phi_\theta \end{bmatrix}_i^T$ and N_{mod} is number of modes that is necessary for a sufficiently accurate solution.

By inserting Equation 3.1 into the equilibrium equations of the system, followed by span-wise integration, the equation of motion in modal frequency domain is obtained:

$$\tilde{\boldsymbol{M}}_0 \cdot \ddot{\boldsymbol{\eta}} + \tilde{\boldsymbol{C}}_0 \cdot \dot{\boldsymbol{\eta}} + \tilde{\boldsymbol{K}}_0 \cdot \boldsymbol{\eta} = \tilde{\boldsymbol{Q}}(t) + \tilde{\boldsymbol{Q}}_{ae}(t, \eta, \dot{\eta}, \ddot{\eta}) \tag{3.5}$$

The modal mass $\tilde{\boldsymbol{M}}_0$, damping $\tilde{\boldsymbol{C}}_0$ and stiffness $\tilde{\boldsymbol{K}}_0$ are obtained in still air and defined by:

$$\begin{matrix} \tilde{\boldsymbol{M}}_0 = diag\left[\tilde{M}_i\right] \\ \tilde{\boldsymbol{C}}_0 = diag\left[\tilde{C}_i\right] \\ \tilde{\boldsymbol{K}}_0 = diag\left[\tilde{K}_i\right] \end{matrix} \quad where \begin{cases} \tilde{M}_i = \int_L (\boldsymbol{\phi}_i^T \cdot \boldsymbol{M}_0 \cdot \boldsymbol{\phi}_i) dx \\ \tilde{C}_i = 2\tilde{M}_i \omega_i \zeta_i \\ \tilde{K}_i = \omega_i^2 \tilde{M}_i \end{cases} \tag{3.6}$$

where ω_i are the eigen-frequencies and ζ_i are the damping ratios associated with the corresponding eigen-modes.

The total modal wind load, $\tilde{\boldsymbol{Q}}(t)$ and the motion induced load, $\tilde{\boldsymbol{Q}}_{ae}(t, \eta, \dot{\eta}, \ddot{\eta})$ on the right hand side of Equation 3.5 are derived by integration over the wind exposed part of the bridge (L_{exp}):

$$\tilde{\boldsymbol{Q}}_i(t) = \int_{L_{exp}} (\boldsymbol{\phi}_i^T \cdot \boldsymbol{q}) dx \tag{3.7}$$

$$\tilde{\boldsymbol{Q}}_{ae_i}(t, \eta, \dot{\eta}, \ddot{\eta}) = \int_{L_{exp}} (\boldsymbol{\phi}_i^T \cdot \boldsymbol{q}^{\boldsymbol{SE}}) dx \tag{3.8}$$

where the cross sectional load vectors contains three components representing drag, lift and moment load per unit length [7].

3.2 Wind Induced Response

When an airflow meets a long-span bridge deck, it will cause the bridge to move. In addition, the interaction between the wind pressure and the deformation of the structure will change the loads that the wind generates [8]. This wind-induced dynamic response can be classified into three main categories; random response due to buffeting by turbulence, vortex shedding, and motion-induced forces. Buffeting is a vibration phenomenon that stems from pressure fluctuations in the oncoming flow. Vortex shedding is vortices with alternating rotations which produce a vertical force. In each vortex, the force changes in direction, causing vibrations of the deck. Motion-induced instabilities are forces from the interaction between the flow and the oscillating structure itself [7].

These mentioned effects occur at all wind velocities but are vital in fairly separate wind velocity regions. Vortex shedding is strongest at low wind velocities, buffeting forces occur in stronger wind velocities, while motion-induced forces are strongest at even higher velocities. Therefore the response calculations are usually treated separately [7]. This is illustrated in Figure 3.1 below.

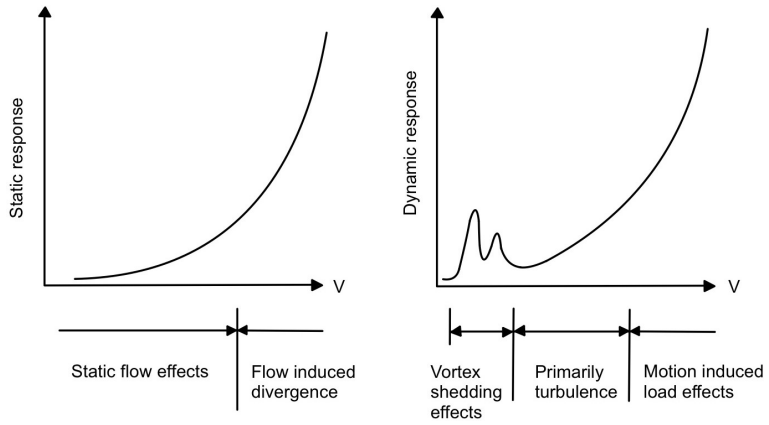


Figure 3.1: Typical behavior for a slender bridge deck, illustration based on [7].

Generally, three force components and three moment components can be considered when a 3D structure is exposed to wind. However, to simplify the idea, 2D alternatives could be considered as convenient mathematical models in many wind engineering problems. Some approximations that can be done are the strip theory assumption and the quasi-steady approximation.

3.2.1 The Strip Theory

The strip theory was originally introduced for aerofoils but is often used for bridges. Instead of looking at the whole structure, a strip of unit thickness cut off by two planes in the mean wind direction is considered. This can be done since bridges are only extended in one direction, and the main concern is their behaviour when the wind comes perpendicular to its longitudinal axis. Then, three components need to be considered; the lift force F_L, the drag force F_D, and the pitching moment F_M [9].

3.2.2 Quasi-Steady Theory

Another well-known approximation in bridge engineering is the Quasi-Steady theory. The approximation ignores the history of motion in the aerodynamic model. Put differently, the aerodynamic

forces at any time depend only on the instantaneous position of the body and velocity at that instant. This is an acceptable assumption for relatively high wind speed but unacceptable, for instance, in the case of vortex shedding [9].

The Buffeting Theory

The buffeting load on a structure is associated with velocity fluctuations in the oncoming flow and motion-induced contributions included in the total wind load. It is assumed that any fluctuating quantity can be split into a time-invariant mean part depending on position and a fluctuating part with zero mean, depending on both position and time. The wind velocity is therefore divided into the stationary wind speed, V, and the fluctuating terms, u and w.

In Figure 3.2 below, the buffeting load is illustrated on a bridge cross-section. There are two displacement configurations; the mean displacement, $\bar{r}_i$ and a fluctuating part around the mean configuration, r_i. The cross section is first given the displacements $\bar{r}_y(x)$, $\bar{r}_z(x)$ and $\bar{r}_\theta(x)$ at an arbitrary position along the span. About this position the structure starts to oscillates, and additional dynamic displacements $r_y(x,t)$, $r_z(x,t)$ and $r_\theta(x,t)$ are given [7].

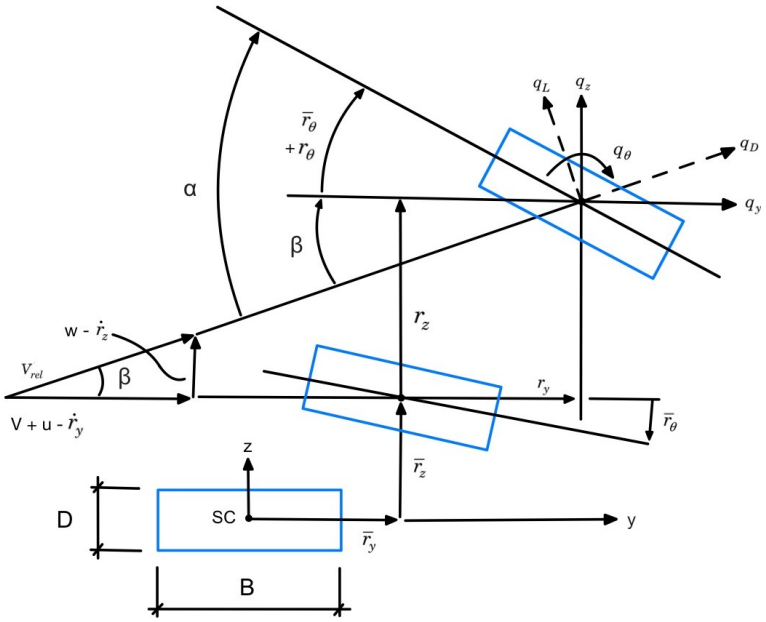


Figure 3.2: Buffeting load acting on a bridge cross-section, illustration based on [7].

The forces and moments in the local coordinate system of the fluctuating wind can be expressed by [7]:

$$\begin{bmatrix} q_D(x,t) \\ q_L(x,t) \\ q_M(x,t) \end{bmatrix} = \frac{1}{2}\rho V_{rel}^2 \begin{bmatrix} D \cdot C_D(\alpha) \\ B \cdot C_L(\alpha) \\ B^2 \cdot C_M(\alpha) \end{bmatrix} \tag{3.9}$$

where ρ is the air density, D and B are the height and depth of the cross-section respectively, C_D, C_L and C_M are the force coefficients, α is the corresponding angle of flow incidence and V_{rel} is the relative wind velocity defined by:

$$V_{rel}^2 = (V + u(t) - \dot{r}_y(t))^2 + (w(t) - \dot{r}_z(t))^2 \tag{3.10}$$

The forces can be transformed to the global coordinate system of the section:

$$\mathbf{q}_{tot}(x,t) = \begin{bmatrix} q_y \\ q_z \\ q_\theta \end{bmatrix} = \begin{bmatrix} cos(\beta) & -sin(\beta) & 0 \\ sin(\beta) & cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} q_D \\ q_L \\ q_M \end{bmatrix} \tag{3.11}$$

where

$$\tan\beta = \frac{w - \dot{r}_z}{V + u - \dot{r}_y} \tag{3.12}$$

Furthermore, it is assumed that the fluctuating flow components (u, w) and the structural displacements $(\dot{r}_y, \dot{r}_z)$ are small compared to the mean wind velocity, V. Then, $tan\beta \approx \beta$, and β can be expressed as:

$$\beta \approx \frac{w - \dot{r}_z}{V} \tag{3.13}$$

And thus:

$$V_{rel}^2 \approx V^2 + 2Vu - 2V\dot{r}_y \tag{3.14}$$

$$\alpha = \bar{r_\theta} + r_\theta + \beta \approx \bar{r_\theta} + r_\theta + \frac{w}{V} - \frac{\dot{r}_z}{V} \tag{3.15}$$

In addition, the nonlinear variation of the force coefficients is replaced by a linear approximation:

$$\begin{bmatrix} C_D(\alpha) \\ C_L(\alpha) \\ C_M(\alpha) \end{bmatrix} = \begin{bmatrix} C_D(\bar{\alpha}) \\ C_L(\bar{\alpha}) \\ C_M(\bar{\alpha}) \end{bmatrix} + \alpha_f \cdot \begin{bmatrix} C'_D(\bar{\alpha}) \\ C'_L(\bar{\alpha}) \\ C'_M(\bar{\alpha}) \end{bmatrix} \tag{3.16}$$

where $\bar{\alpha}$ is the mean and α_f is the fluctuating part of the angle of attack. $C'_D(\bar{\alpha}), C'_L(\bar{\alpha})$ and $C'_M(\bar{\alpha})$ are the slopes of the coefficients curves at $\bar{\alpha}$.

By this, rewriting Equation 3.11, the following is obtained:

$$\mathbf{q}_{tot}(x,t) = \begin{bmatrix} \bar{q_y}(x) \\ \bar{q_z}(x) \\ \bar{q_\theta}(x) \end{bmatrix} + \begin{bmatrix} q_y(x,t) \\ q_z(x,t) \\ q_\theta(x,t) \end{bmatrix} = \bar{\mathbf{q}} + \mathbf{B}_q \cdot \mathbf{v} + \mathbf{C}_{ae} \cdot \dot{\mathbf{r}} + \mathbf{K}_{ae} \cdot \mathbf{r} \tag{3.17}$$

where

$$\mathbf{v}(x,t) = \begin{bmatrix} u & w \end{bmatrix}^T \tag{3.18}$$

$$\mathbf{r}(x,t) = \begin{bmatrix} r_y & r_z & r_\theta \end{bmatrix}^T \tag{3.19}$$

$$\bar{\mathbf{q}}(x,t) = \begin{bmatrix} \bar{q_y}(x) \\ \bar{q_z}(x) \\ \bar{q_\theta}(x) \end{bmatrix} = \frac{1}{2}\rho V^2 B \begin{bmatrix} (C/D)\bar{C}_D \\ \bar{C}_L \\ B\bar{C}_M \end{bmatrix} = \frac{1}{2}\rho V^2 B \cdot \hat{\mathbf{b}}_q \tag{3.20}$$

$$\mathbf{B}_q(x) = \frac{1}{2}\rho V \begin{bmatrix} 2D\bar{C}_D & DC'_D - B\bar{C}_L \\ 2B\bar{C}_L & BC'_L + D\bar{C}_D \\ 2B^2\bar{C}_M & B^2\bar{C}_M \end{bmatrix} = \frac{1}{2}\rho V \cdot \hat{\mathbf{B}}_q \tag{3.21}$$

$$\mathbf{C}_{ae}(x) = -\frac{1}{2}\rho V \begin{bmatrix} 2D\bar{C}_D & DC'_D - B\bar{C}_L & 0 \\ 2B\bar{C}_L & BC'_L + D\bar{C}_D & 0 \\ 2B^2\bar{C}_M & B^2\bar{C}_M & 0 \end{bmatrix} \tag{3.22}$$

$$\mathbf{K}_{ae}(x) = \frac{1}{2}\rho V^2 \begin{bmatrix} 0 & 0 & DC'_D \\ 0 & 0 & BC'_L \\ 0 & 0 & B^2C'_M \end{bmatrix} \tag{3.23}$$

$\bar{\mathbf{q}}$ is a time invariant mean part, $\mathbf{B}_q \cdot \mathbf{v}$ is the dynamic loading associated with the turbulence, while $\mathbf{C}_{ae}$ and $\mathbf{K}_{ae}$ are motion induced loads related to the structural displacement and velocity, respectively [7].

3.2.3 Identification of static load coefficients

The static aerodynamic load coefficients C_D, C_L and C_M are dependent on the angle of attack. They can be estimated with a static wind tunnel test, where the cross-section rotates, and the forces are measured. The static coefficients can be expressed as:

$$\begin{bmatrix} C_D(\alpha) \\ C_L(\alpha) \\ C_M(\alpha) \end{bmatrix} = \frac{1}{\frac{1}{2}\rho V^2 L} \begin{bmatrix} \frac{F_D(\alpha)}{D} \\ \frac{F_L(\alpha)}{B} \\ \frac{F_M(\alpha)}{B^2} \end{bmatrix} \tag{3.24}$$

where C_D is the drag force coefficient, C_L is the lift force coefficient and C_M is the moment force coefficient, respectively, F_D is the drag force, F_L is the lift force and F_M is the moment force. ρ is the air density, V is the wind velocity and D, B and L are the height, width and the length of the bridge cross section.

3.2.4 Vortex Shedding

When a bluff body like a long-span bridge deck is met by an airflow, sharp edges will separate the flow, causing vortices to be shed in the wake of the body. Vortex shedding is vortices with alternating rotations which produce a vertical force. In each vortex, the force changes in direction, by that causing vibrations of the deck. A dominant frequency characterizes the fluctuations in the cross-wind force, the vortex shedding frequency, f_s, is given by:

$$f_s = St \cdot \frac{V}{D} \tag{3.25}$$

where St is the Strouhal Number which is a function of the geometry and Reynolds number, V is the mean wind velocity and D is the across wind width of the deck [7].

Resonance will occur when the vortex shedding frequency, f_s, is equal to any natural frequency of the structure associated with vibrations in the across wind direction or in torsion. If the wind velocity slowly increases from zero, f_s, will increase accordingly. Resonance will occur when f_s

becomes equal to the lowest natural frequency. The next resonance will occur when f_s is equal to the subsequent natural frequency, and so on. Hence, there is a resonance velocity for every natural frequency, which according to Equation 3.25 is expressed by:

$$V = \frac{f_s D}{St} \tag{3.26}$$

Nevertheless, experiments have shown that f_s will deviate from Equation 3.26 for a specific range of wind velocities. This is called lock-in and happens when resonance occurs due to interaction between the flow and the oscillating structure. The vortex shedding frequency, f_s will be equal to or stay close to the natural frequency, f_n. The fluctuating load becomes more correlated in the spanwise direction at a lock-in, adding a motion-induced part. However, these effects decrease when the fluctuating structural displacements become large [7].

Although twin-box bridges have aerodynamic advantages in flutter stability, the gap makes it more responsive to vortex shedding excitation than a single bridge deck. During testing, lock-in may occur and can cause problems if the critical velocity region is the same as the velocity in the wind tunnel test.

3.3 Scaling Laws

In order to get accurate results when comparing the wind tunnel test with a full-scale model, some non-dimensional quantities and scaling laws must be introduced. The cross-section in this thesis is based on a cross-section from an earlier master's thesis with the dimension 1:50 compared to the full-scale scale bridge deck cross-section.

3.3.1 Scaling Wind Tunnel Model

The bridge model dimensions have to be adapted to fit the wind tunnel. It has to be scaled to fit the wind tunnel's size and withstand the maximum wind velocity in the wind tunnel. The geometric scale of the bridge is defined as [10]:

$$\lambda_L = \frac{L_{WT}}{L_{FS}} \tag{3.27}$$

where WT stands for Wind Tunnel and FS stands for Full Scale and L is the length.

3.3.2 Reduced Frequency and Reduced Velocity

The non-dimensional frequency is usually referred to as the reduced frequency and is defined as:

$$f_r = \frac{fB}{V} \tag{3.28}$$

where f is the frequency, B is the width of the deck and V is the wind velocity. The reduced frequency can be used to indicate the unsteadiness of the system.

Reduced velocity is defined as:

$$V_r = \frac{V}{f_n B} \tag{3.29}$$

where f_n is the natural frequency, B is the width of the deck and V is the wind velocity as mentioned.

The relation between reduced frequency and reduced velocity can be used as a comparison between the wind tunnel model (WT) and the full scale model (FS):

$$\frac{V_{FS}}{f_{FS}B_{FS}} = \frac{V_{WT}}{f_{WT}B_{WT}} \tag{3.30}$$

3.3.3 Reynolds Number

Reynolds number (Re) measures the turbulence in a fluid. It is a dimensionless number and the ratio of inertia and viscous forces. The following formula gives the Reynolds number:

$$
\begin{aligned}
Re &= \frac{inertia forces}{viscous forces} = \frac{mass \cdot acceleration}{shear stress \cdot area} \\
&= \frac{\rho L^3 \frac{V_\infty}{T}}{\mu \frac{V_\infty}{L} L^2} = \frac{\rho L^2}{\mu T} = \frac{\rho L^2}{\mu \frac{L}{V_\infty}} = \frac{\rho V_\infty L}{\mu} = \frac{V_\infty L}{v}
\end{aligned}
\tag{3.31}
$$

where L is characteristic length of the gust, V_∞ is characteristic velocity, T is the characteristic time, μ is the dynamic viscosity and v is the kinematic viscosity (μ/ρ).

The wind tunnel has limitations that make it almost impossible to obtain Reynolds number similarity. The kinematic viscosity of the air does not vary much between the test and the full-scale bridge. Therefore, the only way to compensate for the scaled length is to increase the wind speed in the wind tunnel. The increased wind speed is, in most cases, too high and out of reach for boundary layer wind tunnels [11]. Former research shows that bridge decks with sharp edge bodies are less sensitive to change of Reynolds number [12]. This is because the separation point that controls the action of the aerodynamic forces generally occurs at the leading edge, except for a very large angle of attack.

3.3.4 Strouhal Number

The Strouhal number is a dimensionless number and is often used to describe vortex shedding. It is defined as [13]:

$$
S_t = \frac{f_s L}{V} \tag{3.32}
$$

where f_s is the Strouhal frequency or the vortex shedding frequency as described in Section 3.2.4, V is the mean wind velocity and L is the characteristic length.

The Strouhal number is important when analyzing unsteady oscillating flow problems. It represents the ratio between the inertial forces due to the local acceleration of the flow and the inertial forces due to the convective acceleration. In order to scale the frequency, length, time and wind speed, it is necessary that the Strouhal number is equal in full scale as in the wind tunnel [13]:

$$
St_{WT} = St_{FS} \quad \text{or} \quad \frac{T_{WT}}{T_{FS}} = \frac{L_{WT}}{L_{FS}} \frac{V_{FS}}{V_{WT}} \tag{3.33}
$$

If the gap between to girders increases, it will cause the Strouhal number to increase because of the change in flow regime around the bridge deck. This is essential knowledge for full aerodynamic evaluation of a twin deck [14]. The Strouhal number does also display a significant dependence on the Reynolds number as it increases gradually with increasing Re, as shown in a study by Schewe and Larsen [15].

3.3.5 Froude Number

The Froude number is defined as the ratio of the inertia forces to the gravity forces, given by:

$$
F_r = \frac{V^2}{gL} \qquad \frac{V_{WT}}{V_{FS}} = \sqrt{\frac{L_{WT}}{L_{FS}}} \tag{3.34}
$$

where V is the mean wind velocity, g is the acceleration of gravity and L is the characteristic length.

As seen in Equation 3.34, the velocity scale is equal to the square root of the geometric sale since the acceleration of gravity is equal for both the wind tunnel and the full scale. This can cause a problem since the wind tunnel test must be conducted at a low wind velocity and wind tunnels are often less accurate at low wind velocities. For long-span bridges, the gravitational force is important. Therefore, Froude number similarity for the wind tunnel model and the full-scale model should be respected [11].

3.3.6 Wind Turbulence

Wind turbulence can be expressed in the turbulence spectra and has two parameters; the turbulence intensity and the integral length scale. Turbulence intensity is a non-dimensional property and measures the turbulence relative to the mean wind velocity. In other words, it is the standard deviation of the wind speed divided by average wind speed over a period of typically 10 minutes [7]:

$$I_n = \frac{\sigma_n}{V} \qquad \text{where} \quad n = u, v, w \tag{3.35}$$

The auto covariance functions and corresponding auto covariance coefficients of a wind signal, where τ is an arbitrary time lag are expressed by [7]:

$$Cov_n(\tau) = E[n(t)n(t+\tau] = \frac{1}{T}\int_0^T n(t)n(t+\tau)dt \qquad \text{where} \quad n = u, v, w \tag{3.36}$$

$$\rho_n(\tau) = \frac{Cov_n(\tau)}{\sigma_n^2} \qquad \text{where} \quad n = u, v, w \tag{3.37}$$

The auto covariance of the turbulence components diminish at increasing values of the time lag, τ, and at large values of τ they approach zero.

The time scale can be interpreted as the average duration of a u, v or w wind gust:

$$T_n = \int_0^\infty \rho_n(\tau)d\tau \qquad \text{where} \quad n = u, v, w \tag{3.38}$$

A well-known assumption in wind turbulence is Taylor's frozen turbulence hypothesis, which assumes that the turbulence is "frozen". It is based upon that the advection velocity of the turbulence is much larger than the velocity scale of the turbulence itself [16]. That is, the eddy property is not changing during advection, and all eddies are advected at the mean wind velocity. Taylor [17] himself stated:

> *If the velocity of the air stream which carries the eddies is very much greater than the turbulent velocity, one may assume that the sequence of changes in u at the fixed point are simply due to the passage of an unchanging pattern of turbulent motion over the point.*

Adopting Taylor's hypothesis, the turbulence convection in the main flow direction takes place with the mean wind velocity. The average length scales are given by:

$$L_n = V \int_0^\infty \rho_n(\tau) d\tau \qquad \text{where} \qquad n = u, v, w \tag{3.39}$$

These length scales can be interpreted as the average eddy size of u, v and w components in the main flow direction [7]. It is the product of average velocity and integral time scale. It represents the correlation length and is scaled down by the geometric scale, and consists of eddy sizes in meters. The integration length is divided into three different directions; the longitudinal component L_u, the crosswind component L_v and the vertical turbulence component L_w. A geometrical scale is used when scaling down the integral length scale, but some predictions are necessary when determining the integral length scale in a wind tunnel.

When studying wind loading, a lower turbulence intensity level than required will usually result in higher loads. This is considered as the conservative case of wind loading. Therefore, the turbulence intensity in the wind tunnel must be smaller or equal to the value in a full-scale test [13]:

$$I_{u,WT} \leq I_{u,FS} \tag{3.40}$$

3.4 Aerodynamic Admittance Functions

For wind-induced vibration of long-span bridges, buffeting is a central research area. Low wind speed causes buffeting, and the effects can result in fatigue or serviceability issues and are therefore important with the increasing span. The Aerodynamic Admittance Functions (AAF) is an important transfer function for estimating the buffeting response, which transfers the turbulent wind to buffeting forces. Sears (1941) first proposed a theoretical approach for the AAF on streamlined bodies in the frequency domain. Later, Davenport (1962) extended the theory to buffeting analysis of bridges based on the quasi-steady theory. To verify Sears's function for a thin airfoil, Lamson (1966) was the first to carry out the identification of the aerodynamic admittance functions in a wind tunnel [18].

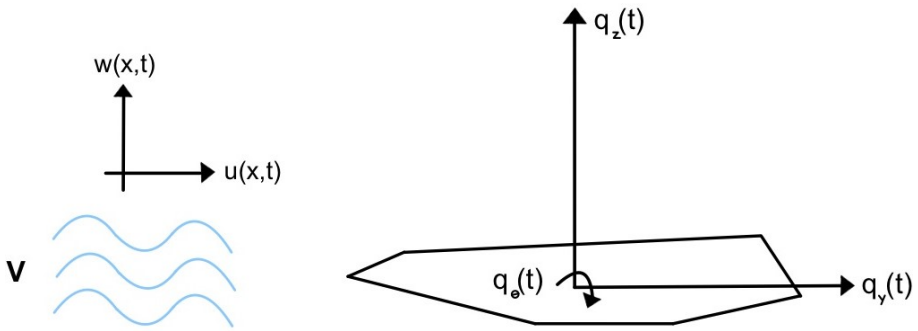


Figure 3.3: Wind action on a bridge girder.

Figure 3.3 illustrates a bridge girder subjected to a 2D wind flow and the forces and moment in the coordinates system of the fluctuating wind. The wind velocity is divided into the stationary wind speed, V and the turbulent vector $\boldsymbol{v}$ containing the fluctuating terms u and w. The linearized buffeting load due to turbulence can be expressed by:

$$\boldsymbol{q}_b(x,t) = \boldsymbol{B}_q(t)\boldsymbol{v}(x,t) \tag{3.41}$$

$\boldsymbol{B}_q\boldsymbol{v}$ is the dynamic loading associated with the turbulence, where $\boldsymbol{B}_q$ is the same as in Equation 3.21 and $\boldsymbol{v}$ is the same as in Equation 3.18.

Similarly, the buffeting load $\boldsymbol{Q}_b$ may be obtained by taking the Fourier transform of Equation 3.41:

$$\boldsymbol{Q}_b(x,\omega) = \boldsymbol{B}_q(\omega)\boldsymbol{v}(x,\omega) \tag{3.42}$$

where

$$\mathbf{B}_q(x,\omega) = \frac{1}{2}\rho V B \begin{bmatrix} 2(D/B)\bar{C}_D\chi_{yu} & ((D/B)C'_D - \bar{C}_L)\chi_{yw} \\ 2\bar{C}_L\chi_{zu} & (C'_L + (D/B)\bar{C}_D)\chi_{zw} \\ 2B\bar{C}_M\chi_{\theta u} & B\bar{C}_M\chi_{\theta w} \end{bmatrix} \tag{3.43}$$

and the frequency dependent admittance functions characteristic to the cross section [7]:

$$\chi_{mn}(\omega) \qquad \begin{cases} m = y, z, \omega \\ n = u, w \end{cases} \tag{3.44}$$

Besides Taylor's frozen turbulence hypothesis described in Section 3.3.6, Taylor also stated how the correlation between two points decreases slower for large eddies than for smaller eddies. In the buffeting theory described in Section 3.2.2, the bridge deck is considered slender with respect to the longitudinal axis, and the quasi-steady theory holds. However, this holds only if the characteristic size of the section is small compared with the turbulence length scale and is accurate for low frequencies [19].

Bridge decks are often very elongated, and the characteristic size of the section cannot be considered small. Therefore, the turbulent eddies cannot be considered as perfectly correlated around the body. In Figure 3.4 the mean pressure is represented by vectors for both a) low frequencies and for b) high frequencies. For low frequencies, the mean pressure is highly correlated because of the slow wavelengths, and the quasi-steady theory is valid. However, for high frequencies, i.e., rapid varying eddies, the bridge experience various fields of turbulence, which results in reduced correlation in the pressure distribution. The aerodynamic admittance functions consider this lack of correlation between the velocity fluctuation in the region surrounding the bridge [19].

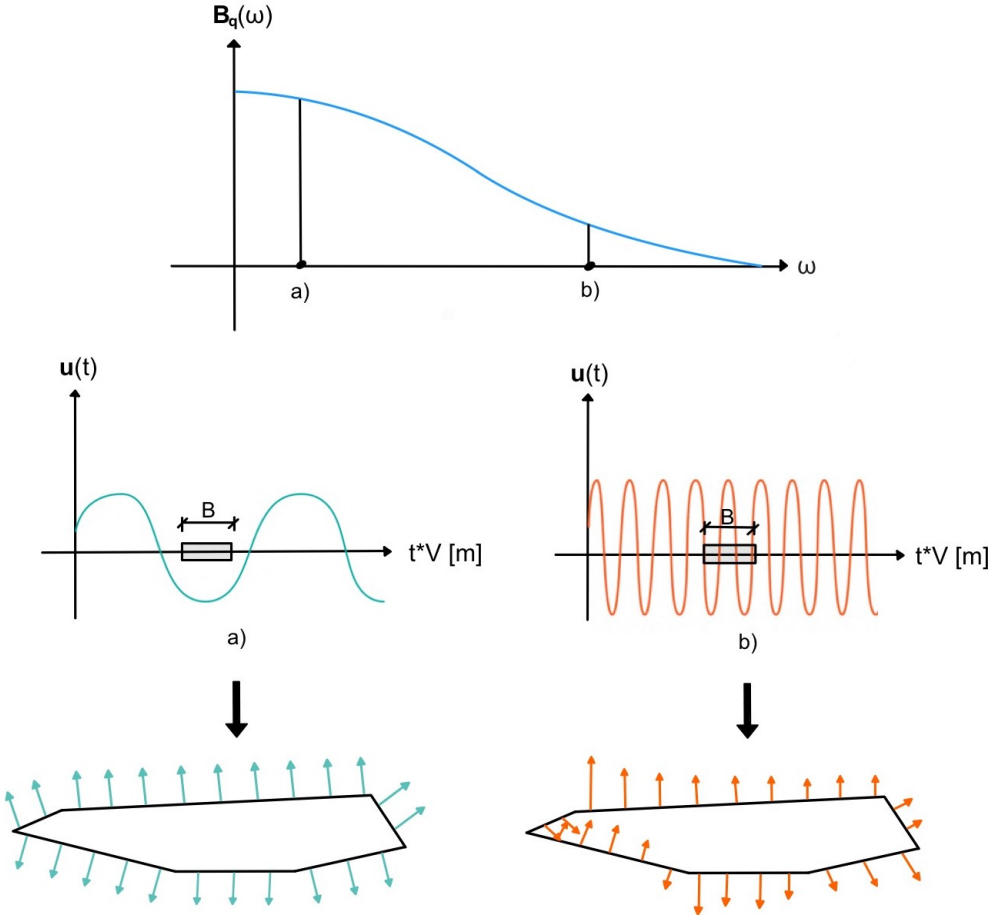


Figure 3.4: The mean pressure around the periphery of a bridge deck for a) low frequency and b) high frequency.

The cross sectional admittance functions may be determined from wind tunnel experiments with time series of drag, lift and moment forces. Another method is by pressure tap measurements around the periphery of the cross section, which will be done in this thesis.

3.4.1 Theoretical Aerodynamic Admittance

A method to obtain the buffeting response is through theoretical calculation. This is often based on the principle of aerodynamics to provide a mathematical model of the relevant wind load before the structural dynamics method is applied to solve the wind-induced response of the structure. The current theoretical model is based on pioneering work done by Davenport, among others, which has been enhanced in several studies [20].

Sears Function

Sears investigated the forces on a thin airfoil due to a sinusoidal coherent gust. The analysis of the unsteady lift force is based on the strip assumption, as described in Section 3.2.1. From linearized equations of fluid motion and the Kutta-Jouknowski condition, Sears introduced the lift force spectrum by:

$$S_L(f^*) = 4\pi^2 |\phi(f^*)|^2 S_w(f^*) \tag{3.45}$$

where Kutta-Jouknowski condition says that there are no singularities at the rear end of the airfoil, f^* is the reduced frequency expressed by $f^* = fB/2$, B is the deck width and $|\phi(f^*)|$ is the Sears function given by [21]:

$$|\phi(f^*)|^2 = \left| \frac{J_0(f^*)K_1(if^*) + iJ_1(f^*)K_0(if^*)}{K_1(if^*) + K_0(if^*)} \right|^2 \tag{3.46}$$

where J_0 and J_1 are Bessel functions of the first kind, while K_0 and K_1 are modified Bessel functions of the second kind. The Sears function is often approximated by an expression suggested by Leipmann:

$$|\phi(f^*)|^2 \approx \frac{1}{1 + 2\pi^2 f^*} \tag{3.47}$$

Equation 3.45 and 3.47 shows that the lift forces reduces as f^* increases. Sears demonstrated that for any flow fluctuation with a finite wavelength, the fluctuating lift will be less than the quasi-steady value [21].

The aerodynamic admittance may be defined as:

$$\chi(f) = \frac{S_L(f)}{C_z'^2 S_\omega(f)} \tag{3.48}$$

Based on this definition, $\chi(f) = 1.0$ for the quasi-steady case. Equivalent to the strip assumption, for a thin airfoil in a fully correlated gust, the aerodynamic admittance can be expressed by:

$$\chi(f) = |\phi(f^*)|^2 \tag{3.49}$$

The Sears function, proposed by Liepmann, is often used in bridge aerodynamics to represent the aerodynamic admittance. However, for streamlined bridge decks in turbulent flow, experiments have shown that the admittance functions are significantly different from the Sears function. It might be an acceptable assumption at low frequencies. However, for high frequencies where the

turbulence length scales are comparable to the thickness, the flat plate assumption is invalid and leads to an overestimation of the lift [22]. Experiments with section models of streamlined bridge decks in turbulent flow have shown that the admittance tends to be lower than the Sears function for low frequencies, and for bluff bridge decks, it tends to be higher for high frequencies [21].

Davenport's Buffeting Theory

The theoretical analysis of buffeting forces on long-span bridges began when Davenport introduced aerodynamic admittance in the 1960s. Based on the theory of aerodynamics, Davenport defined the joint acceptance function and considered the time and space distribution of the aerodynamic forces on a cross-section. It was introduced to express that the wind loading may vary with frequency and is not necessarily quasi-steady. Moreover, to represent the spatial variation in the flow over the region and impact on the forces [20].

The wind loads due to the buffeting action are as described in Section 3.2.2 and are given by:

$$F_{x,b} = \frac{\rho V B}{2} \left[2C_x u + C'_x w\right] \tag{3.50a}$$

$$F_{z,b} = \frac{\rho V B}{2} \left[2C_z u + C'_z w\right] \tag{3.50b}$$

$$M_{\theta,b} = \frac{\rho V B^2}{2} \left[2C_m u + C'_m w\right] \tag{3.50c}$$

From now on, only the lift components will be considered for simplicity. By assuming the buffeting loading is a stationary random process, the lift load can be transformed to the frequency domain by the Fourier transform [21]:

$$S_L(f^*) = \left(\frac{\rho V B}{2}\right)^2 \left(4C_z^2 S_u(f^*)|\chi_{u,z}(f^*)|^2 + C_z'^2 S_w(f^*)|\chi_{w,z}(f^*)|^2\right) \tag{3.51}$$

where $S_L(f^*)$ is the spectrum of the lift force per unit length on a cross-sectional strip of the deck, $S_{u,w}$ are the spectral densities of the u and w components of the wind, respectively, and $|\chi_{u,w;z}(f^*)|^2$ is the lift aerodynamic admittance due to the u and w components of the turbulence.

It is troublesome to distinguish between the effects of u and w in experiments. Therefore, the admittances are generally lumped, and the lift load from Equation 3.51 can be expressed by [21]:

$$S_L(f^*) = \left(\frac{\rho V B}{2}\right)^2 |\chi_z(f^*)|^2 \left(4C_z^2 S_u(f^*) + C_z'^2 S_w(f^*)\right) \tag{3.52}$$

Further, via the joint acceptance function $J_z(f_j^*)$, the point-like load is made into a line-like load on a span with length l:

$$S_{F_z}(f_j^*) = S_L(f^*)|J_z(f_j^*)|^2 \tag{3.53}$$

where

$$|J_z(f_j^*)|^2 = \int_0^L \int_0^L \frac{S_{L_1 L_2}(\Delta y, f^*)}{S_L(f^*)} \mu_j(y_1)\mu_j(y_2) dy_1 dy_2 \tag{3.54}$$

$S_{L_1L_2}$ is the cross-spectrum of the lift force between strip 1 and 2 separated by Δy and μ_j is the j^{th} mode shape, respectively.

The buffeting forces are not fully correlated span-wise. In addition, the effect on the structure from the gust loading pattern will be different for the various mode of vibration. The joint acceptance function takes this into account by measuring the correlation between the spatial distribution of the forces across the span and the mode. Every mode of vibration to the bridge deck has one joint acceptance function.

Under the basis of the strip assumption, the cross-spectrum can be expressed by:

$$\frac{S_{L_1L_2}(\Delta y, f^*)}{S_L(f^*)} \approx \frac{S_{\omega_1\omega_2}(\Delta y, f^*)}{S_\omega(f^*)} = coh_\omega^{1/2}(\Delta y, f^*) \tag{3.55}$$

Furthermore, the spectrum of the response of a given mode, j, to the buffeting force can be determined by:

$$S_{r_z}(f_j^*) = S_{F_z}(f_j^*)|H(f_j^*)|^2 \tag{3.56}$$

where $H(f_j^*)$ is the single degree-of-freedom mechanical admittance function of mode j and can be expressed by:

$$\left|H(f_j^*)\right|^2 = \frac{1}{\left(1-\left(\frac{f^*}{f_j^*}\right)^2\right)^2 + \left(2\left(\zeta_{s,j}+\zeta_{a,j}\right)\frac{f^*}{f_j^*}\right)^2} \tag{3.57}$$

$H(f_j^*)$ is a function of reduced frequency and damping. The influence from the aerodynamic forces is represented by adding the contribution of the aerodynamic damping, ζ_a to the structural damping, ζ_s. The frequency term can also be corrected by including the influence of the aerodynamic stiffness. This has, however, a negligible influence on the buffeting response and is not done here.

The dynamic response can be divided into the background and resonant components. Due to the slow variation of wind speeds, the background response acts quasi-statically. The background response covers a wide frequency band below the lowest natural frequency, while the resonant response is concentrated in a peak at the natural frequency. The contribution of the w component of the turbulence to the expression of the background and resonant components are for the vertical force defined by [21]:

$$\sigma_{B_z}^2 = \left(\frac{\rho\bar{V}^2BC_z'}{2}\right)^2\left(\frac{\sigma_w}{\bar{V}}\right)^2\int_0^\infty \frac{f^*S_\omega(f^*)}{\sigma_\omega^2}\left|\chi_z(f^*)\right|^2\left|J_z(f^*)\right|^2 \mathrm{d}\,\ln f^* \tag{3.58}$$

$$\sigma_{R_{z_j}}^2 \approx \left(\frac{\rho\bar{V}^2BC_z'}{2}\right)^2\left(\frac{\sigma_w}{\bar{V}}\right)^2\frac{f^*S_\omega(f^*)}{\sigma_\omega^2}\left|\chi_z(f^*)\right|^2\left|J_z(f_j^*)\right|^2\frac{(\pi/4)}{\left(\zeta_{s_j}+\zeta_a(f_j^*)\right)} \tag{3.59}$$

The peak response can be expressed by:

$$\hat{r} = \bar{r} + g\sqrt{\sigma_B^2 + \sum \sigma_{R_j}^2} \tag{3.60}$$

where $\bar{r}$ is the mean response and g is a statistical peak factor which for the buffeting response typically have a value between 3 and 4. σ_B^2 is the mean square background response and $\sigma_{R_j}^2$ is the mean square modal response at or close to the j^{th} resonant frequency.

With Equation 3.59, Davenport defined the aerodynamic admittance functions by [11]:

$$|\chi_z(f^*)|^2 = \frac{S_F(f^*)}{\frac{1}{4}\rho^2 \bar{V}^2 B^2 C_z'^2 S_\omega(f)} \tag{3.61}$$

3.4.2 3D Aerodynamic Admittance Functions

The majority of previous studies of aerodynamic admittance are based on the strip assumption employed by Sears and Davenport. The Sears function only considers the variation of the vertical fluctuation in a two-dimensional wind field and can therefore be expressed as a 2D AAF of an airfoil. The influence of the incident turbulence characteristics, especially the turbulence length scale, is neglected when looking into these assumptions. The turbulent length scale represents the average size of the most energetic turbulent eddies. In 1955 Liepmann introduced a two-wavenumber aerodynamic admittance function to consider the spanwise variations to investigate the three-dimensional effect of turbulence on an airfoil. This was defined as the 3D AAF. Later in 1970, Graham developed the exact numerical solution of the 3D AAF for the lift of a thin airfoil, which was experimentally validated by Jackson et al. [23] in 1973 and by Li et al. [24] in 2015.

Graham did not provide an explicit expression for the 3D AAF; therefore, three different expressions by Mugridge, Filotas, and Blake are defined below [24].

Mugridge's 3D AAF

$$|\chi(k_1,k_2)|^2 \approx \frac{1}{1+2\pi\tilde{k}_1}|F(\tilde{k}_1,\tilde{k}_2)|^2 \tag{3.62}$$

k_1 and k_2 is the chordwise and spanwise wavenumbers, $k_{1,2} = n/U$ (n is the frequency in Hz and U is the mean wind velocity), $\tilde{k_1} = 2\pi k_1 B/2$ and $\tilde{k_2} = 2\pi k_2 B/2$. The correlation function is defined as:

$$|F(k_1,k_2)|^2 = \left[\frac{\tilde{k}_1^2 + 2/\pi^2}{\tilde{k}_1^2 + \tilde{k}_2^+ 2/\pi^2}\right] \tag{3.63}$$

The equation by Mugridge is an approximate closed-form expression for the lift aerodynamic admittance in terms of a correlation factor to the traditional Sears function. This expression is of high accuracy for the lower wavenumber range $k_1 < 1/\pi$, when compared to Graham's exact result [24].

Filotas' 3D AAF

$$|\chi(k_1,k_2)|^2 = \frac{\sqrt{\tilde{k}_1^2+\tilde{k}_2^2}}{\sqrt{\tilde{k}_1^2+\tilde{k}_2^2}+\pi\left(\pi\tilde{k}_2^3+\tilde{k}_2^2+\pi\tilde{k}_1\tilde{k}_2+2\tilde{k}_1^2\right)} \tag{3.64}$$

The approximate expression by Filotas is based on linearized incompressible lifting surface theory. For limiting cases where the reduced frequency is either very small or very large, this expression is asymptotically exact.

Blake's 3D AAF

$$|\chi(k_1,k_2)|^2 = \frac{1}{1+2\pi\tilde{k}_1}\left[\frac{1+3.2(2\tilde{k}_1)^{1/2}}{1+2.4(2\tilde{k}_1)^2+3.2(2\tilde{k}_1)^{1/2}}\right] \tag{3.65}$$

This expression by Blake is for use in approximations. It is a closed-form expression fitting Graham's exact solution. When $\tilde{k}_1 > \tilde{k}_2/2$, the approximation agreed with Graham's exact values to within 20%.

3.4.3 Experimental Identification of the Aerodynamic Admittance

In this subsection, several studies for estimation of the aerodynamic admittance functions will be presented; The auto-spectrum and the cross-spectrum method, The Taut Strip Model Approach, The Colligated Residue Least Square Method of Auto and Cross Spectra (CRLSMACS) and The Six Complex Aerodynamic Admittance Functions.

The Auto-Spectrum Method (ASM) and the Cross-Spectrum Method (CSM)

One approach for AAF identification is the auto-spectrum method (ASM), also called the equivalent AAF method. It is based on the measured auto-spectrum of buffeting forces and assume that the admittance of a buffeting force due to the longitudinal fluctuating velocity, u, is equivalent to the force due to the vertical fluctuating velocity, w, i.e.:

$$\chi_{Fu} = \chi_{Fw} = \chi_F \qquad \text{where} \qquad F = L, D, M \tag{3.66}$$

This assumption is made because the derivative of the static wind coefficients for lift and moment (C'_L, C'_M), is much larger than C_L and C_M. The vertical component, w, has therefore a major impact in the buffeting force, and the horizontal component, u, can be neglected. In the frequency domain, the modulus squared value of the equivalent AAFs for each force, $|\chi_F|^2$, may be obtained to ensure that the reproduced force auto-spectrum is equivalent to the tested or the real force auto-spectrum. $|\chi_F|^2$ is a weighted average of $|\chi_{Fu}|^2$ and $|\chi_{Fw}|^2$. For a typical bridge, $|\chi_F|^2$ is usually close to $|\chi_{Dw}|^2$, $|\chi_{Lw}|^2$ or $|\chi_{Mw}|^2$ because these are often significantly larger than those of the other AAFs. The AAF for the lift buffeting force can be expressed as [25]:

$$|\chi_L(\omega)|^2 = \frac{S_L(\omega)}{\left(\frac{\rho U B}{2}\right)^2 \left[4C_L^2 S_u(\omega) + (C'_L + C_D)^2 S_w(\omega)\right]} \tag{3.67}$$

To distinguish χ_{Fu} and χ_{Fw}, a cross-spectrum method was adopted by researchers such as Ma et al. [26] in 2013 and Zhao and Ge [27] in 2015. The method is based on the measured cross-spectra between the fluctuating force coefficient C_F and each of u and w. The cross-spectral equations for solving AAFs are defined by [28]:

$$S_{C_F u} = a_F \chi^*_{Fu} S_u + b_F \chi^*_{Fw} S_{wu} \tag{3.68a}$$

$$S_{C_F w} = a_F \chi^*_{Fu} S_{uw} + b_F \chi^*_{Fw} S_w \tag{3.68b}$$

where

$$a_F = 2C_F(\theta_0)/U \tag{3.69}$$

$$b_F = \begin{cases} [C'_D - C_L], & \text{F = D} \\ [C'_L + C_D], & \text{F = L} \\ [C'_M], & \text{F = M} \end{cases} \tag{3.70}$$

where C_F and C'_F are the aerodynamic force coefficient and its derivative with respect to the angle of attack, where $F = D, L, M$, U is the mean wind velocity and θ_0 is the initial angle of attack of the mean wind.

The two AAF components may be estimated with the following expressions:

$$\chi^*_{Fu} = \frac{S_w S_{C_F u} - S_{wu} S_{C_F w}}{a_F \left(S_u S_w - S_{wu} S_{uw}\right)} \tag{3.71a}$$

$$\chi^*_{Fw} = \frac{S_u S_{C_F w} - S_{uw} S_{C_F u}}{b_F \left(S_u S_w - S_{wu} S_{uw}\right)} \tag{3.71b}$$

where $S_{C_{Fa}} (a = u, w)$ are the cross-spectral densities of the buffeting force coefficient C_F and the fluctuating wind velocity component a.

However, since the correlation between the buffeting force and the fluctuating wind is often quite weak, the identified AFFs show rather strong random behaviour. Thus, the auto-spectra of the fluctuating force reproduced using the identified AAFs usually deviate accordingly [25].

Larose - The Response of a Suspension Bridge Deck to Turbulent Wind: the Taut Strip Model Approach

Larose studied the taut strip model approach to estimate the response of long-span bridges to turbulent wind. Among other things, the research included measurements of the aerodynamic admittance and the span-wise cross-correlation of the aerodynamic forces in a smooth flow with grid-generated turbulence and turbulent boundary layer flow.

From the admittance functions developed by Davenport, Larose [11] included the influence of the vertical (w) and the longitudinal (u) components of turbulence on the fluctuating lift force. These can be expressed as follow:

$$|\chi_{Lu}(f^*)|^2 = \frac{U}{C_z(0)\frac{1}{2}\rho U^2 B} \left[\frac{S_{Lu}(f^*)S_w(f^*) - S_{Lw}(f^*)S_{wu}(f^*)}{S_u(f^*)S_w(f^*) - S_{uw}(f^*)S_{wu}(f^*)}\right] \tag{3.72}$$

$$|\chi_{Lw}(f^*)|^2 = \frac{U}{\frac{dC_z}{d\alpha}\frac{1}{2}\rho U^2 B} \left[\frac{S_{Lw}(f^*)S_u(f^*) - S_{Lu}(f^*)S_{uw}(f^*)}{S_u(f^*)S_w(f^*) - S_{uw}(f^*)S_{wu}(f^*)}\right] \tag{3.73}$$

and for moment:

$$|\chi_{Mu}(f^*)|^2 = \frac{U}{C_m(0)\frac{1}{2}\rho U^2 B} \left[\frac{S_{Mu}(f^*)S_w(f^*) - S_{Mw}(f^*)S_{wu}(f^*)}{S_u(f^*)S_w(f^*) - S_{uw}(f^*)S_{wu}(f^*)}\right] \tag{3.74}$$

$$|\chi_{Mw}(f^*)|^2 = \frac{U}{\frac{dC_m}{d\alpha}\frac{1}{2}\rho U^2 B} \left[\frac{S_{Mw}(f^*)S_u(f^*) - S_{Mu}(f^*)S_{uw}(f^*)}{S_u(f^*)S_w(f^*) - S_{uw}(f^*)S_{wu}(f^*)}\right] \tag{3.75}$$

For a boundary layer flow, Larose observed that the Sears function overestimate the aerodynamic admittance for low reduced frequencies, while it underestimate at high frequencies. Another observation was that the span-wise cross-correlation of the aerodynamic forces on the deck was larger than the span-wise cross-correlation of the oncoming wind velocity fluctuations. Which for the used cross-section, suggests that the strip assumption is not valid. In addition, the bridge extracted more energy from the larger scales turbulence of the boundary layer flow at lower reduced

frequency, than from the small scale turbulence from a 2D-grid. The opposite was observed for higher reduced frequency. This suggests that the measurements of the aerodynamic admittance should be estimated with adequately scaled turbulence in relation with the size of the model [11].

Six Complex Aerodynamic Admittance Functions

In 2010 Han et al.[29] described a new frequency-by-frequency methodology for estimation of six complex aerodynamic admittance functions. The Sears function is a complex theoretical expression for the aerodynamic admittance function for a thin airfoil. The aerodynamic admittance function for a bridge deck should, therefore, also be complex functions. To measure all the six complex AAF, an active turbulence generator was developed. Wind tunnel tests of a thin plate model and a streamlined bridge section were conducted in a turbulent flow. The six complex aerodynamic admittance functions were determined by the developed methodology and compared with the Sears function and Davenport's formula.

The six complex aerodynamic admittance functions are derived theoretically; Six complex aerodynamic functions are derived from the aerodynamic lift force, drag force, and pitching moment when exposed to time-varying harmonic turbulent wind components, $u(t)$ and $w(t)$. Further, the six complex aerodynamic admittance functions are found by taking the FFT of the aerodynamic forces and the harmonic turbulent wind components. The complex, AAF χ_{Lu}, χ_{Du} and χ_{Mu} corresponding to the longitudinal turbulent component $u(t)$ are defined by:

$$\chi_{Lu}(\omega_1) = \frac{L_b(\omega_1)}{\frac{1}{2}\rho U^2 B D C_L \frac{2}{U} \cdot \frac{A_u T}{2} e^{i\phi_1}} \tag{3.76a}$$

$$\chi_{Du}(\omega_1) = \frac{D_b(\omega_1)}{\frac{1}{2}\rho U^2 B D C_D \frac{2}{U} \cdot \frac{A_u T}{2} e^{i\phi_1}} \tag{3.76b}$$

$$\chi_{Mu}(\omega_1) = \frac{M_b(\omega_1)}{\frac{1}{2}\rho U^2 B^2 D C_M \frac{2}{U} \cdot \frac{A_u T}{2} e^{i\phi_1}} \tag{3.76c}$$

Further, the complex AAF χ_{Lw}, χ_{Dw} and χ_{Mw} corresponding to the vertical turbulent component $w(t)$ are be defined by:

$$\chi_{Lw}(\omega_2) = \frac{L_b(\omega_2)}{\frac{1}{2}\rho U^2 B D \frac{(C'_L + C_D)}{U} \cdot \frac{B_w T}{2} e^{i\phi_2}} \tag{3.77a}$$

$$\chi_{Dw}(\omega_2) = \frac{D_b(\omega_2)}{\frac{1}{2}\rho U^2 B D \frac{(C'_D - C_L)}{U} \cdot \frac{B_w T}{2} e^{i\phi_2}} \tag{3.77b}$$

$$\chi_{Mw}(\omega_2) = \frac{M_b(\omega_2)}{\frac{1}{2}\rho U^2 B^2 D \frac{C'_M}{U} \cdot \frac{B_w T}{2} e^{i\phi_2}} \tag{3.77c}$$

where ω_1 and ω_2 are vibration circular frequency of $u(t)$ and $w(t)$, $\omega_1 \neq \omega_2$. L_b, D_b and M_b are the FFT of the lift force, drag force and pitching moment, ρ is the air density, U is the mean longitudinal wind velocity and B is the deck width. C_D, C_L and C_M are the drag force, lift force and pitching moment coefficients, while C'_D, C'_L and C'_M are the associated derivatives with respect to the angle of attack. Further, A_u and B_w are the amplitude of the harmonic functions $u(t)$ and $w(t)$, T is the total duration and ϕ_1 and ϕ_2 are initial phase angle of $u(t)$ and $w(t)$, respectively.

For a bridge deck with a length D, the aerodynamic lift force, drag force and pitching moment can be expressed by:

$$L_b(t) = \frac{1}{2}\rho U^2 BD\left[2C_L\chi_{Lu}\frac{u(t)}{U} + (C'_L + C_D)\chi_{Lw}\frac{w(t)}{U}\right] \tag{3.78a}$$

$$D_b(t) = \frac{1}{2}\rho U^2 BD\left[2C_D\chi_{Du}\frac{u(t)}{U} + (C'_D - C_L)\chi_{Dw}\frac{w(t)}{U}\right] \tag{3.78b}$$

$$M_b(t) = \frac{1}{2}\rho U^2 B^2 D\left[2C_M\chi_{Mu}\frac{u(t)}{U} + C'_M\chi_{Mw}\frac{w(t)}{U}\right] \tag{3.78c}$$

where χ_{Rk} (R = L, D, M; k = u, w) are aerodynamic admittance functions.

Some conclusions from the study were [29]:

- Drag-force admittance functions and admittance functions corresponding to the longitudinal component deviate significantly from the Sears function.
- The admittance functions corresponding to the longitudinal component are different from those corresponding to the vertical component. Thus, it is necessary to estimate all the six admittance functions.
- With the increase of the reduced frequency, some of the identified aerodynamic admittance functions increase.
- Similar to the Sears functions, some of the phases of the estimated admittance functions increase with the increase of the reduced frequency.

Colligated Residue Least Square Method of Auto and Cross Spectra (CRLSMACS)

The above methods for AAF identification have several shortcomings. To overcome these, Zhu et al. [30] presented in 2017 a new method called the colligated residue least square method of auto and cross spectra (CRLSMACS). The method identifies six-component AAFs which is based on force and pressure measurements tests in a passive grid-generated turbulence flow.

The buffeting forces can be expressed by an equation set consisting of six equations. Each expression is a function of the auto and cross-spectra of the fluctuating wind, Su, Sw and Suw, and the AAFs between the distributed buffeting force and the fluctuating wind velocity. By this, the colligated spectral residue functions are obtained. The residual function for the drag force is defined by:

$$R_L(\chi_{Lu}^{Re}, \chi_{Lu}^{Im}, \chi_{Lw}^{Re}, \chi_{Lw}^{Im}) = w_1\varepsilon_{LL}^2 + w_2\left[\left(\varepsilon_{Lu}^{Re}\right)^2 + \left(\varepsilon_{Lu}^{Im}\right)^2\right] + w_3\left[\left(\varepsilon_{Lw}^{Re}\right)^2 + \left(\varepsilon_{Lw}^{Im}\right)^2\right] \tag{3.79a}$$

$$\begin{aligned}\varepsilon_{LL} = {} & 0.25(\rho UB)^2\{4C_L^2|\chi_{Lu}|^2\hat{S}_{uu} + (C_D + C'_L)^2|\chi_{Lw}|^2\hat{S}_{ww} + 4C_L(C_D + C'_L) \\ & \times\left[\left(\chi_{Lu}^{Re}\chi_{Lw}^{Re} + \chi_{Lu}^{Im}\chi_{Lw}^{Im}\right)\hat{S}_{uw}^{Re} - \left(\chi_{Lw}^{Re}\chi_{Lu}^{Im} - \chi_{Lw}^{Im}\chi_{Lu}^{Re}\right)\hat{S}_{uw}^{Im}\right]\} - \hat{S_L}\end{aligned} \tag{3.79b}$$

$$\varepsilon_{Lu}^{Re} = 0.5\rho UB\left[2C_L\chi_{Lu}^{Re}\hat{S}_{uu} + (C_D + C'_L)\left(\chi_{Lw}^{Re}S_{wu}^{Re} + \chi_{Lw}^{Im}S_{wu}^{Im}\right)\right] - \hat{S}_{Lu}^{Re} \tag{3.79c}$$

$$\varepsilon_{Lu}^{Im} = 0.5\rho UB\left[-2C_L\chi_{Lu}^{Im}\hat{S}_{uu} + (C_D + C_L')\left(\chi_{Lw}^{Re}S_{wu}^{Im} + \chi_{Lw}^{Im}S_{wu}^{Re}\right)\right] - \hat{S}_{Lu}^{Im} \tag{3.79d}$$

$$\varepsilon_{Lw}^{Re} = 0.5\rho UB\left[2C_L\left(\chi_{Lu}^{Re}\hat{S}_{uw}^{Re} + \chi_{Lu}^{Im}\hat{S}_{uw}^{Im}\right) + (C_D + C_L')\chi_{Lw}^{Re}\hat{S}_{ww}\right] - \hat{S}_{Lw}^{Re} \tag{3.79e}$$

$$\varepsilon_{Lw}^{Re} = 0.5\rho UB\left[2C_L\left(\chi_{Lu}^{Re}\hat{S}_{uw}^{Im} - \chi_{Lu}^{Im}\hat{S}_{uw}^{Re}\right) - (C_D + C_L')\chi_{Lw}^{Im}\hat{S}_{ww}\right] - \hat{S}_{Lw}^{Im} \tag{3.79f}$$

where $w_i (i = 1, 2, 3)$ are weighting factors, "Re" and "Im" represent the real part and imaginary part of the corresponding aerodynamic admittance or the cross spectra, respectively. The variables marked with "ˆ" have measured values inserted. Further, the real and imaginary parts of the six-component complex aerodynamic admittances can be expressed by seeking the minimal values of the residues defined above.

Traditionally, the measured buffeting force from wind tunnel tests is the total force acting on the whole model, $F(t)$. Hence, the distributed force acting on cross-section strips, $f(t, x)$, is equal to $F(t)/l$, where l is the length of the measured section. This implies that it is full correlation along the longitudinal axes of the cross-section. However, the buffeting force on the cross-section in a turbulent flow is partially correlated along the longitudinal axes, and $f(x, t)$ is larger than $F(t)/l$. CRLSMACS corrects the incomplete span-wise correlation of the buffeting forces. By measurements from pressure tubes arranged on the model and a Cobra Probe, a span-wise correction function is obtained, and the auto-spectra of the distributed buffeting forces.

With the presented method, Zhu et al.[30] identified the six-complete AFFs of a flat closed-box deck of a single tower cable-stayed bridge. The results showed that $|\chi_{Lw}|$ and $|\chi_{Mw}|$ of the flat closed-box deck are close to each other and to the Sears function, which is reasonable and, to some extent, demonstrates the reliability of CRLSMACS. Nonetheless, the other components of AAFs deviated significantly from the Sears function. This is expected since the Sears function can only reasonable represents $|\chi_{Lw}|$ and $|\chi_{Mw}|$. Further, the calculated buffeting response was compared with a full bridge aeroelastic model tests. The results agreed well and verified the feasibility of CRLSMACS, the identified six-component AAFs, and the calculated buffeting response.

3.5 Aerodynamic Admittance of twin-box bridge decks

This section will present the main differences and effects of using a twin-box girder compared to the traditional closed-box girder. The findings and results are based on an experimental study by Wang et al. [23].

In the study by Wang et al.[23], the characteristics of aerodynamic admittance of twin-box bridge decks were investigated. The aerodynamic admittance and the buffeting force coherence along the span wise direction were obtained as well as the pressure distribution around the cross-section. This was done to study the difference between a twin-box bridge deck and a closed bridge deck. To describe the spatial distribution characteristics of aerodynamic forces acting on the bridge, a coherence function must be introduced. The spanwise coherence of measured lift, drag and moment can be described as:

$$Coh_F = \frac{S_F(y_1, y_2)^2}{S_F(y_1), S_F(y_2)} \qquad F = L, D, M \tag{3.80}$$

where $S_F(y_1, y_2)$ is the cross-spectra between forces in two different correlation strips with distance, Δy, and $S_F(y_1)$ and $S_F(y_2)$ are the corresponding auto spectrum for each correlation strip.

One-point spectra of lift and moment on the section model were used to investigate the buffeting force characteristics on twin-box girders. It showed that the total lift and moment on the upstream-box were considerably larger than on the downstream-box. This indicates that the upstream-box mainly provides the buffeting forces on the twin-box. The incoming turbulence and flow separation may affect the formation of the buffeting force acting on a bluff body. Vortex shedding on the trailing edge of the upstream-box will also affect the downstream-box.

The spanwise coherence of the lift and moment of the twin-box girder was found to be larger than those of the incident turbulent wind velocity. This may indicate that traditional methods underestimate the buffeting forces on a twin-box girder and that three-dimensional effects of the incident turbulent wind velocity cannot be neglected. This is due to the relatively small ratio of the turbulence integral scale to the width of the structure. However, it was found that the spanwise coherence of lift and moment on the twin-box girder was smaller than for the closed-box. This suggests that the three-dimensional effect of the incident wind velocity on twin-box girder will be less than for the closed-box. It was also detected that the spanwise coherence on the upstream-box was roughly consistent with the coherence on the closed-box, but much higher than on the downstream-box. Overall, the main reason the coherence on the twin-box is less than the closed-box, is the relatively low spanwise coherence of lift and moment on the downstream-box.

The structure of the vortices is also an important mechanism of buffeting force coherence. To investigate this further, the fluctuating pressure distribution around the twin-box was investigated. It was shown that the pressure distribution of the windward edge of the twin-box and the closed-box was very similar. The difference is more significant on the downstream-box and at the trailing edge of the closed-box. Vortex shedding from the upstream-box can cause higher fluctuating pressure on the downstream-box, as illustrated in Figure 3.5. Wang et al. [31] did another study on the effects of gap width on the buffeting force coherence and aerodynamic admittance of a twin deck. The results showed that under the influence of vortices shed from the upstream-box, the buffeting force coherence on the downstream-box decreased significantly when the gap width increased. The flow pattern created by the gap can be an explanation for why the coherence of a twin-box is smaller than of a closed-box.

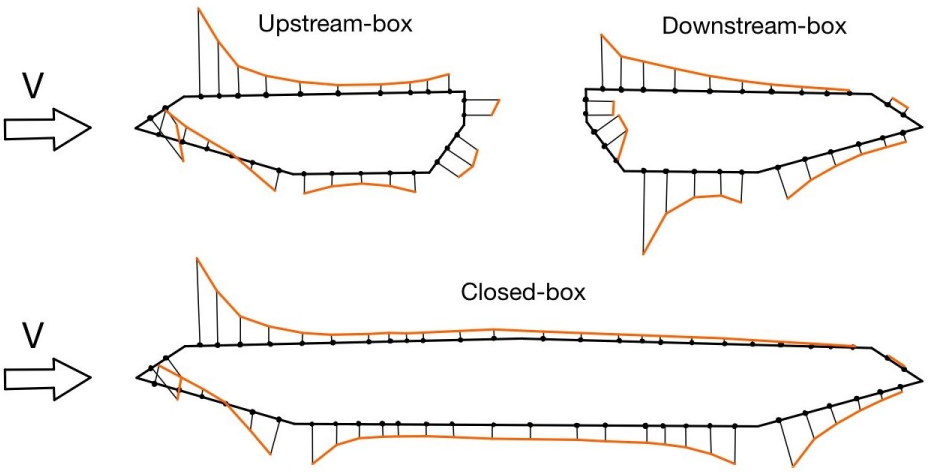


Figure 3.5: Illustration of the mean pressure distribution of the twin-box and closed-box girder based on results obtained by Wang et al. [23].

The aerodynamic admittance functions can be obtained from the experimentally-determined buffeting forces spectrum, the time-averaged aerodynamic force coefficients, and the wind velocity spectrum. It was found that the aerodynamic admittance of a twin-box was higher than for a traditional closed-box. This may indicate an underestimation of the buffeting response of a twin-box if previous research from a closed-box is directly applied to the buffeting analysis of a twin-box. In addition to this, when comparing the aerodynamic admittance of the twin-box with different ratios of integral scale to width, it showed that if the ratio increases, the admittance increases. This fact suggests that the aerodynamic admittance of a twin-box is strongly dependent on the ratio of integral scale to width and on the wind field at low reduced frequencies. With these detailed results, it is clear that the aerodynamic admittance is also dependent on the flow separation and not only the incoming turbulence characteristics. However, both the effect of integral length scale and the effect of flow separation should be included in the three-dimensional effect of turbulence.

3.6 Estimation Methods for Aerodynamic Admittance Functions

As mentioned, the aerodynamic admittance functions is an important transfer function for estimation of the buffeting response as it transfers the turbulent wind to buffeting forces. Due to the complexity of the flow separation on bluff bodies, such as bridge decks, it has always been problematic establishing an exact expression for aerodynamic admittance function. In Section 3.4.3, several experimental methods have been presented. In this thesis, three different methods is used to estimate the admittance functions; the general, the auto-spectral and the cross-spectral.

The general method is a simplified method based on the auto-spectral where the cross-spectra between the horizontal and vertical turbulence components are neglected. A transfer function is found between one force spectrum and one turbulent spectrum which results in an expression that can be used to estimate the admittance functions:

$$|\chi_F|^2 = \frac{S_j}{S_i}, \qquad i = D, L, M \quad \text{and} \quad j = u, w \tag{3.81}$$

where S_i is the power spectra for the drag force, lift force or moment force, while S_j is the turbulent components for the wind in horizontal or vertical direction. This results in three admittance functions; drag, lift and moment, where the horizontal turbulence component is used with lift and moment and the vertical turbulence component is used with drag.

The next method used to estimate the admittance function is the auto-spectra method, also called the equivalent method. The buffeting force spectra can be expressed as [32]:

$$S_D = \left(\frac{\rho U B}{2}\right)^2 \left[4C_D^2 S_u |\chi_{Du}|^2 + (C'_D - C_L)^2 S_w |\chi_{Dw}|^2\right] \tag{3.82a}$$

$$S_L = \left(\frac{\rho U B}{2}\right)^2 \left[4C_L^2 S_u |\chi_{Lu}|^2 + (C'_L - C_D)^2 S_w |\chi_{Lw}|^2\right] \tag{3.82b}$$

$$S_M = \left(\frac{\rho U B^2}{2}\right)^2 \left[4C_M^2 S_u |\chi_{Mu}|^2 + C'^2_M S_w |\chi_{Mw}|^2\right] \tag{3.82c}$$

where χ_{ij} is the aerodynamic admittance functions, V is the mean wind and B is the width of the section model, C_D, C_L, C_M are the drag, lift and moment coefficients. C'_D, C'_L, C'_M are the derivative of the coefficients. This method assumes that the two AFF's (χ_{iu}, χ_{iw}) in each buffeting force spectra are equal to each other, i.e $\chi_{iu} = \chi_{iw} = \chi_i$. The equivalent AAF for each buffeting force can then be expressed as:

$$|\chi_D|^2 = \frac{S_D}{\left(\frac{\rho U B}{2}\right)^2 [4C_D^2 S_u + (C'_D - C_L)^2 S_w]} \tag{3.83a}$$

$$|\chi_L|^2 = \frac{S_D}{\left(\frac{\rho U B}{2}\right)^2 [4C_L^2 S_u + (C'_L + C_D)^2 S_w]} \tag{3.83b}$$

$$|\chi_N|^2 = \frac{S_D}{\left(\frac{\rho U B^2}{2}\right)^2 [4C_M^2 S_u + C'^2_M S_w]} \tag{3.83c}$$

The last method used to estimate the admittance functions is the cross-spectral method as described in Subsection 3.4.3. This method makes it possible to determine both χ_{Fu} and χ_{fw}. The method is based on the measured cross-spectra between the fluctuating force coefficient C_F and each component of the turbulence, u and w. The expression of the cross-power spectrum is defined as [27]:

$$\begin{aligned} S_{Fu} &= \frac{\rho V B}{2}\left(a_F \chi_{Fu} S_u + b_F \chi_{Fw} S_{wu}\right) \qquad F = L, D \\ S_{Fw} &= \frac{\rho V B}{2}\left(a_F \chi_{Fu} S_{uw} + b_F \chi_{Fw} S_w\right) \end{aligned} \tag{3.84a}$$

$$\begin{aligned} S_{Fu} &= \frac{\rho V B^2}{2}\left(a_F \chi_{Fu} S_u + b_F \chi_{Fw} S_{uw}\right) \qquad F = M \\ S_{Fu} &= \frac{\rho V B^2}{2}\left(a_F \chi_{Fu} S_{uw} + b_F \chi_{Fw} S_w\right) \end{aligned} \tag{3.84b}$$

where

$$a_F = 2C_F \qquad b_F = \begin{cases} \left[C'_D - C_L\right], & \text{F = D} \\ \left[C'_L + C_D\right], & \text{F = L} \\ \left[C'_M\right], & \text{F = M} \end{cases} \tag{3.85}$$

where the static coefficients C_F is found by using Equation 3.24. The expression for the six aerodynamic admittance functions can then be found by:

$$\begin{aligned} \chi_{Fu} &= \frac{S_w S_{Fu} - S_{wu} S_{Fw}}{a_F \frac{\rho V B}{2}\left(S_u S_w - S_{wu} S_{uw}\right)} \qquad F = L, D \\ \chi_{Fw} &= \frac{S_u S_{Fw} - S_{uw} S_{Fu}}{b_F \frac{\rho V B}{2}\left(S_u S_w - S_{wu} S_{uw}\right)} \end{aligned} \tag{3.86a}$$

$$\begin{aligned} \chi_{Fu} &= \frac{S_w S_{Fu} - S_{wu} S_{Fw}}{a_F \frac{\rho V B^2}{2}\left(S_u S_w - S_{wu} S_{uw}\right)} \qquad F = M \\ \chi_{Fw} &= \frac{S_u S_{Fw} - S_{uw} S_{Fu}}{b_F \frac{\rho V B^2}{2}\left(S_u S_w - S_{wu} S_{uw}\right)} \end{aligned} \tag{3.86b}$$

3.7 Wind Tunnel Effects

The surrounding environment has a strong influence on the wind. Differences in air temperature and local topography are two effects that significantly affect how the wind acts. The temperature affects the air density and the wind speed. Local topography will also change the air pattern and affect the wind speed. The wind tunnel is incapable of recreating these types of effects and is therefore manipulated to get the desired wind flow. The wind tunnel has some additional effects that differ from the natural flow, resulting from the limiting cross-section area of the tunnel. The most important effects of a wind tunnel test will be discussed in this section.

3.7.1 Boundary Layer

Friction along the surface occurs when the wind moves past an object. As a result of this, the velocity close to the surface will be reduced. This effect is called the boundary layer flow, and it stimulates the natural wind to recreate the outdoor flow system. The local topography has a great influence on the flow pattern, which can create turbulence. Therefore, the model must be placed higher than the boundary layer in order to get laminar airflow with constant velocity. This distance is approximately 200 mm from the surface of the wind tunnel at NTNU, as shown in 3.6 [33].

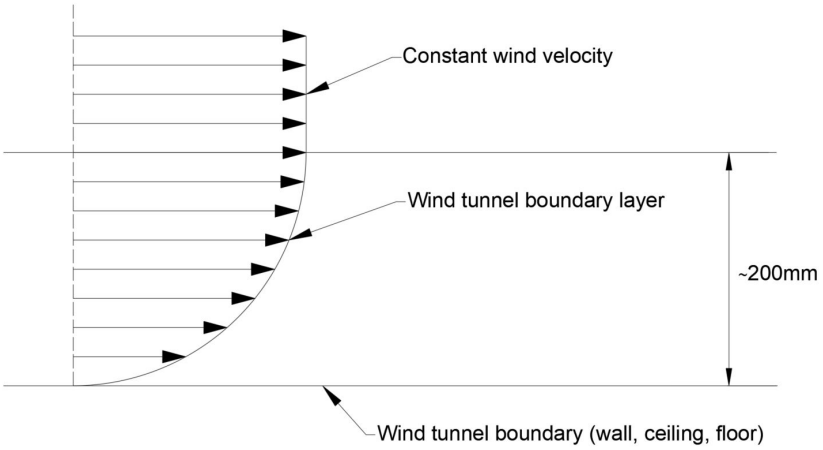


Figure 3.6: Boundary layer effects in the wind tunnel.

As shown in Figure 3.7, it is clear that the bridge is not in the boundary layer.

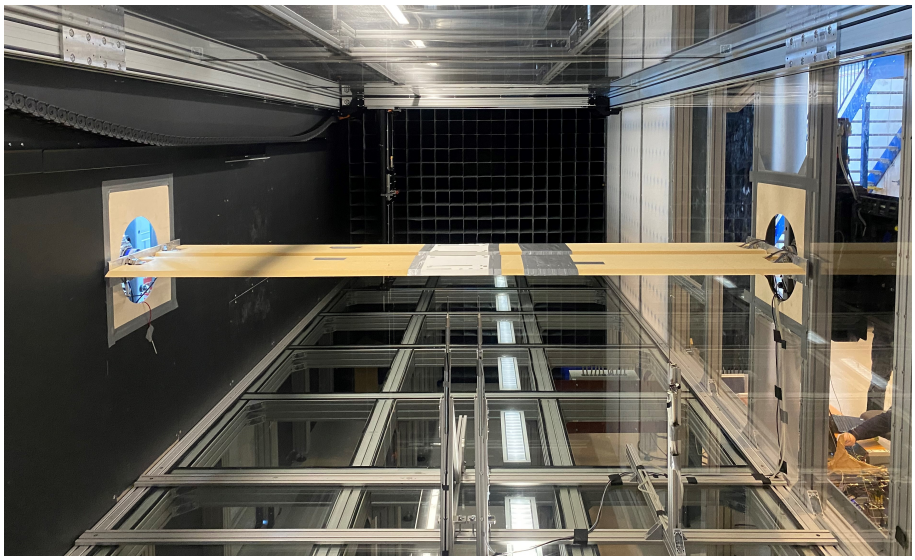


Figure 3.7: Bridge inside the wind tunnel.

3.7.2 Blockage

The wind tunnel will have limited space that restricts the wind flow and causes boundary layers. When the section model is placed inside the wind tunnel, it will obstruct the wind and cause a local flow acceleration. This blockage effect is different for every model. It depends on the model shape, aerodynamic effects, the wind field characteristics, and the blockage ratio $\frac{S}{C}$, where S is the area of the body normal to the wind flow and C is the cross-sectional area of the wind tunnel. If the blockage ratio is below 5%, the distortion can be neglected [10].

3.7.3 End Plates

In order to maintain a two-dimensional flow around the model in the wind tunnel, end plates are mounted at both ends. The purpose of the end plates is to avoid outside flow entering the testing area and to keep the wake two-dimensional. Therefore, the diameter should be at least 8.5 times the model depth to maintain this [34]. In this thesis, the bridge model spans the entire length of the wind tunnel, hence end plates are not needed.

3.7.4 Grid Generated Turbulence

The essential characteristics of turbulence are vortex shedding, separations, and attachments. This can be produced in the wind tunnel by installing a grid net that makes grid-generated turbulence [35]. The gird net is installed upstream of the model to disrupt the wind. The characteristic of the turbulence will be greatly affected by the placement and the shape of the grid.

Grid-generated turbulence will often be described as isotropic and homogeneous. These flows are almost never encountered in practice. However, they can be used to limit the complexity of the flow for numerical and analytical verification. When three fluctuation velocity components are invariant, isotropic flow occurs. This is due to an arbitrary rotation of the defining principal axis and will happen at a specific distance from the grid [36]. Several studies have been done on this topic, such as Liu et al. [35] and Tresso [36], which show the importance of the installation location

of the section model in the wind tunnel. Figure 3.8 shows the active grid in the wind tunnel at the Department of Energy and Process Engineering at NTNU Trondheim.

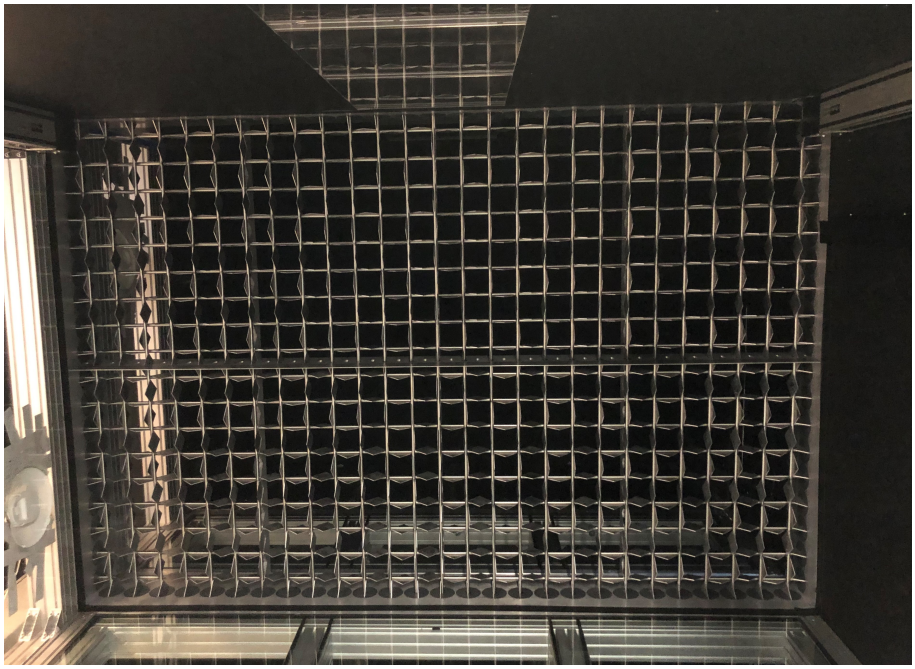


Figure 3.8: Active grid in the wind tunnel at NTNU Trondheim.

3.8 Effects of tube system parameters

The pressure scanners are placed inside the model for some section models, like the one in this master thesis. A tube system is used to connect the pressure scanners to the measure points. The tube system may distort the measurement while transferring the fluctuating pressure to the pressure scanners, leading to inaccurate data. Wang et al. [37] analyzed the relationship between the fluctuating pressure and parameters of the tube system. The tube system parameters; tube length, inside diameter, tube curvature, deflection angle, thickness and material, and the effect of the restrictor were studied. A summary of the effects of the different parameters on the frequency response function (FRF) is given below.

The frequency response function can be used to describe the effects of the tube system on the fluctuating pressure. The FRF is the ratio pressure at the inlet and outlet in the frequency domain and can be defined as [38]:

$$H(\omega) = \frac{Y(\omega)}{X(\omega)} = \frac{FFT(S_{out})}{FFT(S_{in})} \tag{3.87}$$

where FFT is the fast Fourier Transform processing the pressure measured at the surface S_{in} and the output pressure measured by the pressure scanners S_{out}. If the magnitude of FRF, $H(\omega)$, is close to 1 rad and the phase of FRF, $\phi(\omega)$, is close to 0 rad, the effects of the tube system is small.

The results obtained in the study by Wang et al. showed that the magnitude and phase of FRF were affected by the tube length. When the tube length increases, the peak frequency of $H(\omega)$ decreases, and the phase $\phi(\omega)$ increases. The frequency, f_0, is given for a certain tube length. The signal magnitude is amplified for $f > f_0$, while for $f < f_0$, the magnitude is minified. Further, the effects of the tube inside diameter were analyzed, and the results show that the diameter has remarkable effects on the FRF, as the peak value and peak frequency of $H(\omega)$ raises and $\phi(\omega)$ decreases when the diameter increases. The result also showed that tube curvature has almost no effect on the magnitude and phase of FRF.

Moreover, $H(\omega)$ is slightly affected by the deflection angle, but have almost no effect on the $\phi(\omega)$. However, twisting of the tube system in the wind tunnel should be avoided. When the FRFs of the tubes with different thicknesses and materials were analyzed, it was detected that these parameters had a significant effect on the FRF. It showed that when the strength increases, the peak value and peak frequency of $H(\omega)$ increases, while $\phi(\omega)$ decreases. This indicates that the transmission of the fluctuating pressure in the tube is a fluid-solid-interaction phenomenon and is influenced by the material strength and surface smoothness, among other things. Lastly, the results showed that the FRF is remarkably affected by the restrictor. Tube systems with the restrictor have a lower peak and are closer to 1 as $f < f_0$, compared to the tube system without the restrictor.

Overall, all the parameters have non-negligible effects on the FRF of the tube system for fluctuating pressure measurement, except for the curvature.

3.9 Simulation of Turbulence

This section presents the wind field characteristics defined in N400 and a method for simulation of a 2D turbulence field using the Monte Carlo simulation.

3.9.1 Turbulence Spectrum

Relevant definitions of a wind field defined in N400 [39] are presented in this subsection. This is further used for the simulation of a 2D turbulence field together with the spectral properties of a Kaimal spectrum.

The integral length scale ${}^{x}L_u$ given in N400 is defined by:

$$ {}^{x}L_u = \begin{cases} L_1(z/z_1)^{0.3}, & z > z_{min} \\ L_1(z_{min}/z_1)^{0.3}, & z \leq z_{min} \end{cases} \tag{3.88} $$

where L_1 is the reference length scale equal to 100 m and z_1 is the reference height equal to 10 m.

For a 2D approximated homogeneous wind field, the other turbulence intensities and integral length scale is expressed by:

$$ I_w = 1/2 \cdot I_u \quad \text{for} \quad \begin{bmatrix} {}^{y}L_u \\ {}^{z}L_u \\ {}^{x}L_w \\ {}^{y}L_w \\ {}^{z}L_w \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/5 \\ 1/12 \\ 1/18 \\ 1/18 \end{bmatrix} {}^{x}L_u \tag{3.89} $$

where the turbulence intensity, I_u, in the main wind direction can be calculated according to NS-EN 1991-1-4:2005+NA:2009, table NA.4.1.

The auto-spectral density of the turbulence component i, $S_i(f)$, is given by:

$$ \frac{S_i(f)f}{\sigma_i^2} = \frac{A_i \hat{f}_i}{(1 + 1.5A_i\hat{f}_i)^{5/3}} \quad \text{for} \quad i = u, w \tag{3.90} $$

where σ_i is the standard deviation of the turbulence component i and the reduced frequency, $\hat{f}_i$, is defined by:

$$ \hat{f}_i = \frac{f {}^{x}L_i(z)}{V(z)} \tag{3.91} $$

where $V(z)$ is the mean wind velocity at a given height, z.

Using this, the normalized single point Kaimal auto spectrum is presented in Figure 3.9 for the turbulence components u and w.

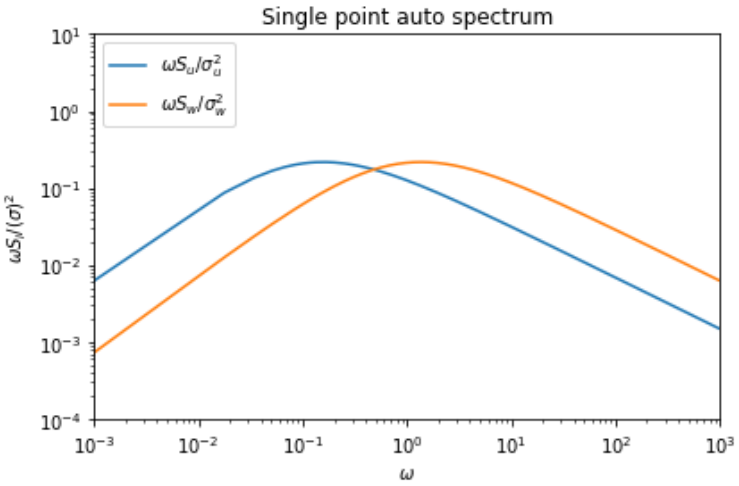


Figure 3.9: A normalized single point auto spectra for the turbulence components u and w.

Further, as a base for the Monte Carlo Simulation, the cross-spectral density of the wind field are used, which can be expressed by:

$$Re[S_{i_1 i_2}(f, \Delta s_y)] = \sqrt{S_{i_1}(f) \cdot S_{i_2}(f)} \cdot e^{-C_{iy} \frac{f \Delta s_y}{V(z)}} \tag{3.92}$$

where Δs_j is the horizontal distance between the considered points, $i_1, i_2 = u, w$, $C_{uy} = 10.0$ and $C_{wy} = 6.5$, respectively.

3.9.2 Monte Carlo Simulation of turbulence

Monte Carlo simulation is based on random number generation and is a well-established and useful tool for the design of complex wind-excited structures. A realizations of a stochastic process can be simulated by [40]:

$$X(t) = \sum_{k=1}^{N} A_k cos(\omega_k t + \phi_k) \quad \text{for} \quad k = 1, 2, 3, ..., \tag{3.93}$$

where $\omega_k = (k - \frac{1}{2})\Delta\omega$, $\Delta\omega$ is a measure of frequency resolution, t is the time, ϕ_k are random phase angles uniformly distributed in the range $[0, 2\pi]$, and A_k are the deterministic constants which are currently unknown.

The mean value of the process is defined by:

$$E[X(t)] = \int_0^{2\pi} \sum_{k=1}^{N} cos(\omega_k t + \phi_k) \frac{1}{2\pi} d\phi_k \tag{3.94}$$

Further, the autocorrelation function for an ensemble average can be expressed by:

$$
\begin{aligned}
R_X(t+\tau,t) &= E[X(t+\tau)X(t)] \\
&= \int_0^{2\pi}\int_0^{2\pi}\sum_{k=1}^{N}\sum_{k=1}^{N} A_k A_l cos(\omega_k(t+\tau)+\phi_k)cos(\omega_l t+\phi_l)\frac{1}{(2\phi)^2}d\phi_k d\phi_l \\
&= \sum_{k=1}^{N} A_k^2\frac{1}{2}cos(\omega_k t) = R_X(\tau)
\end{aligned}
\tag{3.95}
$$

Hence, it can be observed that the process is at least weakly stationary. By using the central limit theorem, it can be verified that the process converges towards a Gaussian process when $N \to \infty$, which signify that the process converges towards a stationary process. The time-averaged value of the process is defined by:

$$
\begin{aligned}
\langle X(t)\rangle &= \lim_{T\to\infty}\frac{1}{2T}\int_{-T}^{T} X(t)dt \\
&= \lim_{T\to\infty}\frac{1}{2T}\sum_{k=1}^{N}\int_{-T}^{T} A_k cos(\omega_k t+\phi_k)dt = 0
\end{aligned}
\tag{3.96}
$$

Accordingly, the autocorrelation function can be expressed by taking the time average and is given by:

$$
\begin{aligned}
R_X(\tau) &= \langle X(t+\tau)X(t)\rangle \\
&= \lim_{T\to\infty}\frac{1}{2T}\int_{-T}^{T} X(t+\tau)X(t)dt \\
&= \lim_{T\to\infty}\frac{1}{2T}\sum_{k=1}^{N}\sum_{l=1}^{N}\int_{-T}^{T} cos(\omega_k(t+\tau)+\phi_k)cos(\omega_l t+\omega_l)dt \\
&= \sum_{k=1}^{N} A_k^2\frac{1}{2}cos(\omega_k\tau)
\end{aligned}
\tag{3.97}
$$

By using the central limit theorem and comparing Equation 3.94 and 3.97 with Equation 3.95, it it evident that the process converges towards an ergodic process when $N \to \infty$.

Provided that N is sufficiently large, Equation 3.93 can be used to simulate an ergodic Gaussian process with a prescribed spectral density $S_X^0(\omega)$ or an autocorrelation function $R_X^0(\omega)$. The following expression is introduced:

$$
A_k = \sqrt{2S_X^0(\omega_k)\Delta\omega}, \quad \omega_k = k\Delta\omega \tag{3.98}
$$

By inserting the expression in Equation 3.95, the following is obtained:

$$
R_X(\tau) = \sum_{k=1}^{N} S_X^0(\omega_k)\Delta\omega cos(\omega_k\tau) \tag{3.99}
$$

and

$$\begin{aligned} R_X(\tau) &= \lim_{T\to\infty,\Delta\omega\to d\omega} R_X(\tau) \\ &= \lim_{T\to\infty,\Delta\omega\to d\omega} \sum_{k=1}^{N} S_X^0(\omega)\Delta\omega cos(\omega_k\tau) \\ &= \int_{-\infty}^{\infty} S_X^0(\omega)cos(\omega\tau)d\omega \end{aligned} \tag{3.100}$$

Hence, $R_x(\tau)$ converges to the desired correlation function if A_k is defined as in Equation 3.98 and $N \to \infty$.

It is noted that the simulated stochastic process $X(t)$ will be periodic with the period $T_0 = \frac{2\pi}{\Delta\omega}$ and therefore only half the period will be utilized. Similarly, $\Delta\omega = \frac{\pi}{T}$. Additionally, $\Delta\omega$ should be selected such that narrow peaks in the spectral density is reasonably represented. A requirement for this is often that $\Delta\omega$ is significantly smaller than the effective bandwidth of the narrowest peak, which leads to a large number of harmonic component. Consequently, this is an expensive and time consuming approach.

Another method for the realization of the stochastic process is by applying the Fast Fourier Transform (FFT) to Equation 3.93 and 3.98. By using the FFT technique, this drastically improve the computational efficiency of the algorithm and Equation 3.93 can be expressed by:

$$X(t) = Re\left(\sum_{k=1}^{N}(A_k e^{i\phi}k)e^{i\omega}k^t\right) \tag{3.101}$$

where the discrete Fourier transform can be recognized. The transformation can be performed by using the FFT algorithm in Spyder.

By using the method described and the Kaimal turbulence spectrum presented in Subsection 3.9.1 a 2D simualted turbulence field is simulated and shown in Figure 3.10.

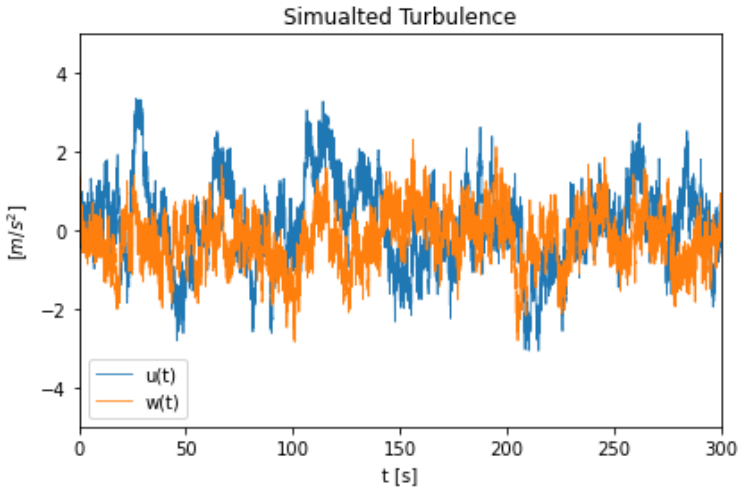


Figure 3.10: Simulated 2D Turbulence field.

Further, is the simulated turbulence component, u, compared with the Kaimal spectrum in Figure 3.11.

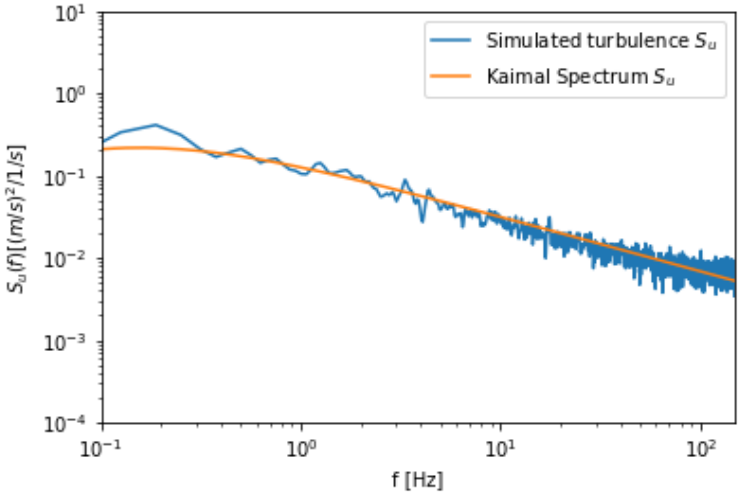


Figure 3.11: Turbulence spectrum of the 2D simulated turbulence compared to the Kaimal Spectrum.

Chapter 4

Design and Building Process of the Bridge Model

This chapter describes the design and building process of the twin-box bridge model. The model was built in the Structural engineering laboratory at NTNU Trondheim, Department of Structural engineering. The choice of cross-section is based on a previous master's thesis at NTNU, where different twin-deck configurations of suspension bridges were studied. The complete model consists of the material Divinycell, an aluminium pipe, and a mid-section. The mid-section consists of a 3D-printed section and several plastic tubes measuring the pressure.

4.1 Choice of Cross Section

The model in this thesis is based on a previous master's thesis were nine different twin-deck configurations with different geometry and gaps were explored to achieve sufficient aerodynamic stability. The chosen cross-section for this thesis showed sufficient aerodynamic stability, and a linear behaviour [41]. The bridge model is built in a 1:50 scale and consists of two identical decks. Figure 4.1 illustrates the geometry of one of the girders, while Figure 4.2 illustrates both girders with the gap between them.

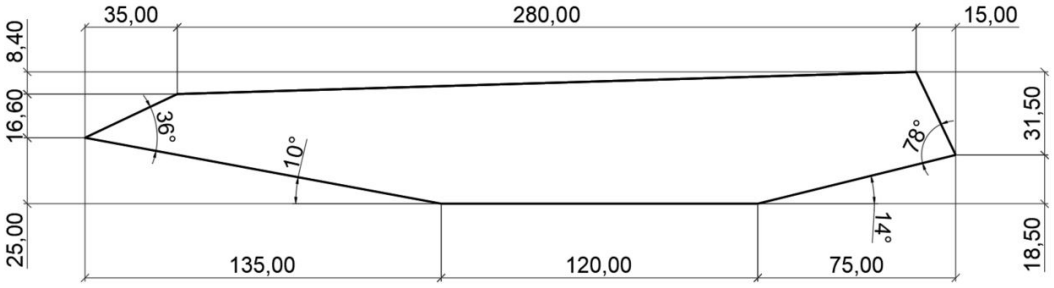


Figure 4.1: Cross-section of the upstream-box.

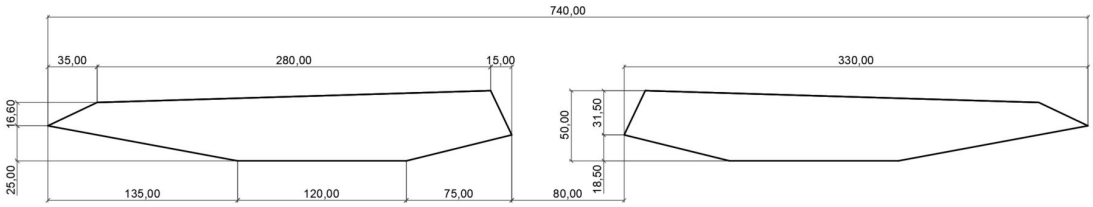


Figure 4.2: Cross-section of twin-box bridge model with gap.

4.2 Material Properties

Each section consists of the light material, Divinycell, a core aluminium pipe and a 3D-printed mid-section. The main contribution to the stiffness is the aluminium pipe. The mid-section, where the pressure on the bridge surface is measured, is a 3D-printed section in the plastic material PLA. The different materials used and their purpose are listed in Table 4.1.

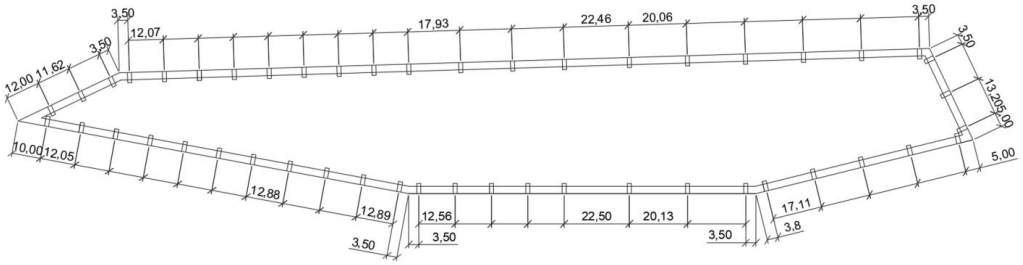
Main Part	**Purpose**	**Material**
Aluminium pipe	Stiffness of model	40x1.5mm Aluminium pipe
Foam model	Cross section shape	Divinycell
	Adherent	Epoxy Resin + hardener
3D-printed section	Cross section shape	PLA
	Pressure tubes	1.5 Urethane tubes
	Screws	2xM4 30mm and 2xM4 40mm

Table 4.1: Materials used in the bridge model and their function.

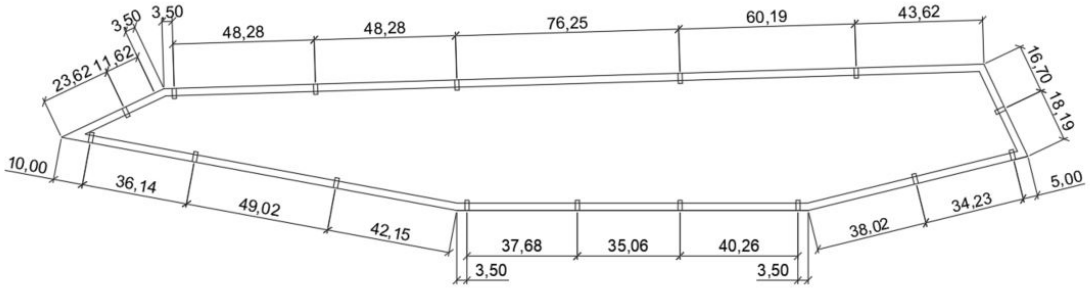
4.3 Distribution of Pressure tubes

In order to measure the fluctuating pressure on the bridge deck, 256 pressure tubes were placed around the 3D-printed mid-section. According to Rocchi et al.[42], the pressure tubes should be distributed closer where a strong pressure gradient is expected. The pressure gradient will be significantly larger at the windward edge and decreases along the deck's surface. This corresponds with the pressure distribution obtained for the wind tunnel test done by Larose[11] as well as the previous study by Wang et al.[23].

256 pressure tubes were distributed along six strips around the 3D-printed mid-section, half on each box. The first strip consists of 48 tubes distributed around the cross-section, while the remaining five strips consist of 16 tubes each, see Figure 4.3. The five strips with 16 tubes will act as correlation lines and are used to measure the pressure correlation in the spanwise direction. The span length, where the correlation lines are distributed, is 220 mm, and the strips are spaced at various distances, as shown in Figure 4.4.



(a)



(b)

Figure 4.3: Distribution of pressure tubes in millimeter, where (a) is the first strip with 48 tubes and (b) is the remaining strips with 16 tubes.

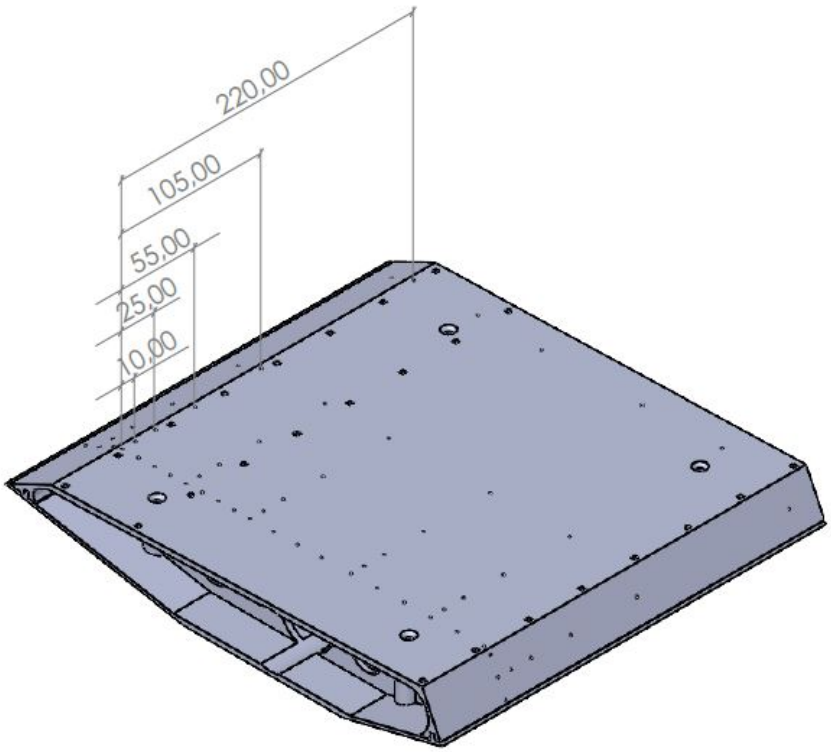


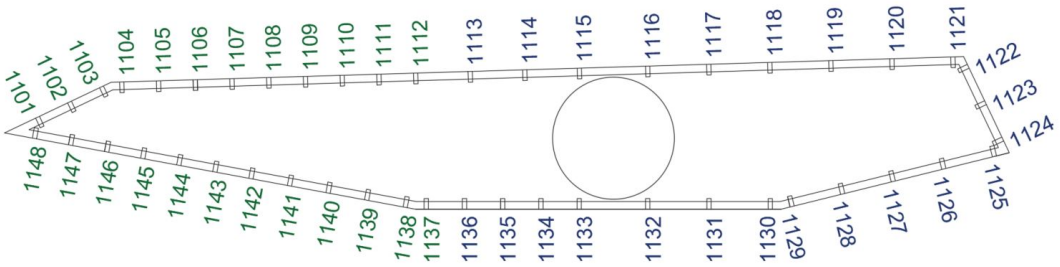
Figure 4.4: Distance of correlation lines in millimeter illustrated on the 3D-printed mid-section.

In order to have an overview of which tube that is attached to which channel on the pressure scanners, a numbering system is made. The numbering system consists of four numbers informing where the tube is located on the model. The first number identifies which box the tube belongs to, the second number tells which correlation line the tube belongs to, and the two last numbers are the tube position on the given correlation line. Table 4.2 illustrates the numbering system of the pressure tubes.

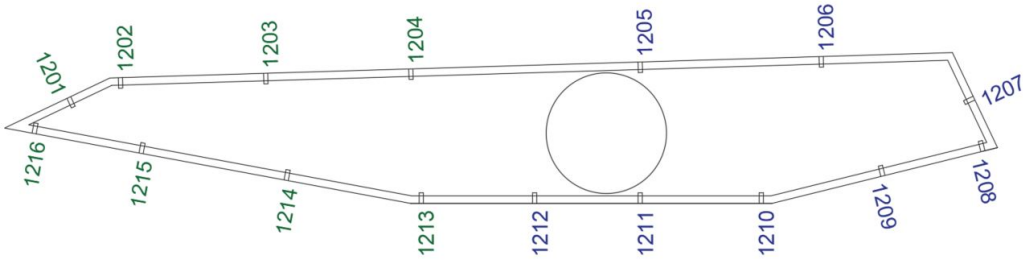
	Line 1	**Line 2**	**Line 3**	**Line 4**	**Line 5**	**Line 6**
Upstream-box	1101-1148	1201-1216	1301-1316	1401-1416	1501-1516	1601-1616
Downstream-box	2101-2148	2201-2216	2301-2316	2401-2416	2501-2516	2601-2616

Table 4.2: Numbering system of the pressure tubes.

For the upstream-box, the numbering starts at the left edge and clockwise around the cross-section and mirrored for the downstream-box. Figure 4.5 illustrates the numbering system on the first and second correlation lines on the upstream-box. The colour on the numbers corresponds to which scanner the tube belongs to. Green numbers for the upstream-box indicate that the tube belongs to scanner no. 179, while blue is for scanner no. 180. As illustrated, more than half of the tubes are on the left side of the aluminium pipe. Therefore, the tubes with blue numbering on the left side must be threaded through the aluminium pipe to the correct scanner. The same implies for the downstream-box.



(a)



(b)

Figure 4.5: Numbering system on correlation line one and two on the upstream-box.

4.4 Building Process

This section will present the building process of the final wind tunnel model. The mid section is designed in SolidWorks and 3D-printed at NTNU Gjøvik.

4.4.1 SolidWorks

SolidWorks is a mechanical design automation application that makes it possible to sketch out ideas, experiment with features and dimensions, and produce models and detailed drawings. It uses a 3D design approach where a 3D-model is created from the initial sketch to the final results. Further, 2D drawings can be created from the 3D-model. "Part", as seen in Figure 4.6, is the basic building block in SolidWorks and is a 3D representation of a single design component. Moreover, "Assembly " is a 3D arrangement of parts or other assemblies [43]. This software is used to design the mid-section of the twin-box bridge before it is 3D-printed.

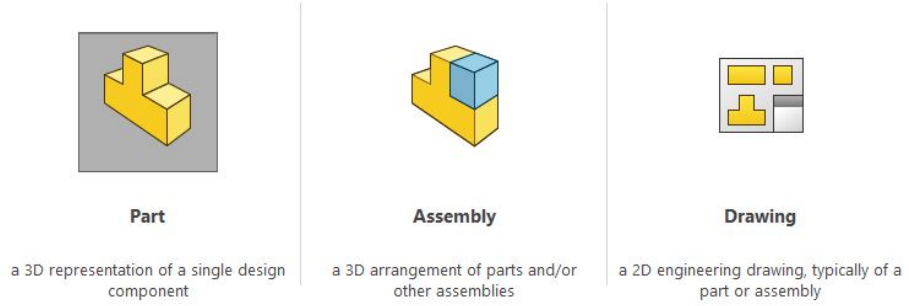
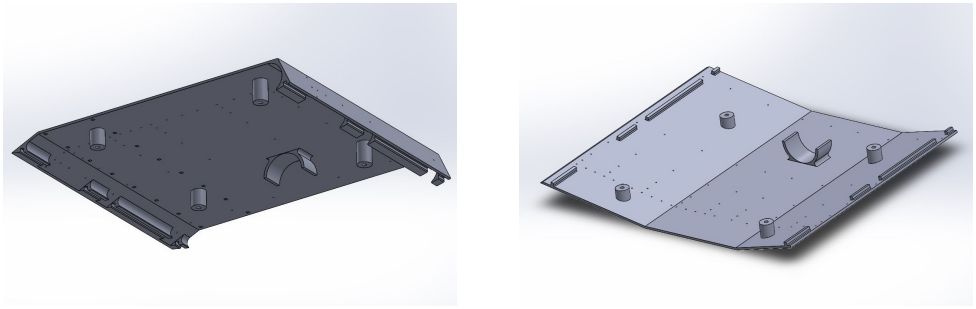


Figure 4.6: Options in SolidWorks

4.4.2 3D-printed mid-section

A previous master's thesis by Haldosen and Jahren [44], used Lexan plates that were glued together around the mid section. These plates were not completely sealed and caused some irregularities that affected the flow. In addition, some of the pressure tubes detached from the Lexan plate before the test was executed. With the plates glued together, it was difficult to access these tubes. Therefore, it was desirable to create a model that was possible to take apart in order to get access to the tubes if needed. It was suggested early in the process to 3D-print the mid-section where the pressure tubes are supposed to be. The solution was, therefore, to 3D-print two separate parts that could be fastened together with screws, see Figure 4.7. The mid-section was sketched and designed in SolidWorks. The first concept was a 2.5 mm thick cross-section with four screw connections, as seen in Figure 4.8. It was designed with holes with a diameter of 1.4 mm for the tubes, distributed as discussed in Section 4.3. There are locks at both sides of the cross-section to hold the two parts together before being fastened with four M4 screws, in addition, to preventing wind from entering the cross-section.



(a) The top part (b) The bottom part

Figure 4.7: The first concept for the 3D-printed section.

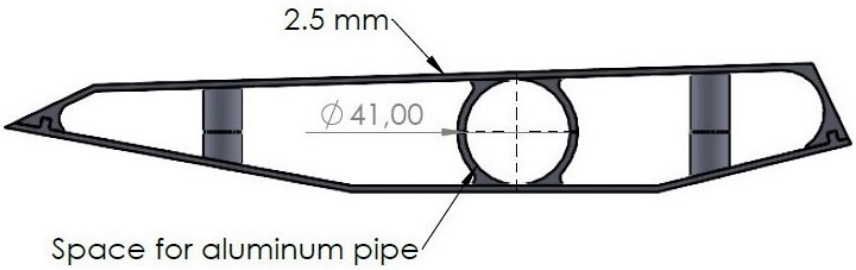


Figure 4.8: Cross-section of the first concept.

The model is 3D-printed at the Department of Manufacturing and Civil Engineering, NTNU Gjøvik at the Addlab. The printer supports SolidWorks files, saved as a STEP file format. The 3D-printer, EOS P395 is an additive manufacturing system that produces parts from plastic powder by using a laser. The chamber is heated up to approximately 180°C during the process, and the laser is used to add the rest of the energy needed to melt the powder. The building volume is 300x300x580 mm. Therefore, the model has a total length of 290 mm to fit innside the 3D-printer. Figure 4.9 shows the 3D-printer that was used to print the model.

Figure 4.9: 3D-printer EOS P395

When finalized, the 3D-printed section showed some defects. Since the thin section consists of relatively large surfaces with little support, tension occurred when it cooled down, which caused the section to bend, see Figure 4.10. This also applied when the two parts were fastened with screws. The bent section would affect the pressure measurements and give inaccurate results.

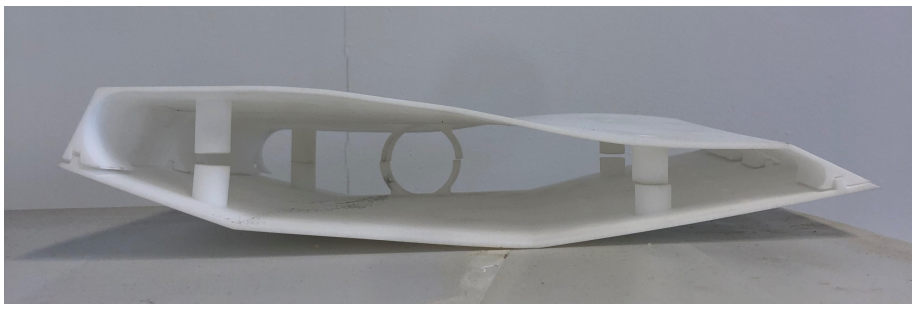


Figure 4.10: 3D-printed model of the first concept

For the 3D-section to be functional for testing in the wind tunnel, it had to be enhanced. An attempt to heat the model and slowly cool it down was done. The cross-sectional shape of the bridge was milled out of Divinycell such that the 3D-printed parts could be pressed down to the desired shape. A heating gun was used to heat the model before being pressed down with planks and clamps, see Figure 4.11. This outcome was not optimal as there was still some tension and bending of the cross-section.

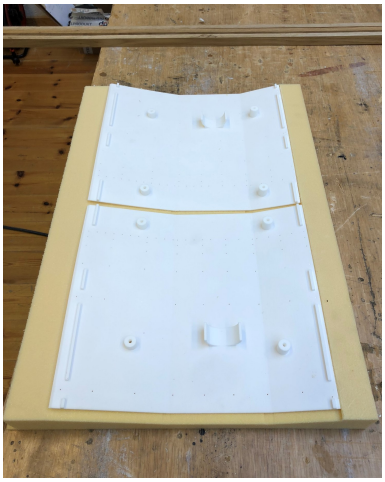
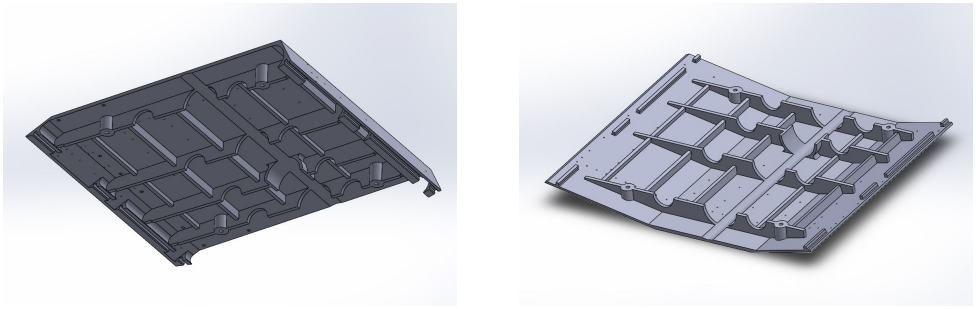


Figure 4.11: An attempt to press the 3D-section into the desired shape.

After discussing possible solutions, the conclusion was to make the cross-section more rigid and 3D-print an improved model. The thickness of the upper and lower surfaces was increased from 2.5 mm to 4 mm, and stiffeners were inserted in both directions, see Figure 4.12 and 4.13. These improvements resulted in a more rigid cross-section. It still had some tension, but it was not an issue when the two parts were fastened together with screws, see Figure 4.14.



(a) The top part (b) The bottom part

Figure 4.12: The second concept for the 3D-printed section.

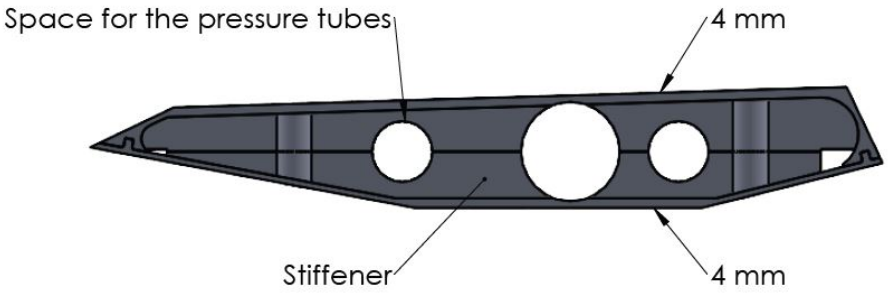


Figure 4.13: The cross-section of the second concept.

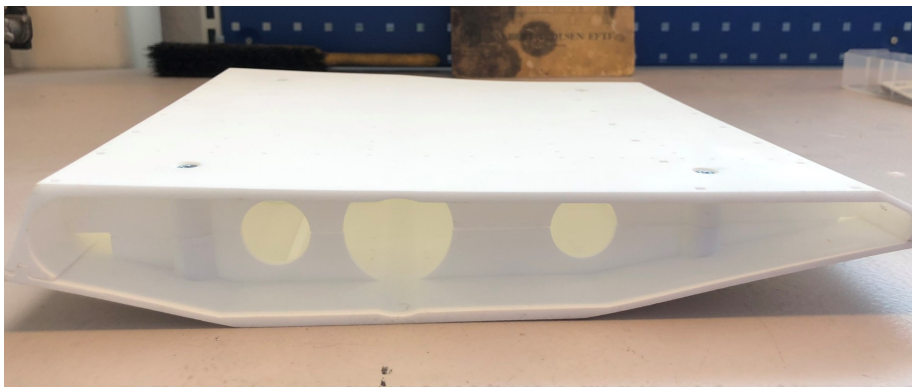


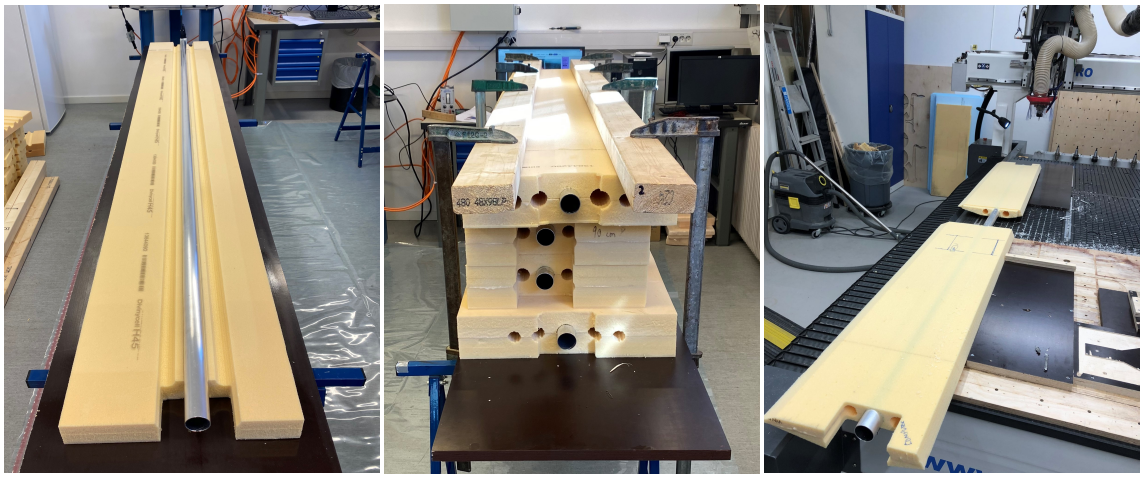
Figure 4.14: 3D-printed model of the second concept.

4.4.3 Girders

The two identical bridge decks are milled out of the PVC-based material Divinycell. This light material is easy to use and mill into different shapes. The Divinycell section itself will not give the desired strength and stiffness. Therefore, an aluminium pipe was inserted throughout both sections as a reinforcement. The aluminium pipes will also be connected to the load cells in the wind tunnel. The first step was to mill out four parts in Divinycell with space for the pipe and the tubes. For this, a milling machine (CNC-router), used the coordinates from a Matlab script to mill out the bridge sections. The Matlab script was provided by the supervisor Ole Andre Øiseth and adapted so that the 3D-printed section would fit.

The next step in the process was to glue two parts, a bottom, and a top part, together with the aluminium pipe inside. Epoxy resin was used to glue the bridge together. Before the glueing could start, the epoxy had to be mixed with a hardener. Mixing the epoxy and the hardener starts a chemical reaction that transforms the mixed liquid into a solid. This means that the assembly of the bridge deck had to happen fairly quickly and effectively before the glue started to cure.

The epoxy was applied to one side of the section before the bottom, and the top part were glued together with the aluminium pipe inside. The sections were then clamped together while the epoxy cured and hardened. This formed the base for the bridge section. After this, the outside geometry of the model was milled out. Space for placement of the scanners was manually milled out. Figure 4.15 shows parts of the building process.



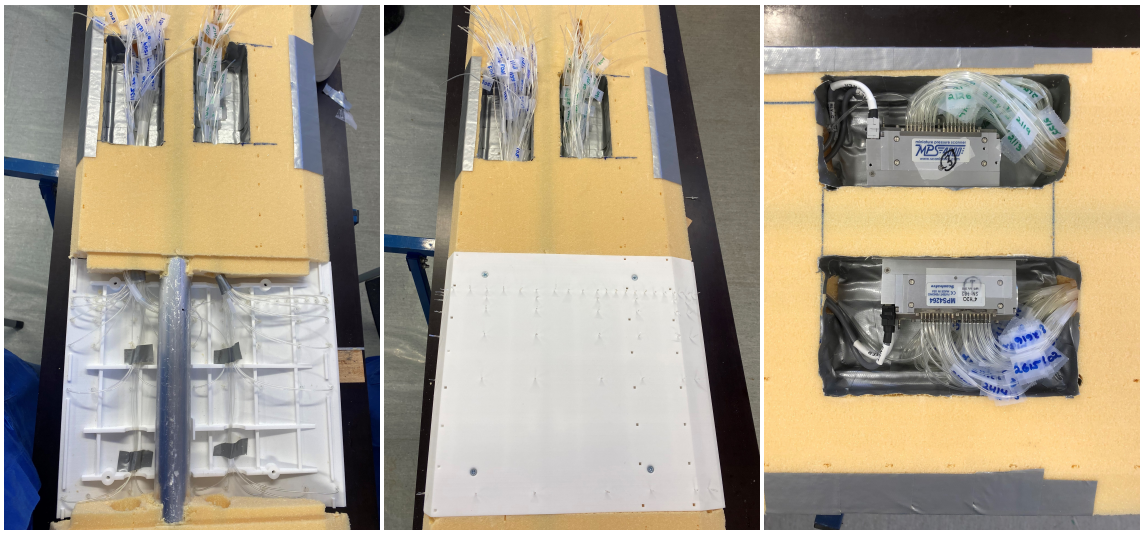
(a) Element with aluminum pipe (b) Glued Divinycell elements (c) Milling with CNC-router

Figure 4.15: The building process

4.4.4 Application of the 3D-printed section and pressure tubes

The 3D-printed sections was designed with 256 holes for the pressure tubes. However, some deviations occurred in the 3D-printing process. Therefore, the holes were additionally drilled out before the tubes were pulled through to ensure the correct size of the holes. The diameter of the holes is 1.4 mm, while the tubes have a diameter of 1.5 mm. This is to ensure that the tubes is fastened sufficiently. Since the aluminium pipe is not centred in the width direction of the bridge box and the number of tubes on each side is not even, some had to go through the pipe to the other side. Therefore, holes in the aluminium pipe were drilled out. Further, the tubes were cut to a suitable length and pulled through the holes. In order to have an overview of which tube would go to which scanner and channel, the tubes were marked with the numbering system described in Section 4.3. In addition, they were marked by the colours, green and blue, to make it easier sort the tubes and connect them to the right scanner. The tubes were then glued to secure and prevent them from detaching from the 3D-section.

Before the 3D-sections were attached to the girders, the tubes were pulled through the model. This was done for both the top and bottom 3D-part, before they were fastened together with four screws. Each tube was then trimmed to the right length and attached to the right channel on the scanners. Further, tape was used to make the surfaces even and airtight between the 3D-printed parts and the rest of the model. Last, the tubes were cut to the surface of the 3D-printed parts to make it as even as possible.



(a) Overview of the inside of the 3D-printed section.

(b) Overview of the 3D-printed sections.

(c) Pressure scanner placed inside the bridge model.

Figure 4.16: Details of the 3D-printed model with the pressure tubes and scanners.

4.4.5 Built-in Tuned Mass Damper

In order to improve the stability of the girders, TMDs were used. A previous master's thesis by Grongstad and Kildal[45] used a self-made TMD to damp out the vortex shedding vibrations, and it showed to be favourable. Another master's thesis by Sivertsen and Strehl[41] used an upgraded design for the TMD where the TMD was made of a wood handle, a wood skewer, and 20 coins on the tip. The coins were replaced with a piece of steel reinforcement in this master's thesis. The self-made TMDs were placed inside both sections, see Figure 4.17. The wood skewer and the reinforcement piece functions as a cantilevered mass that can vibrate freely if the section vibrates. To tune the TMDs, the skewers were ulled to the length where the TMD and the girder approximately had the same frequency.

Figure 4.17: The self-made TMD

4.4.6 Railings

Research by Siedziako and Øiseth[46] showed that attachments to the girder, such as railings, can have a significant influence on the results from the wind tunnel tests. Therefore, the twin-box model is tested with and without railings to investigate how the measurement is affected. Laima et al.[47] experimentally studied the influence of attachments on the aerodynamic characteristics and vortex induced vibrations(VIV) of a twin-box girder. Handrails and crash barriers showed to have a weak influence on the pressure distribution on the upper and lower surface except in the vicinity of the attachments. However, they could cause a decrease in the vortex shedding frequency. Railings are also a necessary safety measure on a finished bridge and cannot be disregarded. The railings are milled out by the CNC-router in a hard plastic transparent material, as shown in Figure 4.18. Both handrails and crash barriers were used on the model.

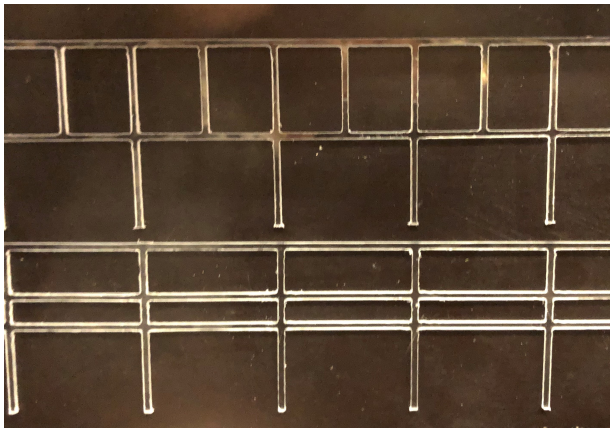


Figure 4.18: Handrails (top) and crash barriers (bottom).

Chapter 5

Wind Tunnel Testing

The wind tunnel testing of the twin-box model is executed at the Fluid Mechanics Laboratory at the Department of Energy and Process Engineering, NTNU Trondheim. The wind tunnel testing is used to study the pressure distribution on the surface of the bridge caused by the incoming flow and estimate the aerodynamic admittance functions. This chapter describes the experimental setup, the measurement system, and the different tests executed in the wind tunnel.

5.1 Experimental Setup

This section describes the instruments used in the wind tunnel tests and how the different components are assembled. The instruments required to achieve the pressure and wind data consisted of a Pitot Probe, two Cobra Probe and four MSP4264 Miniature Pressure Scanners. The experimental setup synchronizes the Cobra Probe and the pressure scanners to measure simultaneously and register identical timer histories.

A test of the instruments was done using a table fan before the final tests in the wind tunnel. It is important to test the system prior to the actual wind tunnel test to exclude possible errors during the instruments' assembly or errors with the pressure tube system. After the test, it was detected that two of the tubes, no.1115 and no.2115, were squeezed due to being too close to the aluminium pipe. The solution to this and its effect on the measurements will be discussed further in Section 5.3. Figure 5.1 illustrates a flow chart of the experimental setup from sampling in the wind tunnel to raw data and further data processing.

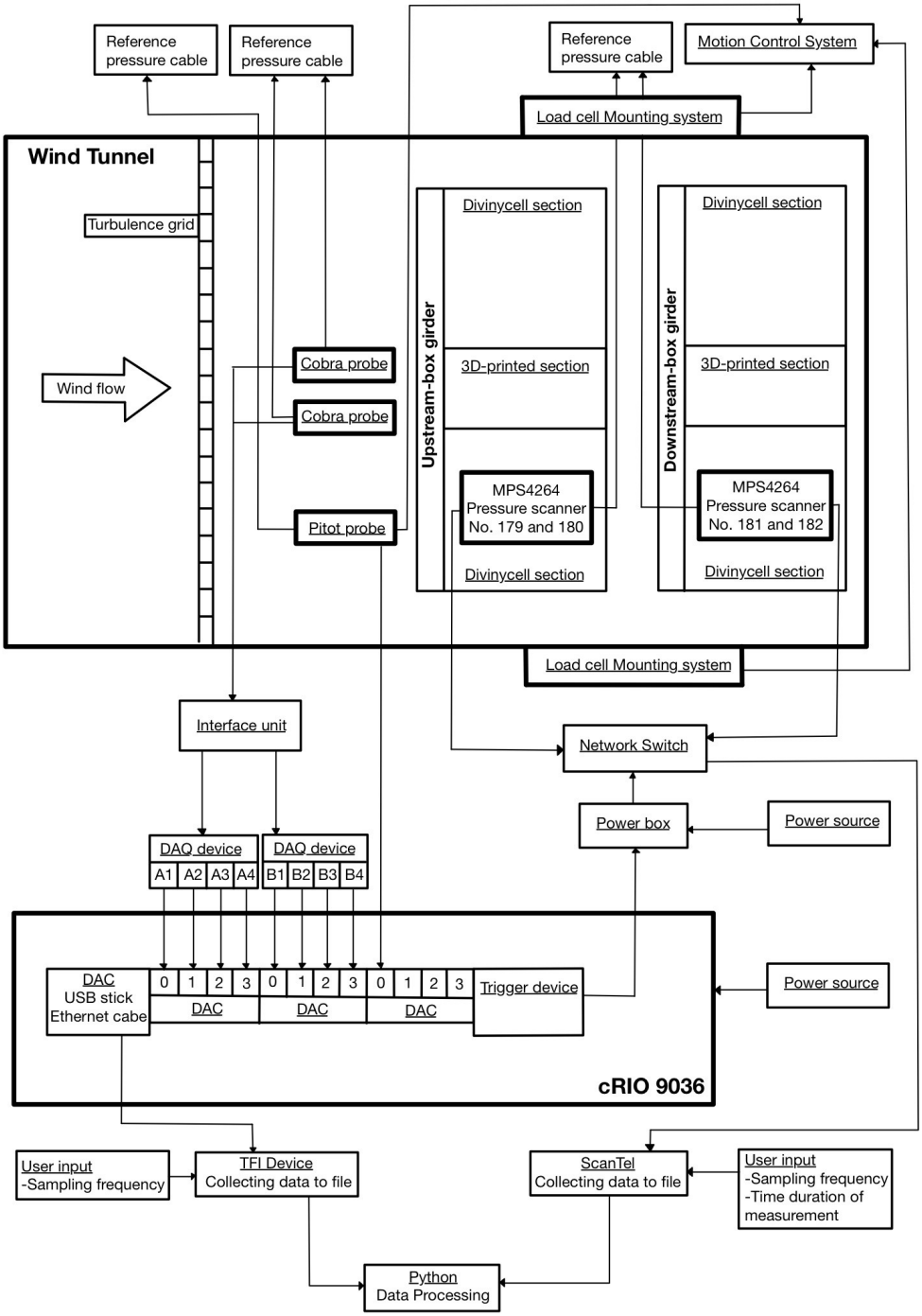


Figure 5.1: Flow chart of the experimental setup.

5.1.1 General Experimental Setup

The wind tunnel at the Fluid Mechanics Laboratory is a closed loop, where the test section is 2.7 m wide, 1.8 m high and 11.1 m long. The experimental setup inside the wind tunnel is shown in Figure 5.2.

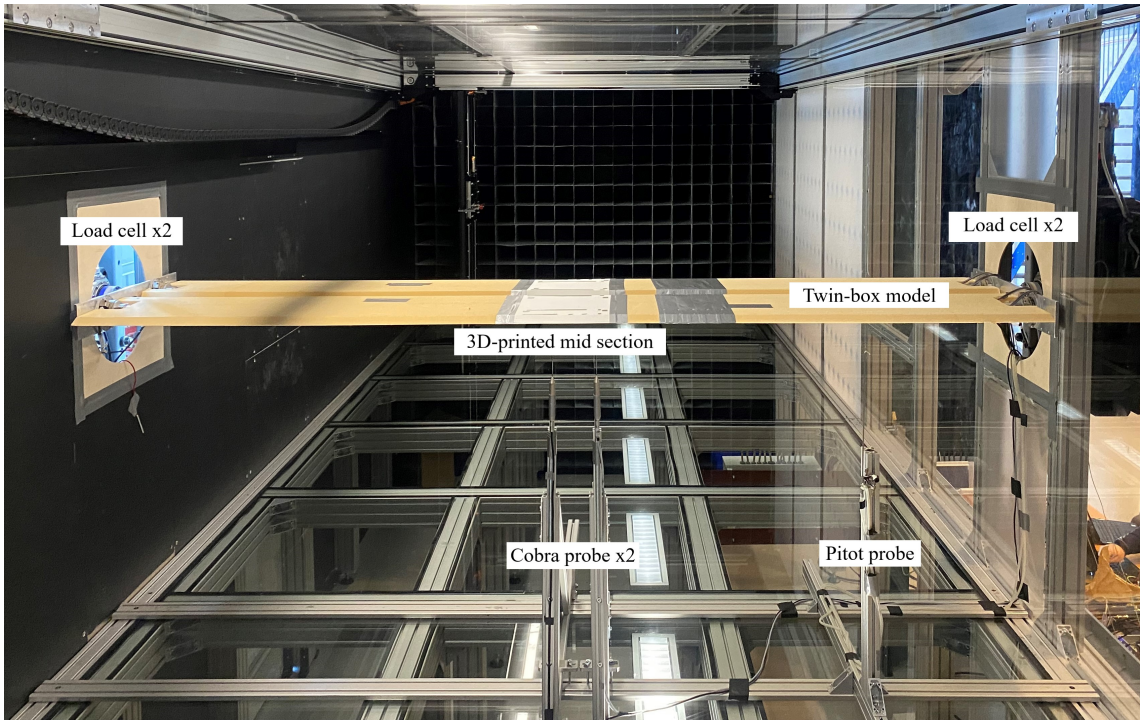


Figure 5.2: The experimental setup inside the wind tunnel.

The bridge section was mounted to the load cells with a clamping system. The load cells were spaced at a distance of 340 mm to get a gap of 80 mm between the girders. The loads cells measures the wind-induced forces in three directions at each end of the sectional model. Further, the load cells are mounted to the actuators connected to the outer wall. The actuators generate motion in 3 degrees of freedom; horizontal, vertical and rotational. Figure 5.3 shows how the twin-box section is mounted to the load cells and the actuator.

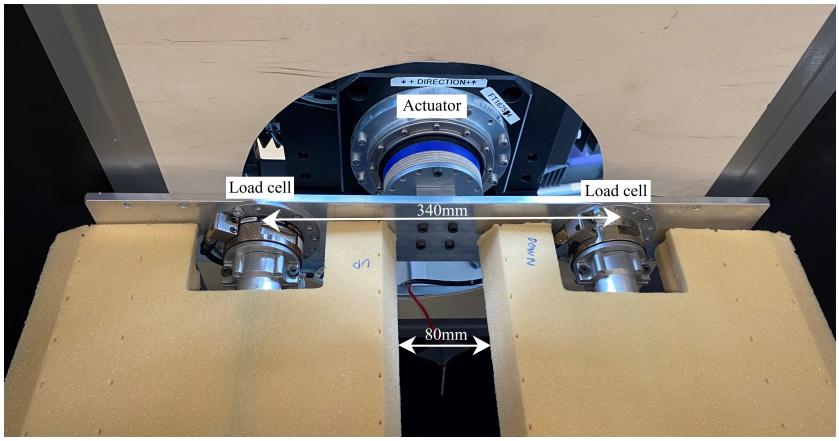


Figure 5.3: Distance between the load cells and the gap width.

The Cobra Probes measures the turbulence and were placed in front of the first and fifth correlation line at the upstream side. This resulted in a distance of 105 mm between the Cobra Probes, see Figure 5.4(a). However, other distances were used in the flow tests, which will be presented in Subsection 5.2.4. The Cobra Probes were placed 40 cm from the windward edge of the upstream-box, see Figure 5.4(b). The Pitot Probe measures the wind velocity inside the wind tunnel. Since the Cobra Probes were placed in front of the mid-section, the Pitot Probe were placed to the right.

An active grid is placed between the inlet and the twin box model, shown in Figure 5.4(c). The grid generates turbulence and was used as a still open grid and with two different gird rotations, 0.5 Hz and 7 Hz. At the inlet, the velocity is uniform, and the flow is close to laminar [48].

(a) The distance between the Cobra Probes

(b) The distance between a Cobra Probe and the windward edge

(c) Active grid used for turbulence generation

Figure 5.4: Details inside the wind tunnel.

5.1.2 MPS4264-Miniature Pressure Scanner

The MSP4264 Miniature pressure scanner measures the fluctuating pressure around the twin-box bridge. The MPS is a versatile scanner that is specifically designed for use in a wind tunnel where operational conditions and pressures do not exceed 50 psi [49]. The small size of the MPS make it easy to be installed inside the wind tunnel model, and it is therefore a user-friendly interface ideal for wind tunnel testing.

A total of four MPS pressure scanners were used in this master's thesis, two scanners in each box. Each scanner incorporates 64 individual piezoresistive pressure sensors. The power and Ethernet connection is located at the end of the MPS. The power connection also serves as a trigger connection that synchronizes the data collected between the MPS scanners and the Cobra probes. The Ethernet cable is connected to a computer via a network switch and is mainly used for communication with the MPS. When the MPS has been connected, the communications can be established. This is done by the communication utility *ScanTel*. ScanTel is a text-based, command-line program that allows the user to modify the configuration variables and collect data in both TCP/IP and UDP format [49]. The MPS4264 Pressure scanner is shown in Figure 5.5.

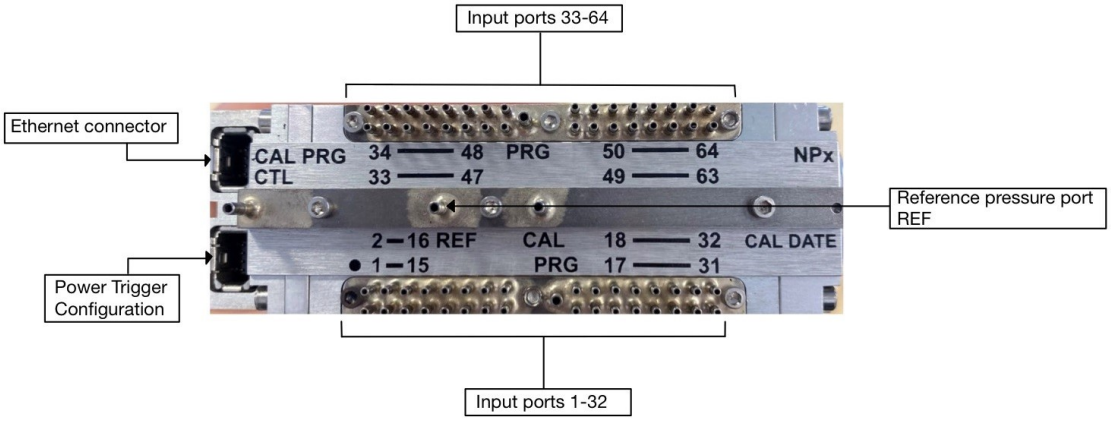


Figure 5.5: A MPS Miniature Pressure Scanner

The pressure tubes are attached to the input ports, 1-64, on the MPS scanners. Appendix A presents an overview of which pressure tube that are attached to which pressure channel and scanner. To ensure that no unwanted offsets are introduced when a zero offset calibration is performed, an additional tube is connected to the reference port (REF) and placed outside the wind tunnel. The pressure scanners used in this master's thesis have the unique serial numbers; 179, 180, 181 and 182.

5.1.3 Cobra Probe

The Cobra Probe measures the wind data from the wind tunnel tests. In the user manual by *Turbulent Flow Instruments* (TFI), the Cobra probe is described as [50]:

> *The Cobra Probe is a multi-hole pressure probe that provides dynamic, 3-component velocity and local static pressure measurements in real-time. The Cobra Probe is capable of a linear frequency-response from 0 Hz to more than 2 kHz and is available in various ranges for use between 2 m/s and 100 m/s. It can measure flow angles in a ±45° cone, all six Reynolds stresses and allows calculation of other higher order terms.*

Figure 5.6 illustrates the Cobra Probes main features, while Figure 5.7 and Table 5.1 gives an overview of the components needed in order to get the measurements of the wind data from the Cobra probe.

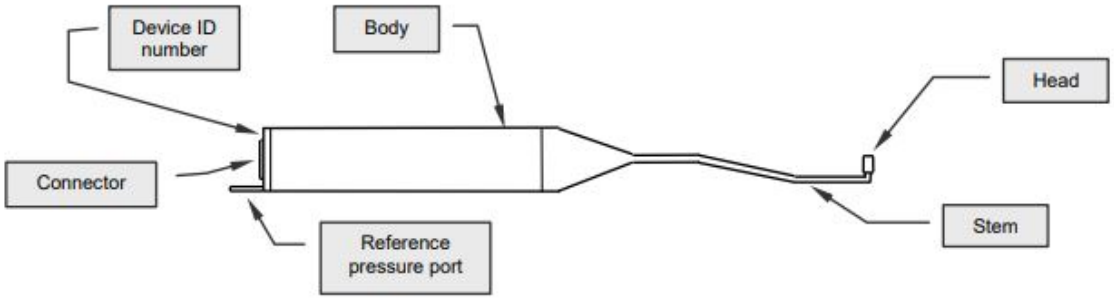


Figure 5.6: Series 100 Cobra Probe main features [50].

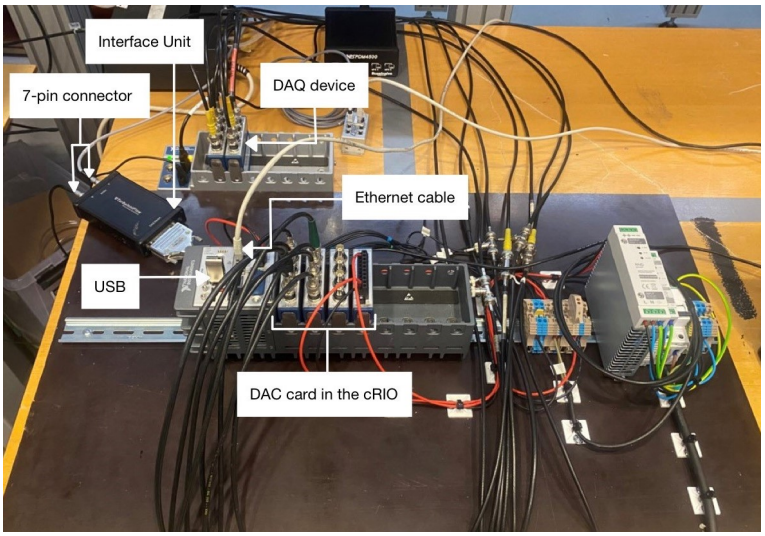


Figure 5.7: The overview of the Cobra Probe setup.

Component	**Purpose**
Series 100 Cobra Probe	Measures the wind data from the wind tunnel test. Connected to the interface unit via the 7-pin connector that is connected to two groups of four on the DAQ device, labeled A1-A4 and B1-B4.
DAQ device	Convert analogue data signals to a digital format.
DAC card placed in the cRIO	The Cobra Probe is connected to the DAC card through the DAQ device. It is important to insert the cables A1-A4 and B1-B4 in the right order from 0-3 in the DAC card.
USB stick	Used to store the data from the cRIO.
Ethernet cable	Provides communication between the wiring closet and the computer.
Reference pressure cable	The tube is placed on the reference pressure port on the Cobra Probe and put outside the wind tunnel.

Table 5.1: Overview of the components used for the Cobra Probe setup and their purpose.

5.1.4 Pitot Probe

The Pitot Probe is used to measure the wind velocity inside the wind tunnel. There are two holes in the Pitot Probe. One hole measures the stagnation pressure, while the other measures the static pressure. In order to obtain the wind speed, the velocity pressure can be used. This is calculated by measuring the difference between the stagnation and static pressure.

5.2 Wind Tunnel Tests

A series of tests were executed on the twin-box model in the wind tunnel. The tests were performed with different wind velocities, turbulence flow, motions, Cobra Probe distance (CPD) and with and without railings. An overview of the various tests performed in the wind tunnel is presented in Table 5.2.

Vortex shedding					
Model config.	**Motion**		**Wind [m/s]**	**RPM**	**Grid Rotation**
Without railings	None		Approx. 0-10		Still open grid
			Approx. 0-10		7 Hz
			Approx. 0-10		0.5 Hz
Static tests					
Model config.	**Motion**		**Wind [m/s]**	**RPM**	**Grid Rotation**
Without railings	Linear quasi steady 10 deg		6, 7, 9, 10	220, 260, 330, 370	Still open grid
			6, 7, 9, 10	280, 325 410, 450	7 Hz
			6, 7, 9, 10	260, 310, 400, 440	0.5 Hz
Admittance tests					
Model config.	**Motion**		**Wind [m/s]**	**RPM**	**Grid Rotation**
Without railings	Still -5 deg/+5 deg		7, 9	260, 330	Still open grid
	Still -2 deg/+2 deg		7, 9	260, 330	Still open grid
	Still 0 deg		7, 9	260, 330	Still open grid
	Still -5 deg/+5 deg		7, 9	325, 410	7 Hz
	Still -5 deg/+5 deg		7, 9	310, 400	0.5 Hz
	Still -2 deg/+2 deg		7, 9	325, 410	7 Hz
	Still -2 deg/+2 deg		7, 9	310, 400	0.5 Hz
	Still 0 deg		7, 9	325, 410	7 Hz
	Still 0 deg		7, 9	310, 400	0.5 Hz
With railings	Still -5 deg/+5 deg		7, 9	260, 330	Still open grid
	Still -2 deg/+2 deg		7, 9	260, 330	Still open grid
	Still 0 deg		7, 9	260, 330	Still open grid
	Still -5 deg/+5 deg		7, 9	325, 410	7 Hz
	Still -5 deg/+5 deg		7, 9	310, 400	0.5 Hz
	Still -2 deg/+2 deg		7, 9	325, 410	7 Hz
	Still -2 deg/+2 deg		7, 9	310, 400	0.5 Hz
	Still 0 deg		7, 9	325, 410	7 Hz
	Still 0 deg		7, 9	310, 400	0.5 Hz
Flow tests					
Model config.	**Motion**	**CPD**	**Wind [m/s]**	**RPM**	**Grid Rotation**
Without railings	None	55 mm	9	410, 400	7 Hz, 0.5 Hz
Without railings		105 mm	9	410, 400	7Hz,0.5 Hz
Without railings		220 mm	9	410, 400	7Hz,0.5 Hz
Without railings		500 mm	9	410, 400	7Hz,0.5 Hz
Without railings		1000 mm	9	410, 400	7Hz,0.5 Hz

Table 5.2: The various tests performed in the wind tunnel.

Before the testing started, the TMD were manually tuned by tapping the girder gently and pulling the TMD to get the same natural frequency as the section model. A still air test was done before every test as a reference and control check. The RPM had to be increased and adapted to get the desired wind velocity when the active grid generated turbulence. This is because the turbulence will reduce the wind velocity.

5.2.1 VIV Tests

VIV tests are used to detect vibrations due to vortex shedding. The tests are performed by gradually increasing the wind velocity while the section model is fixed in the neutral position. Vibrations can be observed when the bridge is in resonance, and the natural frequency corresponds with the vortex shedding frequency. The wind velocity causing vibrations will be avoided in the reaming tests due to disruption of the measured loads. No vibrations due to vortex shedding were observed during the VIV test, and therefore it was not necessary take this into consideration.

5.2.2 Quasi-Static Tests

Quasi-static tests are executed to estimate the static coefficients for lift, drag, and moment. The tests is performed with different wind velocities; 6, 7, 9 and 10 m/s, to indicate Reynolds number dependency. The angle of attack is continually changed as the section model is quasi-statically rotated with a max amplitude of 10 degrees. For comparison of the measurements, both the loads cells and pressure scanners were used.

5.2.3 Admittance Tests

The aerodynamic admittance functions can be estimated by performing admittance tests in the wind tunnel. The tests are executed with different wind velocities, angles of attack and turbulent flow. In addition, the tests are done with and without railings on the section model to indicate the impact they have on the measurements. Turbulence components, mean wind and surface pressure are measured during the test.

5.2.4 Flow Tests

Flow tests were performed with and without the section model inside the wind tunnel. The Cobra Probes were placed at different distances of 55 mm, 105 mm, 220 mm, 500 mm and 1000 mm. This is to indicate turbulence correlation in the spanwise direction. These tests may be used to estimate 3D aerodynamic admittance functions. This thesis does not focus on estimating the 3D AAF, and therefore, the measurements were not further used.

5.3 Accuracy and Error Sources

There may be several errors in the experimental setup, which can lead to inaccurate results. As mentioned in Section 5.1, a test was conducted with a table fan before the final wind tunnel testing. This was done to exclude errors with the instrument setup, the pressure scanners, and the tube system. After testing, an error was detected on tube no. 1115 and 2115. Both had extremely high-pressure measurements compared to the other tubes. They are placed at the same location at strip one, on separate boxes. After investigating various reasons for the unexpected error, it was concluded that both tubes were squeezed due to being too close to the aluminium pipe. At the location of the tubes, there was expected less variation in the pressure distribution. In addition, since the pressure tubes are evenly distributed and have a relatively short distances between them on strip one, it was determined to disregard the measurements from these tubes. When the pressure distribution was calculated, it was therefore chosen to set the mean value of the neighbouring tubes on both tubes no. 1115 and 2115.

Other sources of error could occur from the mounting of the twin-box bridge in the wind tunnel and the cross-section itself. A level meter was used to position the sections correctly. Since this is done manually, there may have be some deviations. Further, it is uncertain if the 3D-printed parts were completely sealed. Even though this was accounted for when designing the parts, wind may have entered the cross-sections and could, in that case, affect the flow pattern and the pressure distribution. In addition, duck tape was used to seal the 3D-printed parts and the rest of the cross-sections. There were some uneven transitions and uncertainty if it was completely sealed. This could eventually affect the wind flow and further the pressure distribution. Moreover, the placement of the Cobra Probe and the Pitot Probe may affect the wind flow in front of the twin-box.

Dynamic amplification and signature turbulence are sources that could disturb the signal. Dynamic amplification can be observed as peaks in plots of the power spectral density with vibrations around the eigenfrequency of the cross-section. Signature turbulence may be formed by vortex shedding and will affect the buffeting response of the twin-box. In addition, electrical noise and lose components in the wind tunnel can lead to unwanted noise and vibration of the measured signals.

Another source of error is the reference pressure for the Cobra Probes and the Pitot Probe. Changes in the pressure inside the laboratory can occur, which could affect the measured reference pressure. For example, if a door to the laboratory closes, this may lead to changes in the reference pressure.

5.4 Post Processing

The measurements from each test in the wind tunnel were saved in separate files for the MPS pressure scanners, the Cobra Probes, the Pitot Probe, and the load cells. The data acquisition system and measurements were measured at a sampling rate of 200 Hz per second. The supervisors processed the raw data to processed data, while further data processing was done in Spyder.

5.4.1 Pressure Measurement

The purpose of the pressure tubes was to estimate the aerodynamic forces around the periphery of the cross-section for different wind velocities, angles of attack, and flow conditions. The pressure tubes measures the pressure perpendicular to the surface of the bridge. Raw positive pressure is defined as pushing pressure and is faced inward on the cross-section, while negative raw pressure is defined as suction and is faced outward. The mean value of a time series of 5 minutes was used to obtain a static pressure and load representation. Further, to ensure times series with minimal disturbance, the time series of the pressure measurements were cut at the start and end. Two different methods were used to calculate the aerodynamic forces of the twin-box bridge; the piece-wise load method and the interpolated method. Appendix B shows the Spyder scripts for both methods.

Figure 5.8 shows the surfaces on the cross-section for the estimation of the forces and moments.

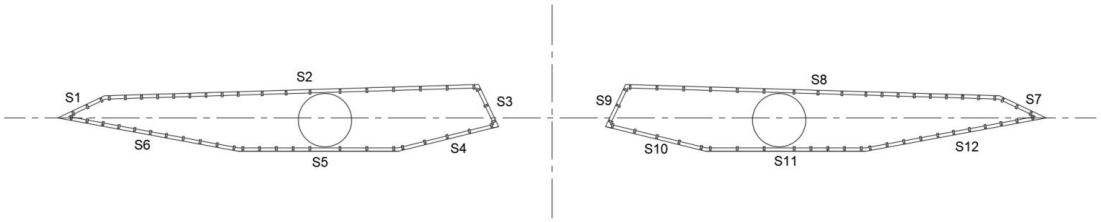
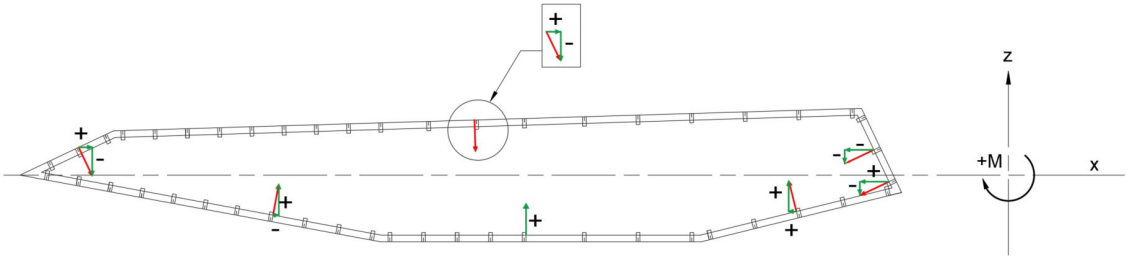
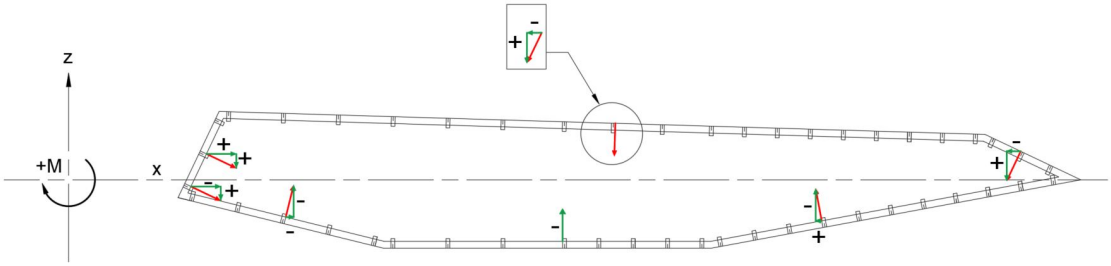


Figure 5.8: Definitions of surfaces on the bridge.

In addition, for the calculation of moments, Figure 5.9 illustrates the definitions of positive and negative forces.



(a)



(b)

Figure 5.9: Illustration for the moment calculations on (a) the upstream-box and (b) the downstream-box.

The piece-wise load method assumes that each pressure tube is a point load. Each point load is multiplied by the surface area to obtain the loads around the cross-section. The width of the calculated surface area is the sum of the distance between the two neighbouring pressure tubes in the same correlation line. If the pressure tube is located at the end of a surface, the width is set to the sum of the distance to the end of the surface and half the distance to the neighbouring pressure tube. Figure 5.10 illustrates widths on the top surface of the upstream-box associated with the pressure tubes. The length of the surface area is set to 1 meter.

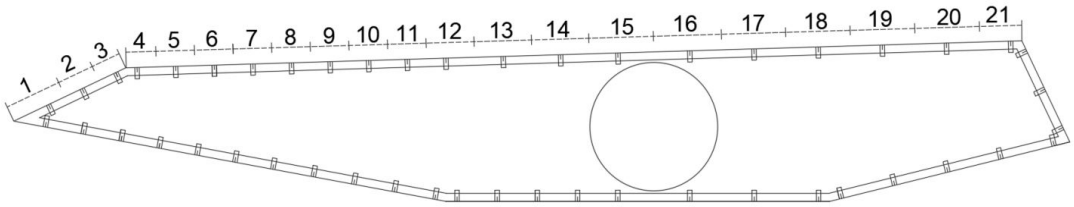


Figure 5.10: Illustration of the widths on the to surface of the upstream-box

Figure 5.11 and 5.12 shows an example of the point pressure and point load obtained by the piece-wise load method.

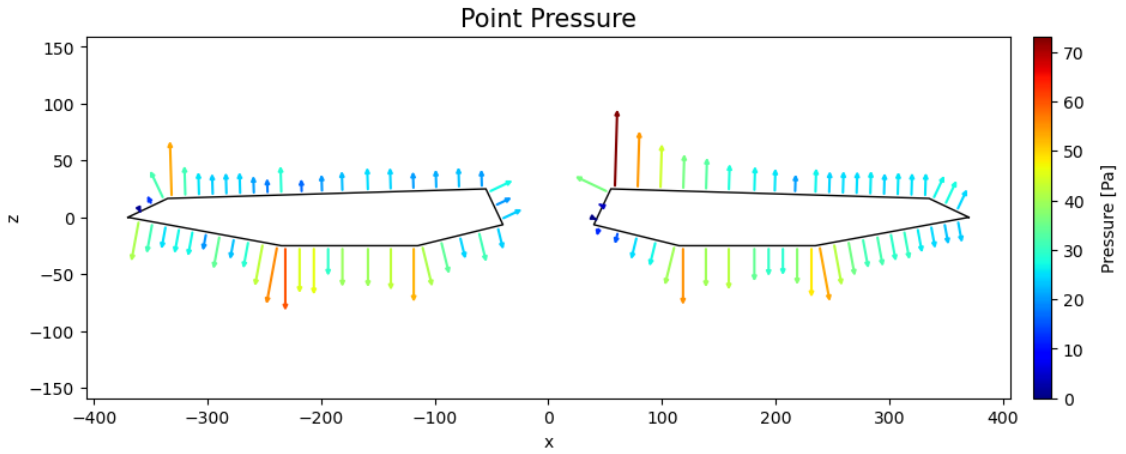


Figure 5.11: Example of point pressure by the piece-wise load method.

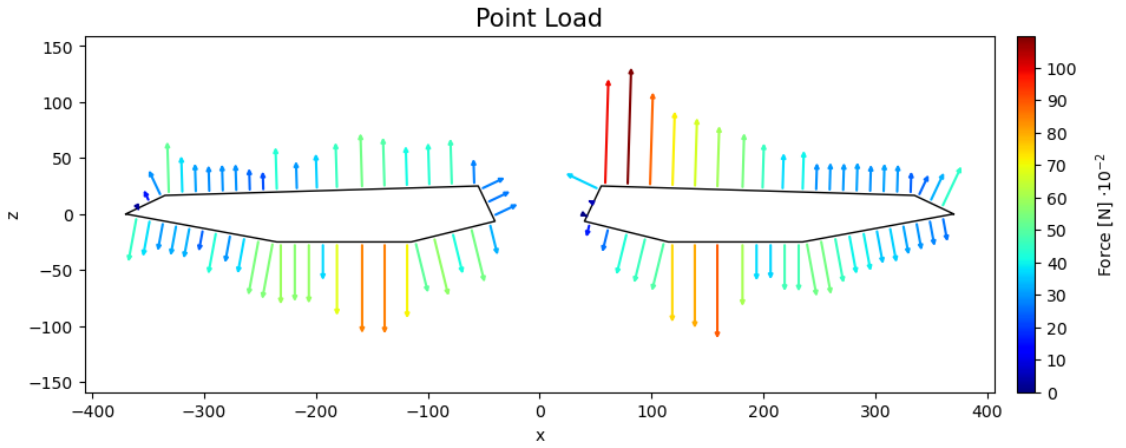


Figure 5.12: Example of point load obatined by the piece-wise load method.

The interpolated load method estimates the forces and moments as a distributed load on the surface of the cross-section. To obtain the distributed pressure, query points with equal spacing of 1 mm are arranged around the cross-section. The pressure is further interpolated with a piece-wise cubic polynomial function in Spyder. All the forces are multiplied with the same surface area with a width of 0.001 m and a length of 1 m. The pressure on the ends of the surfaces is, however, multiplied by half the width. Figure 5.13 and 5.14 shows an example of an interpolated pressure and load distribution.

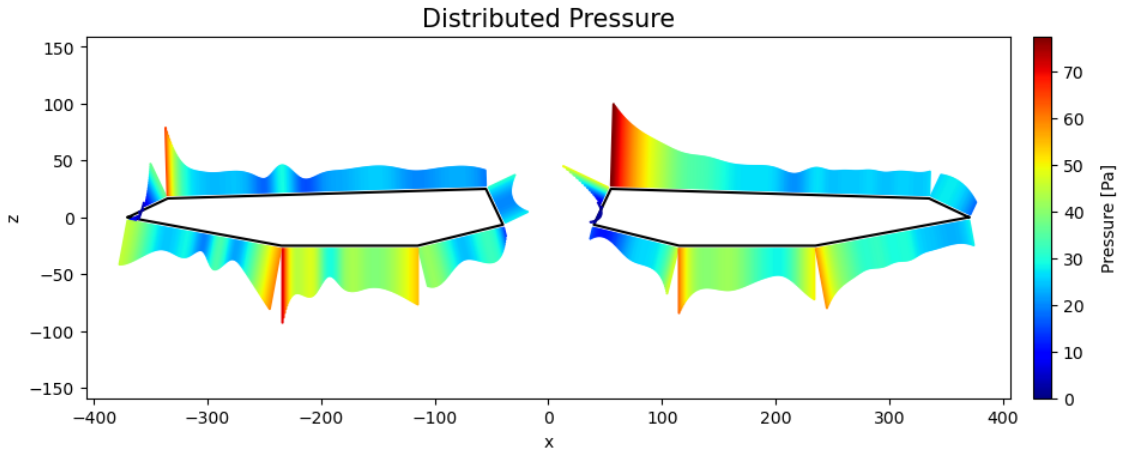


Figure 5.13: Example of an interpolated pressure distribution by the interpolated load method.

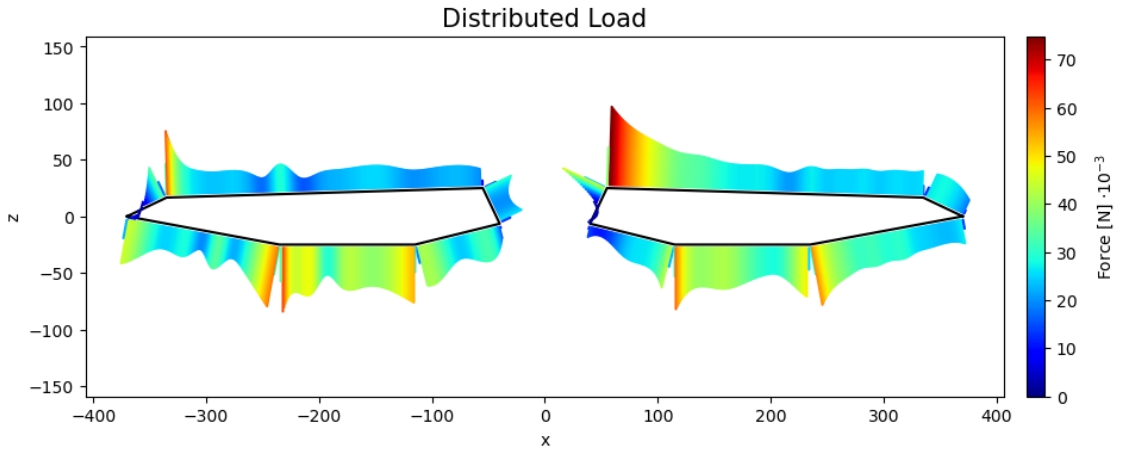


Figure 5.14: Example of an interpolated load distribution by the interpolated load method.

5.4.2 Wind Data

The sampled data from the Cobra Probe was converted from voltage to velocity by Associate Professor Øyvind Wiig Petersen. The fluctuating part of the wind is used to estimate the spectral density. This is obtained by subtracting the mean signal from the times series of the wind. The wind data is transferred to the frequency domain to estimate the aerodynamic admittance functions. The Welch's method is used in Spyder for calculating the power spectral density using 10 Hamming Windows.

5.4.3 Force Spectra

The data obtained from the MSP pressure scanners are zeroed by subtracting the mean of the signal. The time series of the loads obtained from the pressure measurements with the methods described above are transformed from the time domain to the frequency domain by Welch's method using 10 Hamming windows. The spectral densities of drag, lift and moment for the different correlation lines are plotted with logarithmic axes. Equation 3.80 is used to find the coherence between the lines.

5.4.4 Admittance Estimation

The aerodynamic admittance functions are estimated in the frequency domain with three different methods; the general, the equivalent and the cross-spectral, as described in Section 3.5. The estimated functions are plotted with the Sears function for comparison with logarithmic axes. The script used to calculate the aerodynamic admittance functions is attached in Appendix C.

Chapter 6

Results and Discussions

6.1 Turbulence Spectra

The wind spectra are studied before the admittance functions are estimated. This is important in order to detect possible errors that could affect the final estimation of the admittance functions. The turbulence spectra are used to analyze the wind flow. Figure 6.1 and 6.2 presents the horizontal and vertical spectra of three different wind flows, still open grid and grid rotations of 0.5 Hz and 7 Hz.

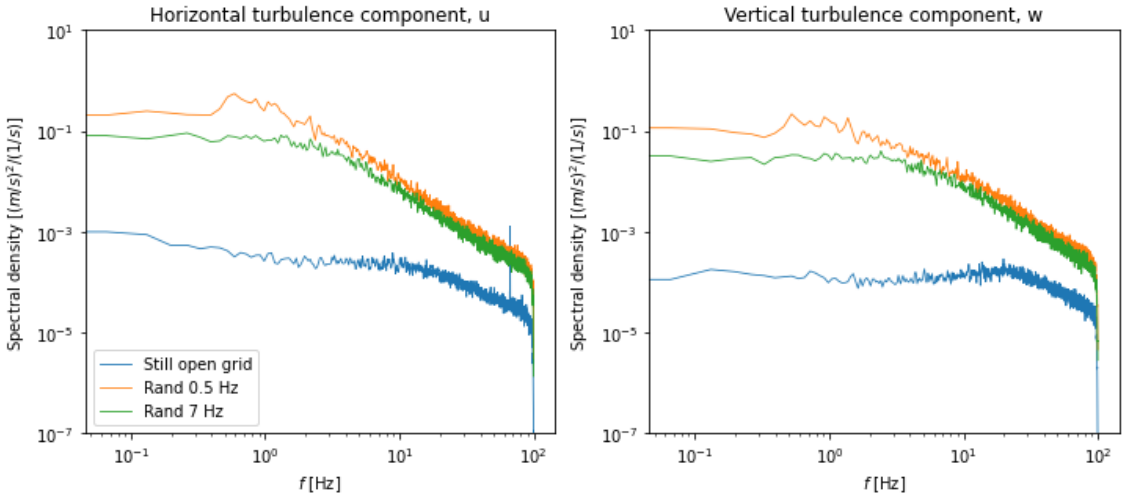


Figure 6.1: Turbulence spectra for the horizontal and vertical component, $V \approx 7m/s$.

As expected, the turbulent flow with grid rotations of 0.5 Hz and 7 Hz has higher spectral densities than the flow with still open grid open. When the grid is still open, the flow display uniform tendencies, as seen in Figure 6.1. It can also be observed that the spectral densities decrease gradually as the frequency increases. A small peak for the horizontal turbulence component, u, is detected for the still open grid turbulence. The peak could be caused by vibrations from the Cobra probe at one of the natural frequencies of the Cobra Probe mount, or by electric noise. The turbulence flow with grid rotation of 0.5 Hz displays higher spectral densities than with grid rotation of 7 Hz.

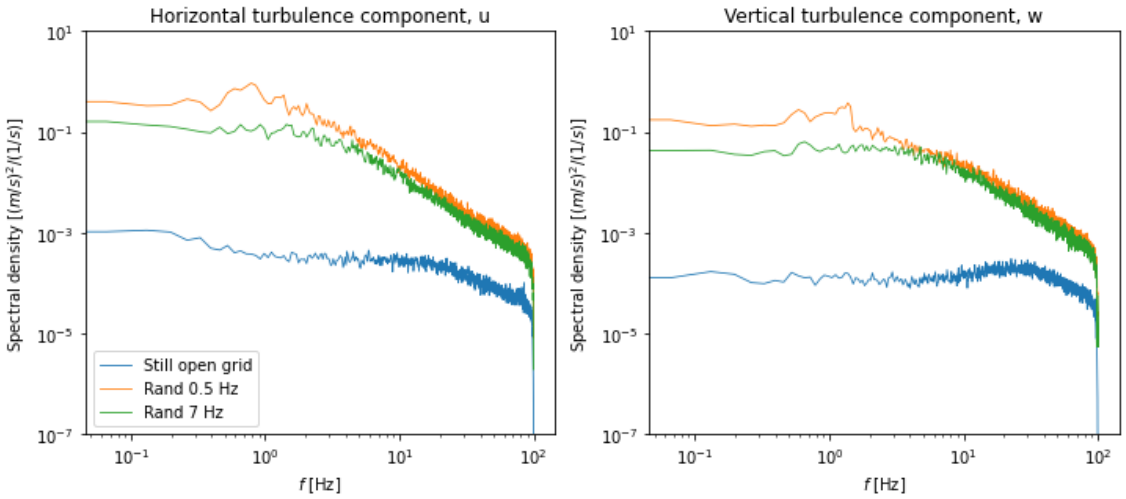


Figure 6.2: Turbulence spectra for the horizontal and vertical component, $V \approx 9m/s$.

Figure 6.2 presents the turbulence spectra with the wind velocity, $V \approx 9m/s$. The turbulence spectra show great similarity compared to the turbulence spectra with wind velocity, $V \approx 7m/s$. A small difference is detected for the spectra of turbulent flow, where the spectral density gets a little higher as the wind velocity increases. The spectral density of the still open grid is almost unchanged with the increased wind velocity.

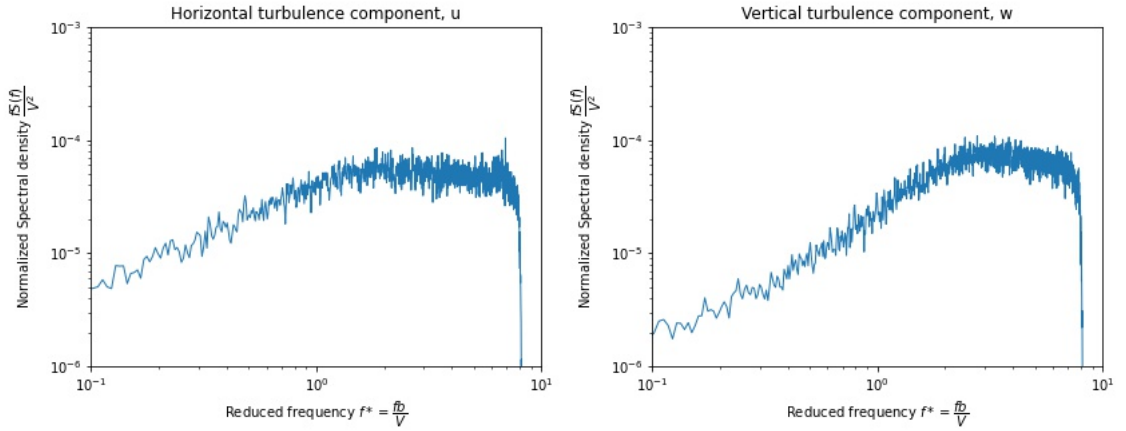


Figure 6.3: Turbulence spectra for the horizontal and vertical component with still open grid, $V \approx 7m/s$.

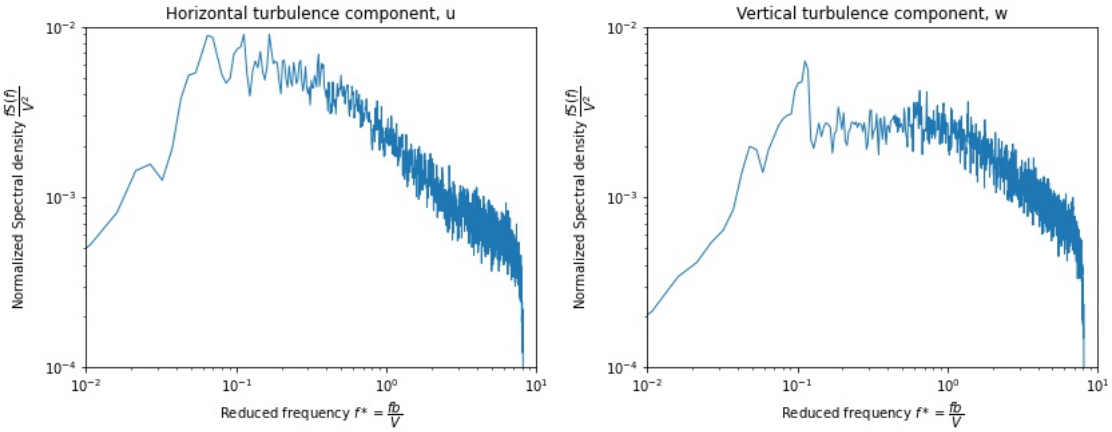


Figure 6.4: Normalized wind spectra with grid rotation of 0.5 Hz, $V \approx 9m/s$,

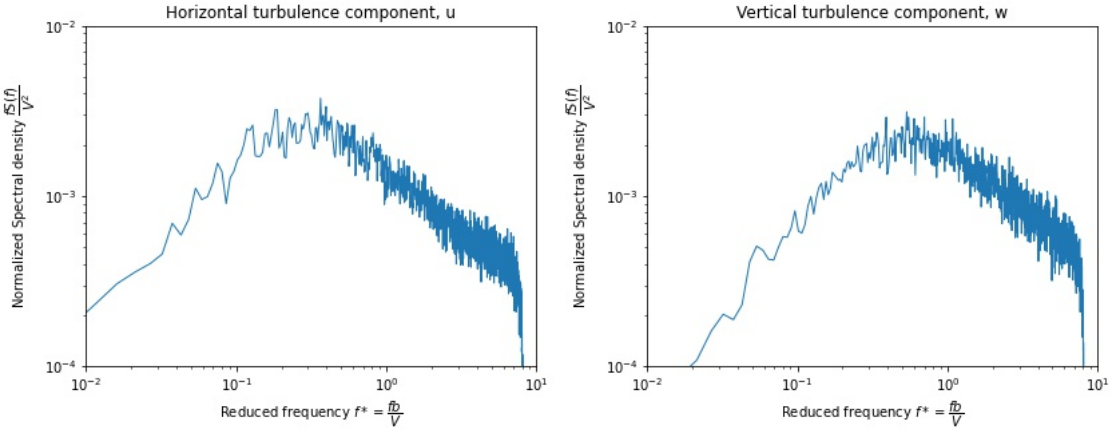


Figure 6.5: Normalized wind spectra with grid rotation of 7 Hz, $V \approx 9m/s$.

Figure 6.3, 6.4 and 6.5 presents the normalized spectra for horizontal and vertical direction for the three different wind flows. The horizontal and the vertical spectra for the still open grid have a maximum peak at approximately 3 Hz. The vertical spectra for the turbulent flow have a maximum peak around the reduced frequency 1 Hz. The horizontal spectra seem to have a maximum peak at a slightly lower reduced frequency. When studying the turbulence spectra and comparing them to the turbulence spectra obtained by Larose [11] and Wang et al. [23], similarity is observed.

The turbulence intensities are calculated by Equation 3.40 and shown in Figure 6.1 for still open grid, Figure 6.2 for a grid generated turbulence with grid rotation of 0.5 Hz, and Figure 6.3 with grid rotation of 7 Hz. The longitudinal turbulence intensity, I_u, and the lateral intensity, I_w, are equal for still open grid. However, for grid-generated turbulence, the longitudinal turbulence intensity, I_u, is the maximum. The turbulence intensity with a still open grid is approximately the same for the different wind velocities but has a small decrease with an increase in wind velocity. This indicates that the mean wind velocity increases more than the fluctuating components. However, for grid-generated turbulence, the turbulence intensities increases with increased velocity.

V [m/s]	RPM	I_u	I_w
7	260	1,3 %	1,3 %
9	330	1,2 %	1,2 %

Table 6.1: Turbulence intensity with still open grid.

V [m/s]	RPM	I_u	I_w
7	310	14,3 %	11,4 %
9	400	14,7 %	11,3 %

Table 6.2: Turbulence intensity with grid generated turbulence, grid rotation 0.5 Hz.

V [m/s]	RPM	I_u	I_w
7	325	8,7 %	7,6 %
9	410	9,3 %	8,3 %

Table 6.3: Turbulence intensity with grid generated turbulence, grid rotation 7 Hz.

Figure 6.6 presents the normalized measured turbulence spectrum, in which the Kaimal spectrum given by Equation 3.90 is also plotted for validation and comparison. The horizontal position of the peaks in the Kaimal spectra is determined by the value of the integral length scale. However, in a wind tunnel experiment, it is difficult to determine the integral length scale. For the presented plots, the integral length scales are put equal to $xLu = 0.25m$ and $xLw = 0.021m$. It is found that the measured spectrum agrees well with the Kaimal spectrum as it shows a similar distribution of energy. The measured spectra are, therefore, a good representation of the turbulent wind and are acceptable for further calculations.

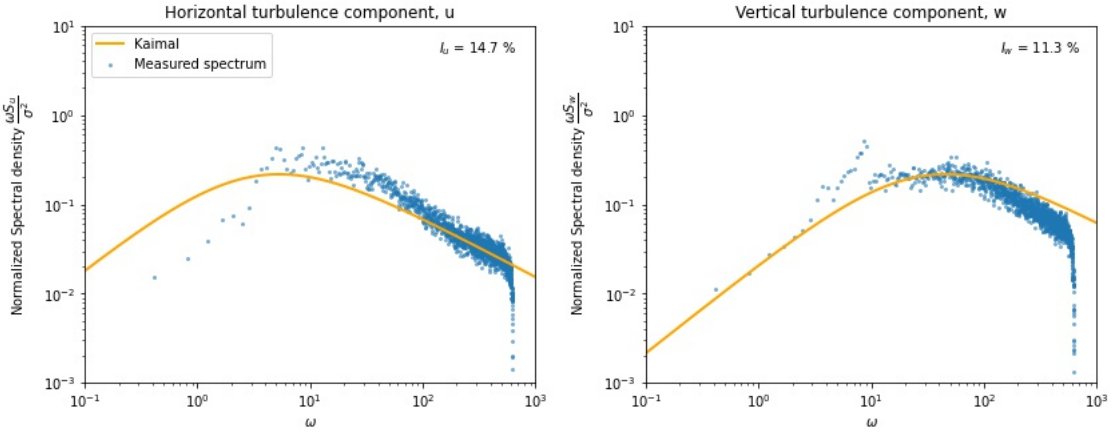


Figure 6.6: Normalized wind spectra with grid rotation 0.5 Hz together with the Kaimal spectra, $V \approx 9m/s$.

6.2 Pressure Distribution

This section presents the pressure distribution from the wind tunnel tests measured by the MPS4264 Pressure Scanners. A comparison of tests is done to detect the impact of the railings, and turbulent wind flow with different grid rotations is studied. Due to the dense distribution of the pressure holes, the first correlation line will be used. The remaining five correlation lines will be presented for one test to investigate the spanwise pressure correlation. At each pressure hole, the pressure distribution is given by the mean value of the time series. Each surface on the bridge cross-section is defined with names S1-S12, to make it easier to analyze and explain the pressure distribution on the different surfaces, as described in Subsection 5.4.1. Pressure pointing inwards is defined as positive pressure (pushing pressure), while pressure pointing outwards is defined as negative pressure (suction).

6.2.1 Still open grid

The tests presented below are executed with a still open grid. The tests are executed with the wind the velocities of 7 m/s and 9 m/s, and the angle of attack, α, varying between - 5°, - 2°, 0°, 2° and 5°. The pressure distribution is studied without the railings attached to the girder.

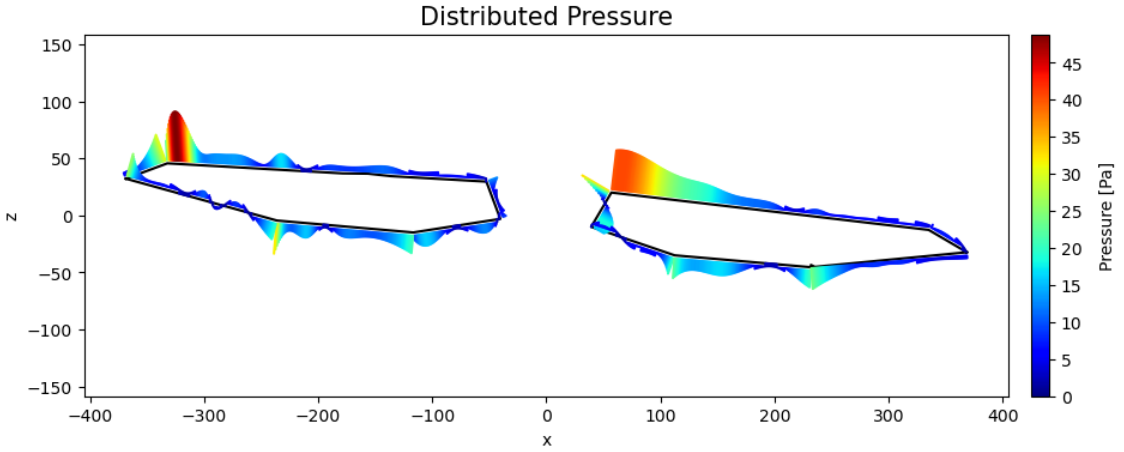


Figure 6.7: Distributed pressure with still open grid, RPM = 260, $V \approx 7m/s$ and $\alpha = 5°$.

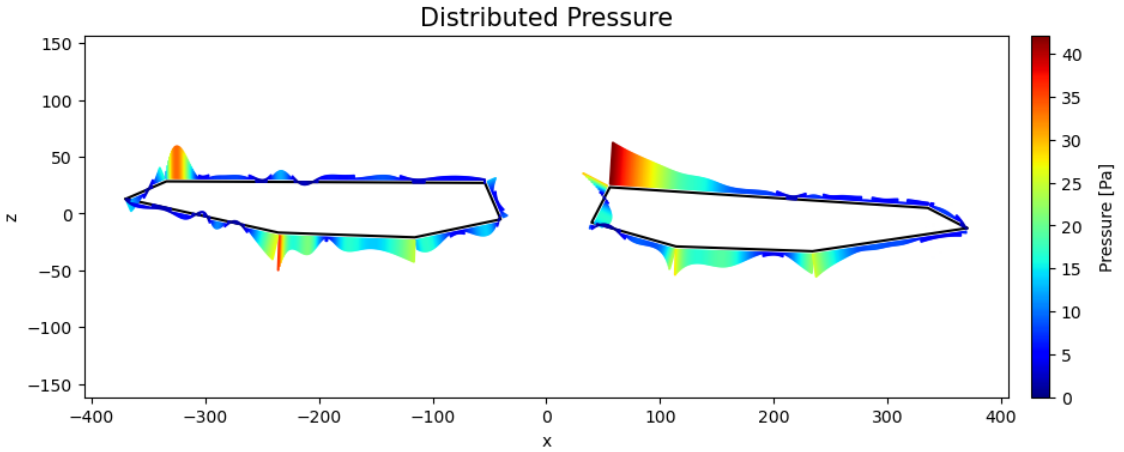


Figure 6.8: Distributed pressure with still open grid, RPM = 260, $V \approx 7m/s$ and $\alpha = 2°$.

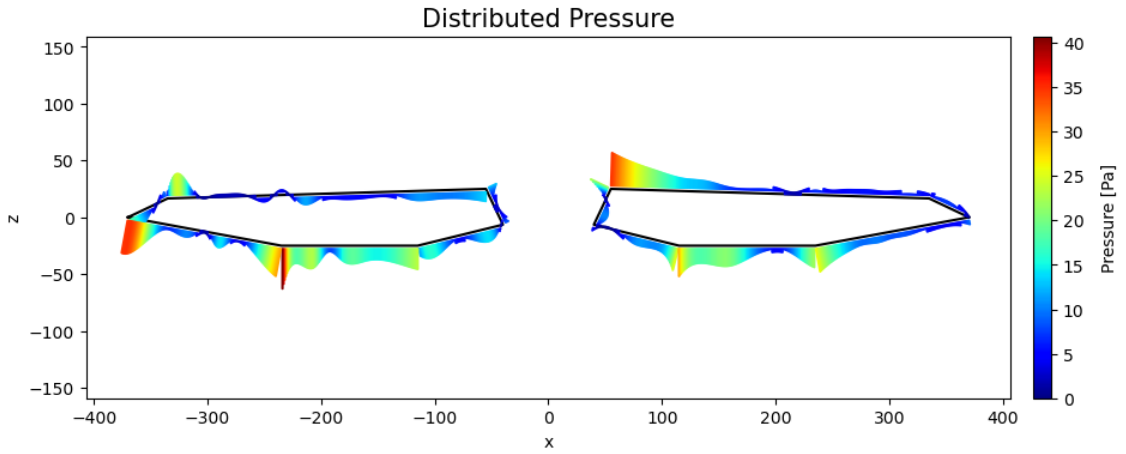


Figure 6.9: Distributed pressure with still open grid, RPM = 260, $V \approx 7m/s$ and $\alpha = 0°$.

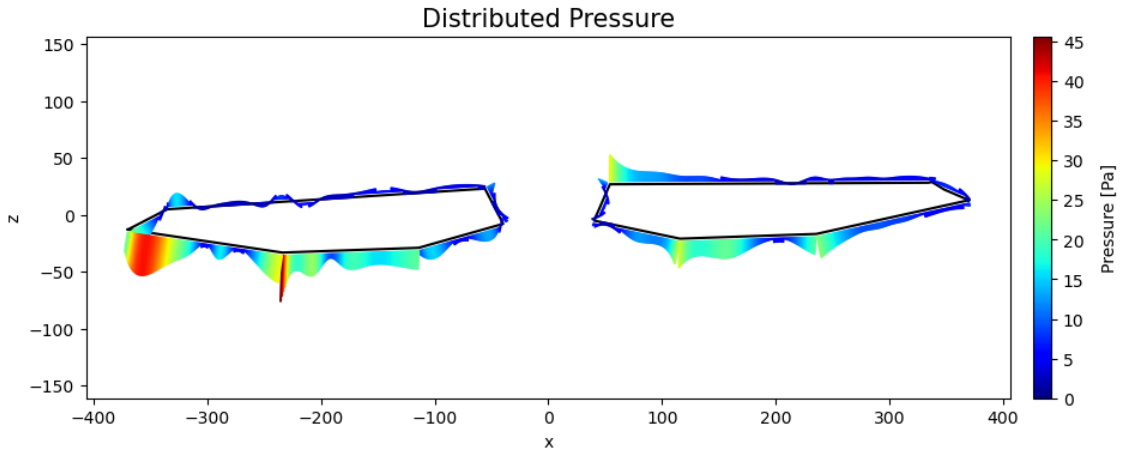


Figure 6.10: Distributed pressure with still open grid, RPM = 260, $V \approx 7m/s$ and $\alpha =$ - 2°.

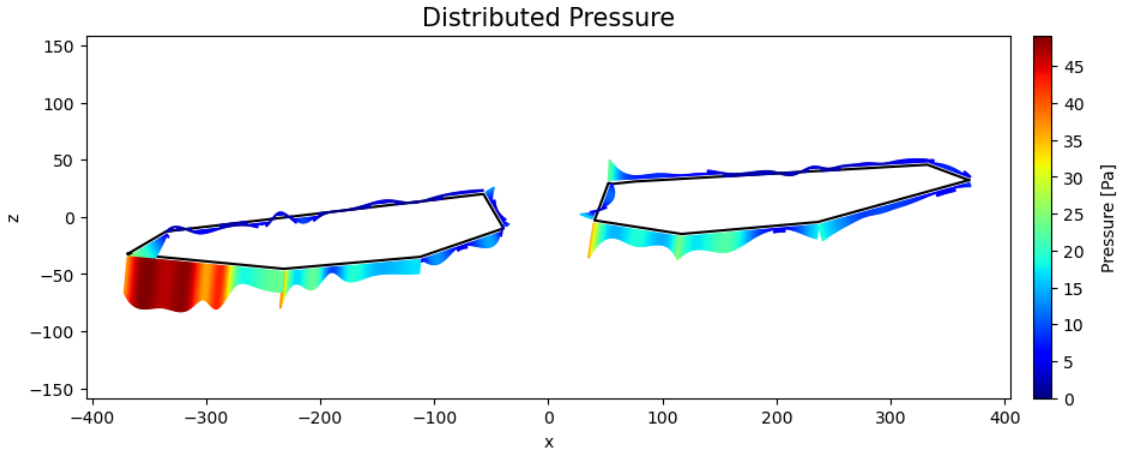


Figure 6.11: Distributed pressure with still open grid, RPM = 260, $V \approx 7m/s$ and $\alpha =$ - 5°.

Figure 6.7-6.11 presents the distributed pressure with wind velocity $7m/s$. The pressure distribution changes as the angle of attack changes. When comparing the pressure distributions with

angles of attack from 5° to −5°, a reduction of the negative pressure on the leading edge, S2 and a significant development of negative pressure on S6, is observed. The illustrations shows that the pressure distribution on the upstream-box changes from having maximum negative pressure on the upper surface, S2, to maximum negative pressure on the lower surface, S6. Further, negative pressure on S1 changes to positive pressure and increase gradually with decreased angle of attack. The pressure distribution on S5 is quite similar for all the angles of attack. However, the negative pressure increases slightly in the transition between S5 and S6. There are minor to no changes observed on S3 and S4 when the angle of attack changes.

For the downstream-box, the pressure distribution shows similarities for the five different angles of attack. The negative pressure on the leading edge of S8 decreases slowly when the angle decreases. On the corner between S9 and S10 positive pressure is observed. However, when angle of attack is −5°, negative pressure occurs, see Figure 6.11. For the angle of attack, 5°, some positive pressure is observed on S10 on the leading edge. The positive pressure turns negative towards the corner between S10 and S11 and remains negative on S11 and S12. The negative pressure on S11 and S12 remains approximately the same for angles of attack.

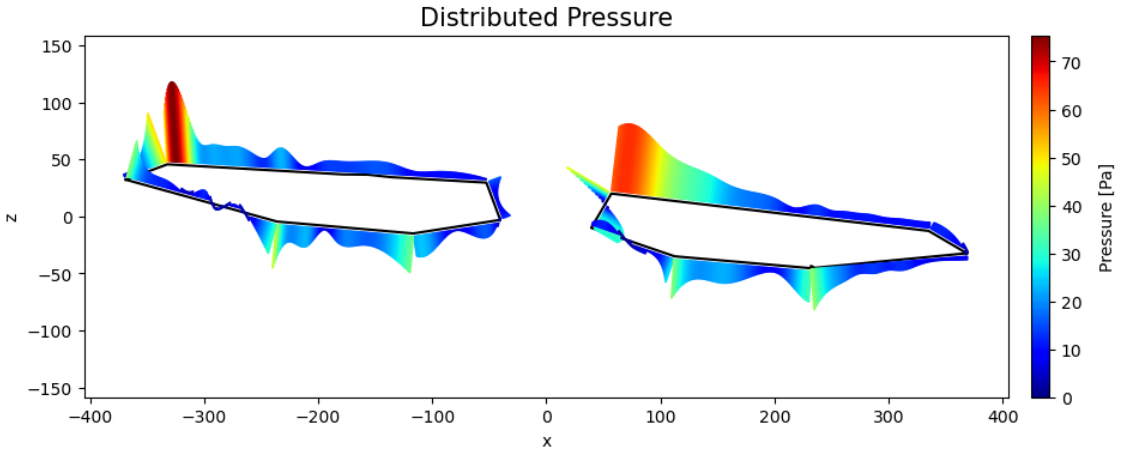


Figure 6.12: Distributed pressure with still open grid, RPM = 330, $V \approx 9m/s$ and $\alpha = 5°$.

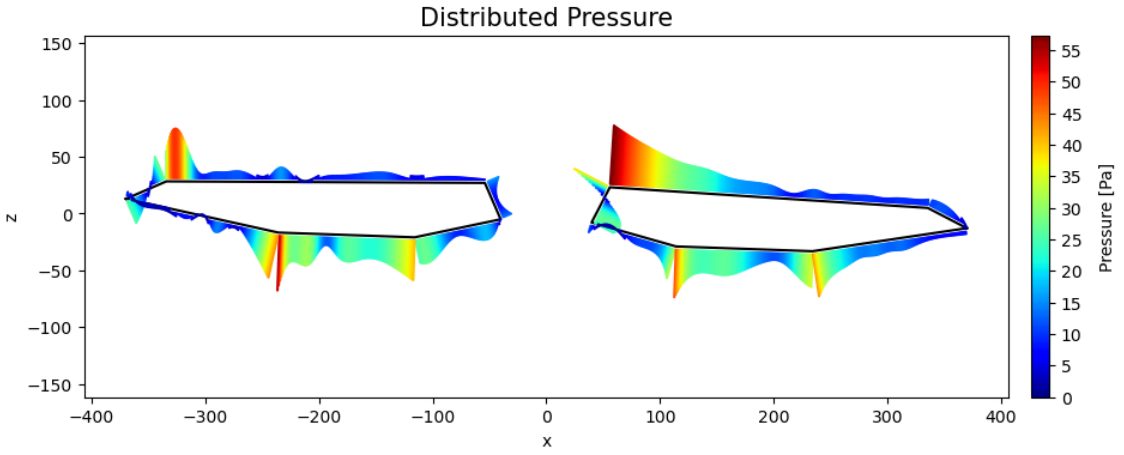


Figure 6.13: Distributed pressure with still open grid, RPM = 330, $V \approx 9m/s$ and $\alpha = 2°$.

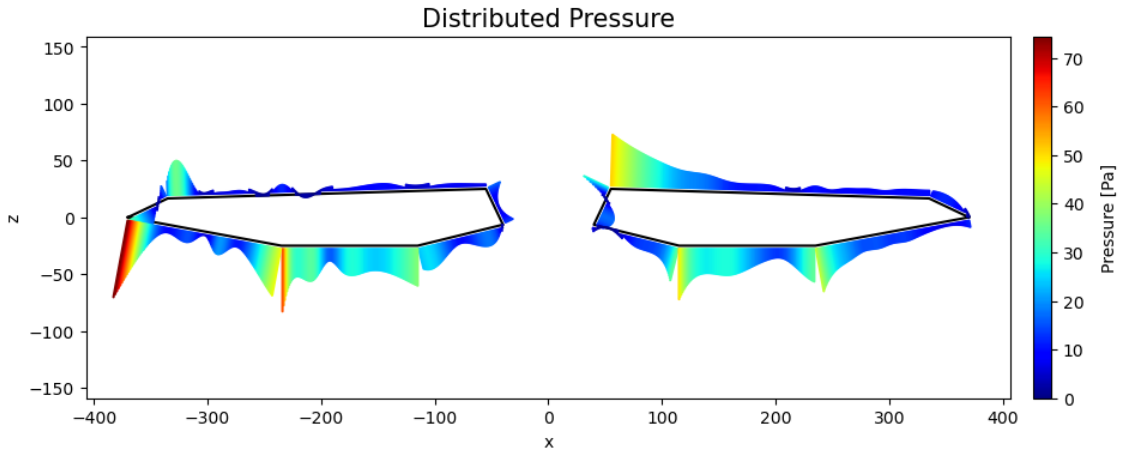


Figure 6.14: Distributed pressure with still open grid, RPM = 330, $V \approx 9m/s$ and $\alpha = 0°$.

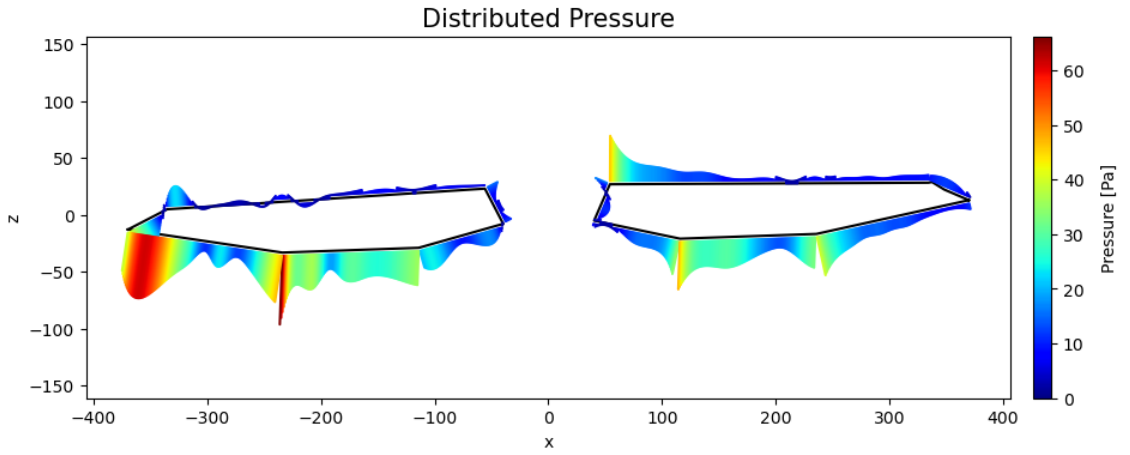


Figure 6.15: Distributed pressure with still open grid, RPM = 330, $V \approx 9m/s$ and $\alpha =$ - 2°.

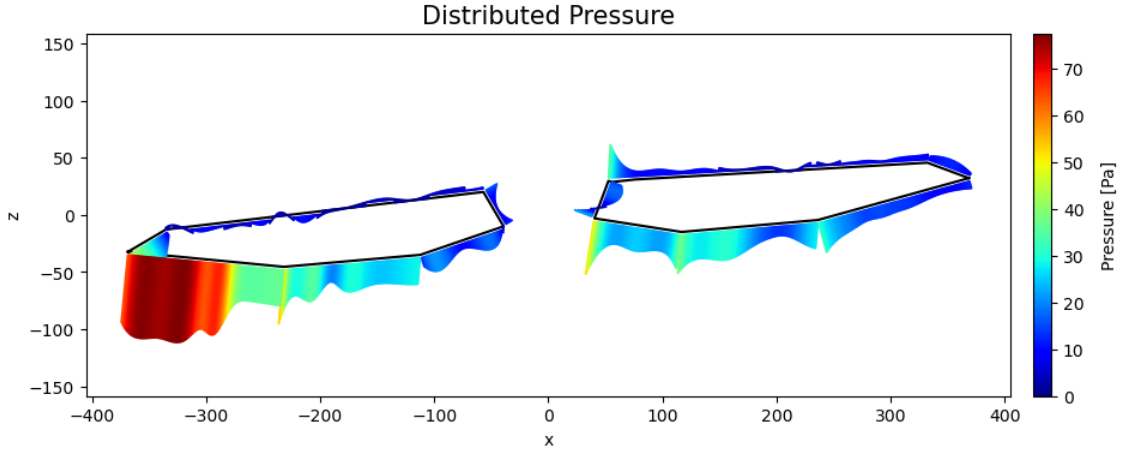


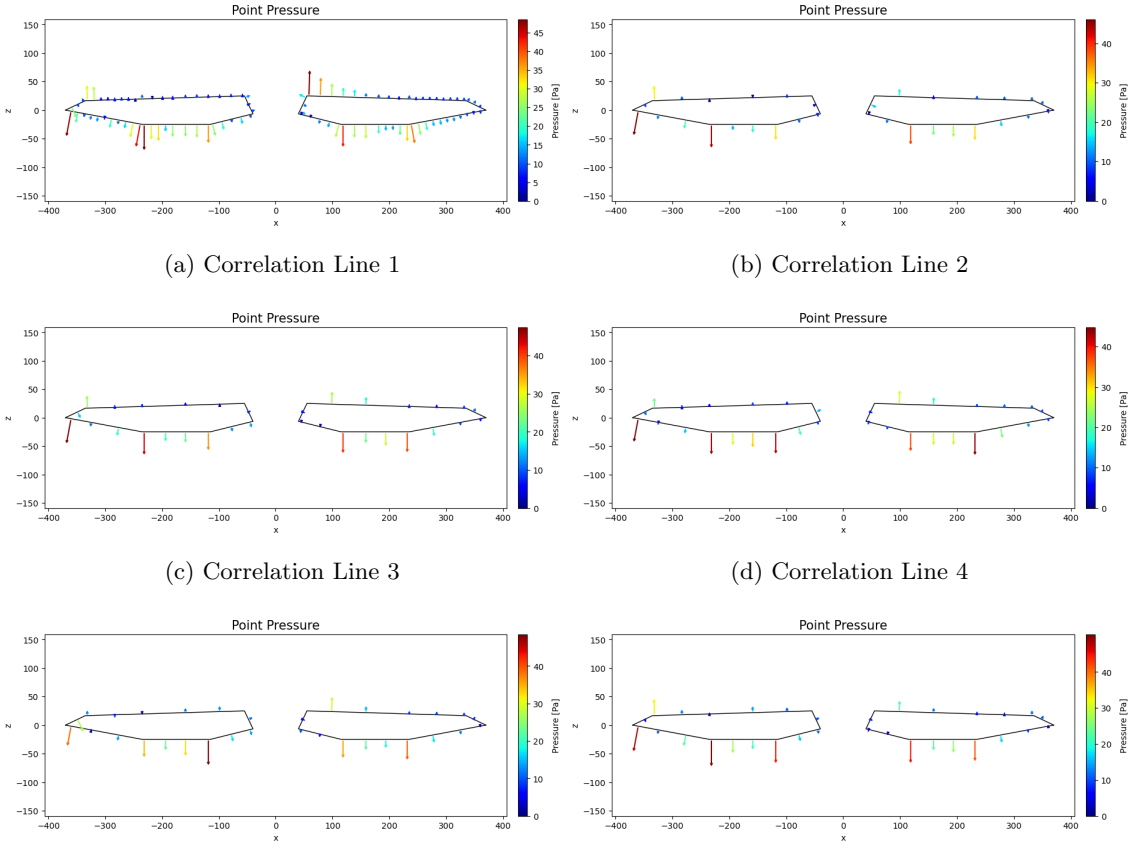
Figure 6.16: Distributed pressure with still open grid, RPM = 330, $V \approx 9m/s$, and $\alpha =$ - 5°.

Figure 6.12-6.16 presents the distributed pressure with wind velocity $9m/s$. When the twin-box is subjected to higher wind velocity, the distribution remains unchanged, but the magnitude increases.

The negative pressure on the upper surface, S2, of the upstream-box decreases as the angle of attack decreases. The negative pressure becomes more dominant at the bottom surfaces, where the maximum amount of negative pressure occurs on S6 on the leading edge, when $\alpha = -5°$, as seen in Figure 6.16. Similar observations are done for the downstream-box, where the maximum negative pressure changes from being on the upper surface to becoming more dominant at the bottom surfaces.

Pressure Distribution for the six Correlation lines

Figure 6.17 illustrates the correlation of pressure distribution in the span-wise direction. Some minor changes can be seen for correlation line five on S5, where the negative pressure is more dominant towards the corner between S5 and S4. This differs from the other correlation lines where the negative pressure is more dominant towards the corner between S5 and S6. Otherwise, no significant differences are observed when comparing the pressure distribution for the different correlation lines. This indicate that the pressure remains approximately the same as the separation distance between the lines increases. Due to a lower number of pressure tubes with a larger separation for correlation lines 2-6, the pressure distribution may be more inaccurate than for correlation line one.



(a) Correlation Line 1

(b) Correlation Line 2

(c) Correlation Line 3

(d) Correlation Line 4

(e) Correlation Line 5

(f) Correlation Line 6

Figure 6.17: Point pressure for the six correlation lines with still open grid, RPM = 330, $V \approx 9m/s$ and $\alpha = 0°$.

6.2.2 Turbulent wind flow

The pressure distribution from the test executed with turbulent wind flow is presented below. The tests are performed with the grid rotations of 7 Hz and 0.5 Hz and the wind velocity 7 m/s and 9 m/s. This section presents tests with the angles of attack of 5°, 0°and - 5°. Figure 6.18-6.23 presents the pressure distribution of the twin-box with a wind velocity of 9 m/s for the two different grid-generated turbulence's. In addition, the pressure distribution on the six correlation lines is presented for a test with a grid rotation of 7 Hz and wind velocity 9 m/s.

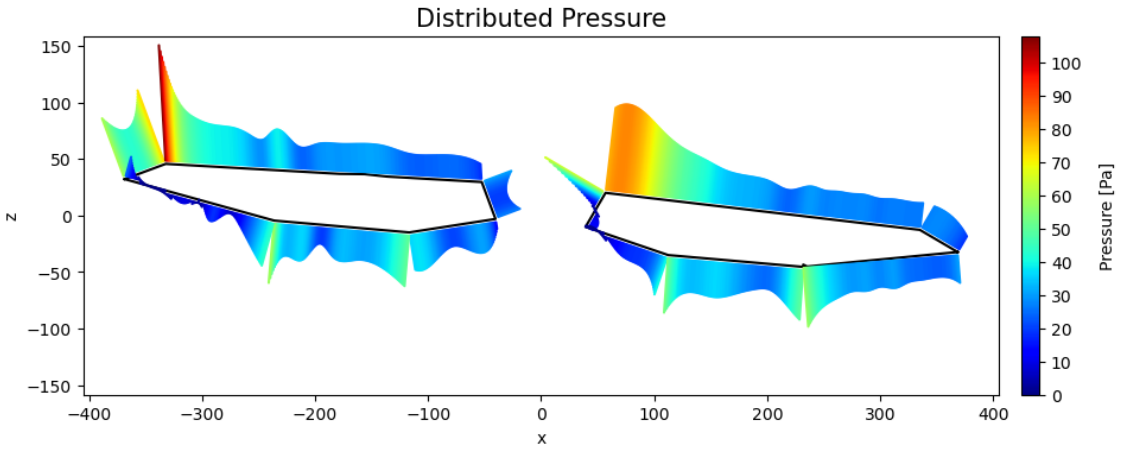


Figure 6.18: Distributed pressure with grid rotation 7 Hz, RPM = 330, $V \approx 9m/s$ and $\alpha = 5°$.

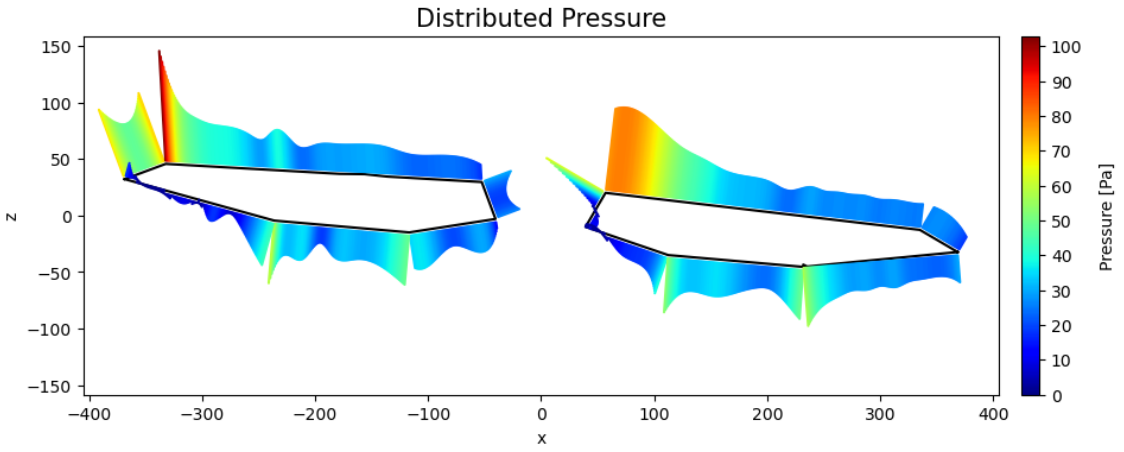


Figure 6.19: Distributed pressure with grid rotation 0.5 Hz, RPM = 330, $V \approx 9m/s$ and $\alpha = 5°$.

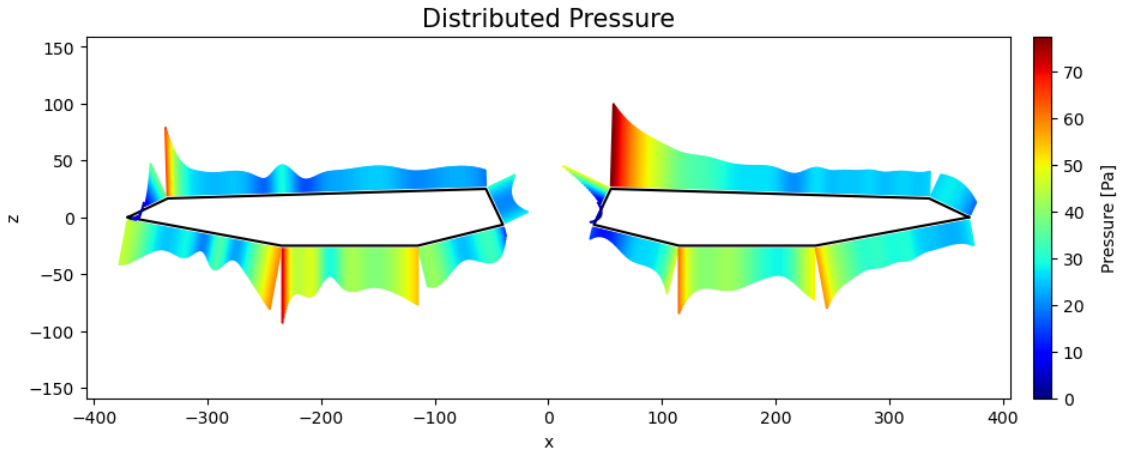


Figure 6.20: Distributed pressure with grid rotation 7 Hz, RPM = 330, $V \approx 9m/s$ and $\alpha = 0°$.

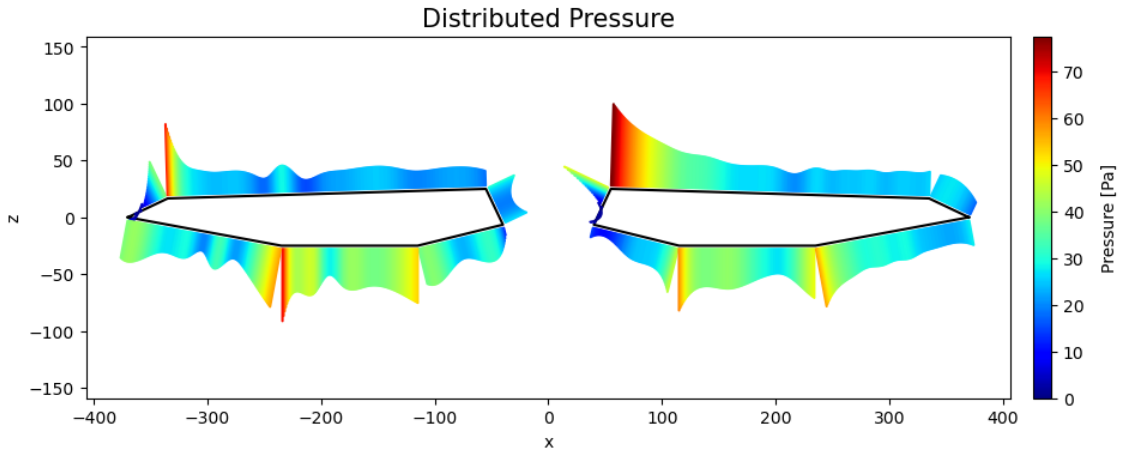


Figure 6.21: Distributed pressure with grid rotation 0.5 Hz, RPM = 330, $V \approx 9m/s$ and $\alpha = 0°$.

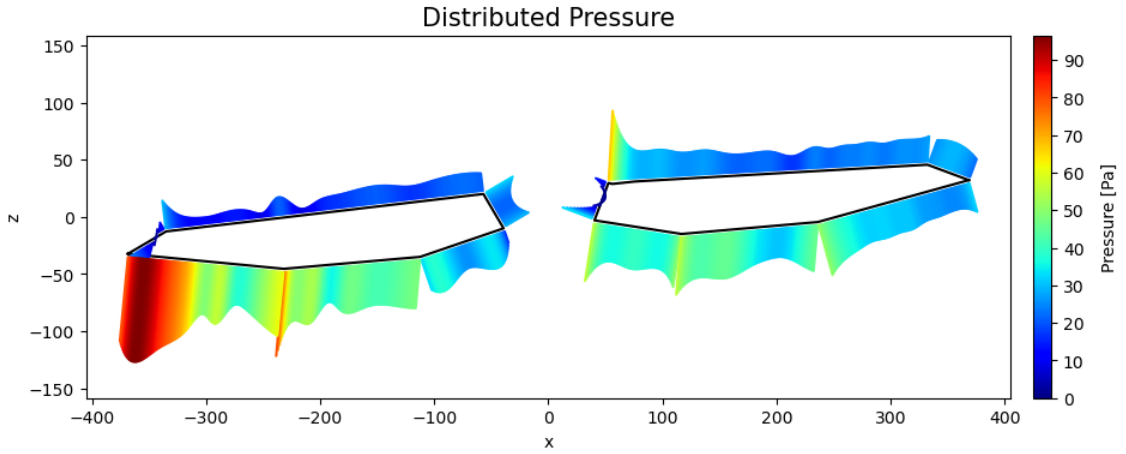


Figure 6.22: Distributed pressure with grid rotation 7 Hz, RPM = 330, $V \approx 9m/s$ and $\alpha = -5°$.

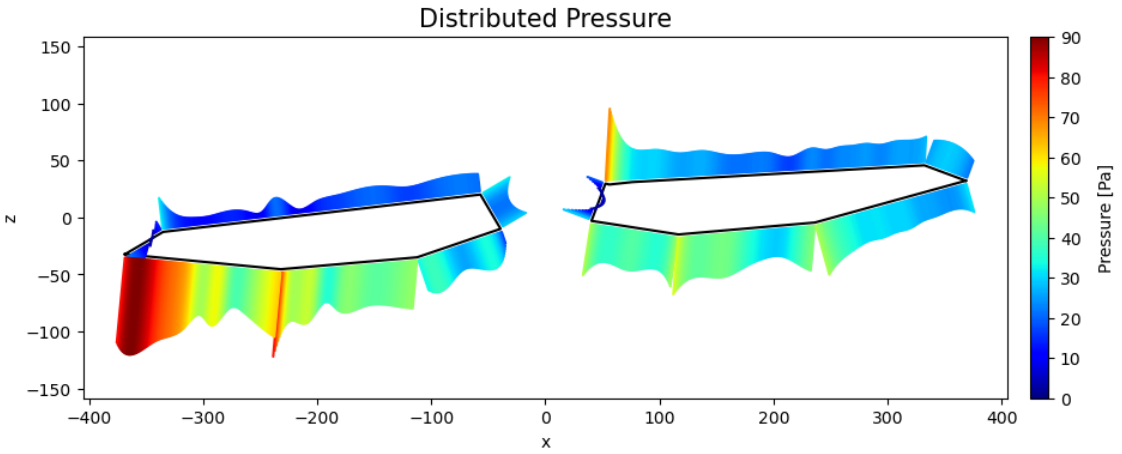
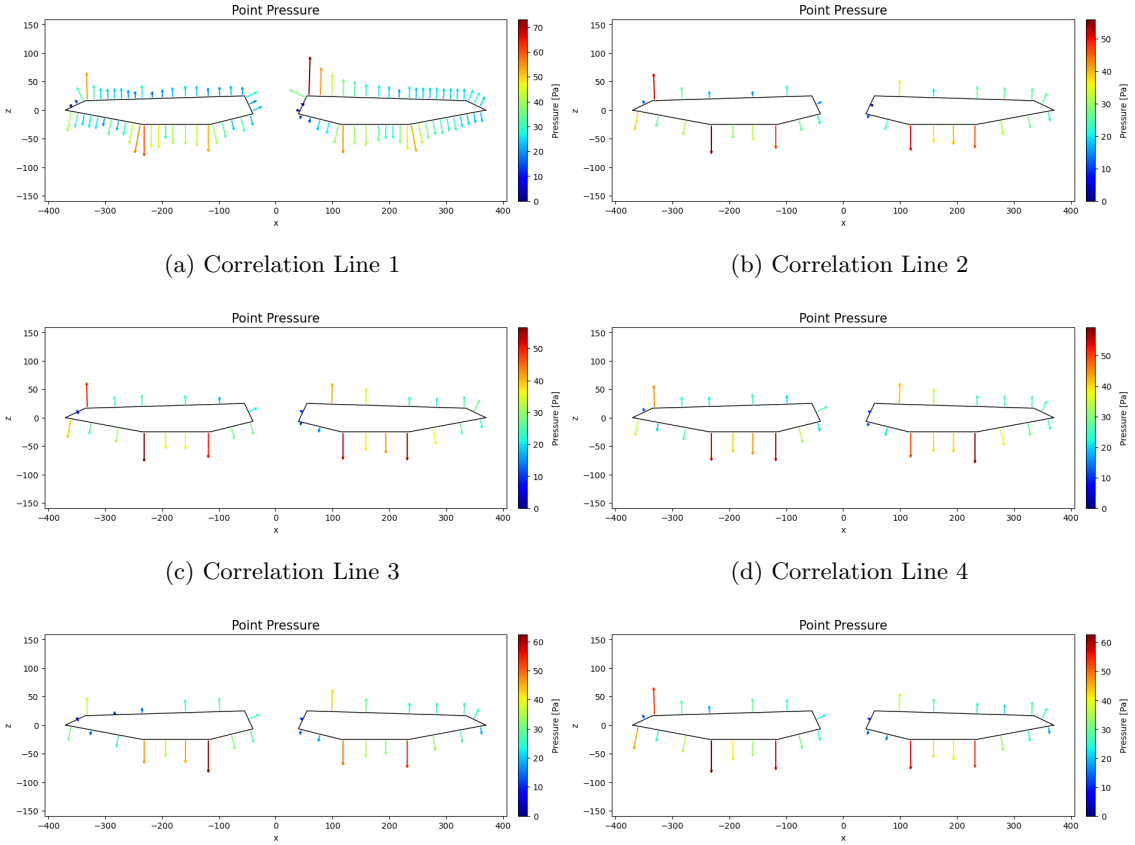


Figure 6.23: Distributed pressure with grid rotation 0.5 Hz, RPM = 330, $V \approx 9m/s$ and α = - 5°.

The pressure distribution displays the same trend as for a still open grid, but with a much larger magnitude of the pressure. The negative pressure is observed on all surfaces on the downstream box, except for small values of positive pressure on S9. The negative pressure becomes more dominant at bottom surfaces for both the upstream-box and the downstream-box as the angle of attack decreases.

Pressure distribution for the six correlation lines

The pressure distribution for correlation line 1-6 with wind velocity, $V \approx 9m/s$, angle of attack, $\alpha = 0°$ and grid rotation of 7 Hz, is presented below.



(a) Correlation Line 1

(b) Correlation Line 2

(c) Correlation Line 3

(d) Correlation Line 4

(e) Correlation Line 5

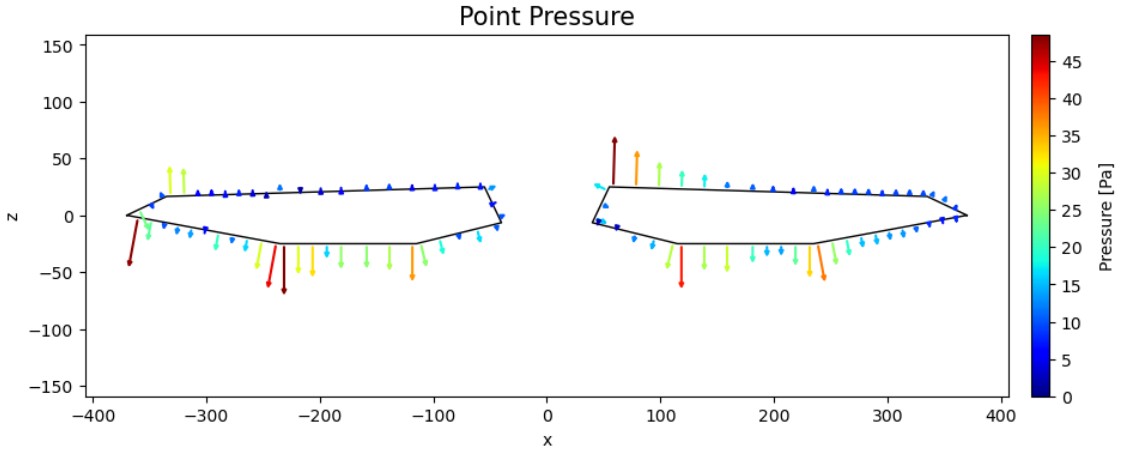
(f) Correlation Line 6

Figure 6.24: Point pressure with grid rotation 7 Hz for the six correlation lines, RPM = 330, $V \approx 9m/s$ and $\alpha = 0°$.

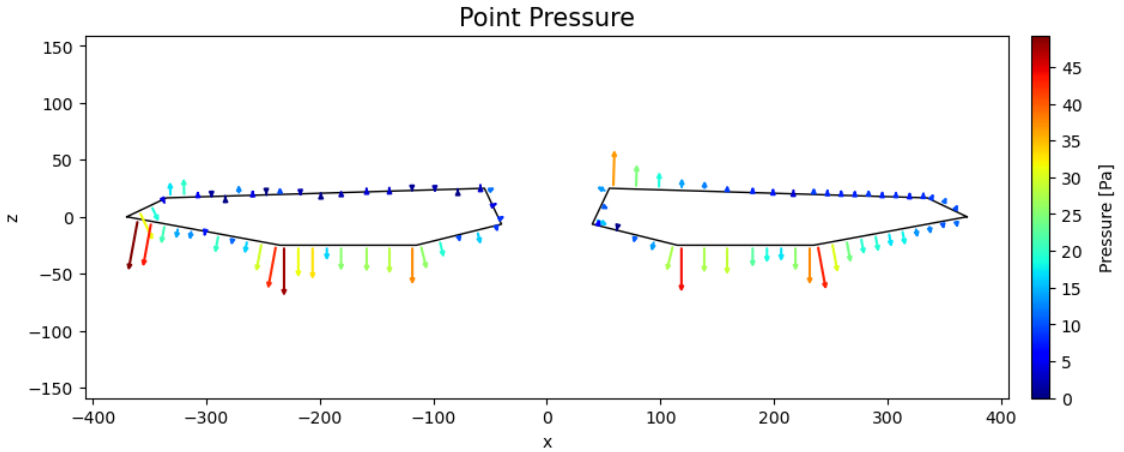
The pressure distribution for the different correlation lines in Figure 6.24, displays similarity to the measurements with a still open grid, shown in Figure 6.17. The pressure on S5 is slightly different on correlation line 5 than for the other correlation lines. The same observation was done for the test executed with no grid rotation. Since this is observed on several tests, there may be something that has disturbed the wind flow near correlation line 5. The Cobra Probes were placed 40 cm from the windward edge, in front of the first and fifth correlation line. This could affect the measurements of the fluctuating pressure around the cross-section as the Cobra Probes may disturb the wind flow and cause a slight change in the pressure distribution.

6.2.3 The Effect of Railings

In order to detect the impact of the railings, the pressure distribution is studied. The pressure distribution for the tests executed with and without railings is compared, and the main differences are presented below. The tests are done with a still open grid, a mean wind velocity of $9m/s$, and angle of attacks 0°, 2° and 5°. Figure 6.25-6.27 shows the point pressure without and with railings.

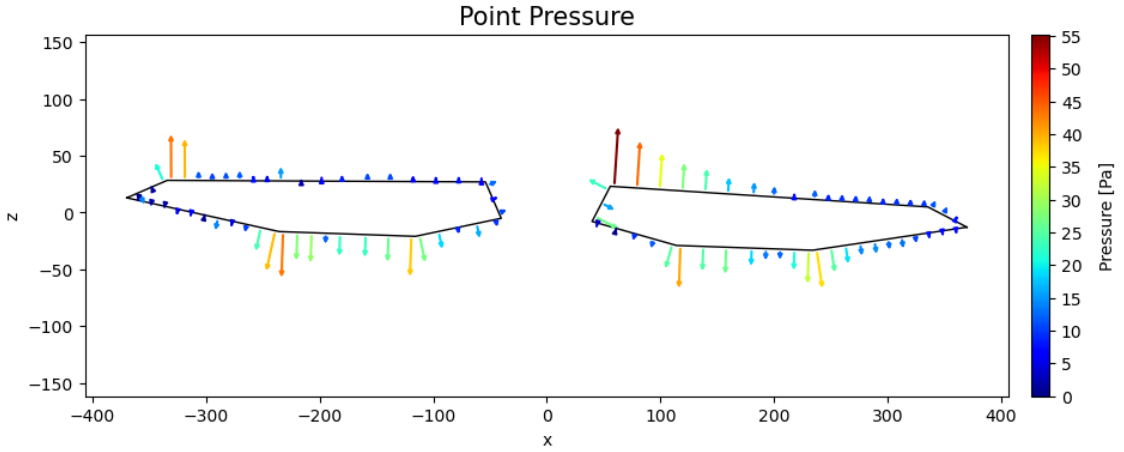


(a) Without railings

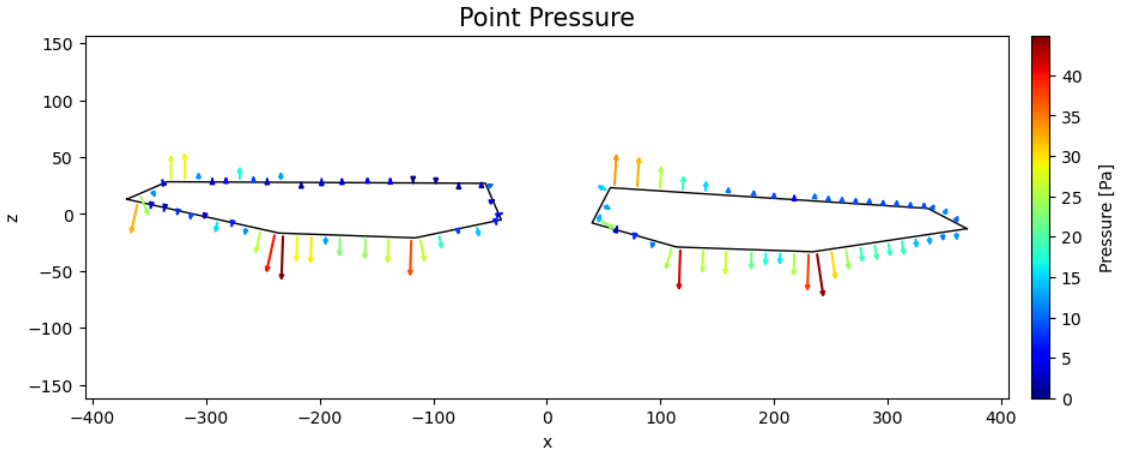


(b) With railings

Figure 6.25: Distributed pressure with still open grid, RPM = 330, $V \approx 9m/s$ and $\alpha = 0°$.

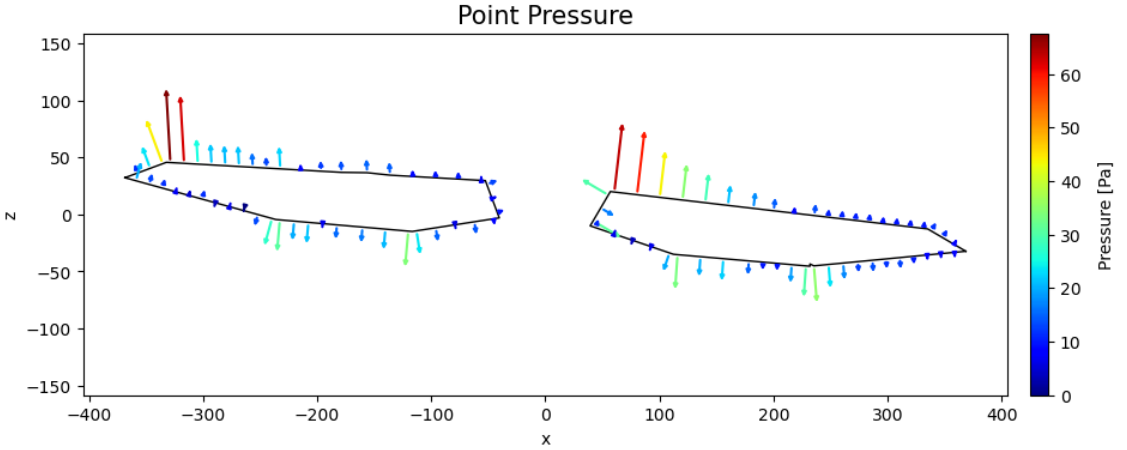


(a) Without railings



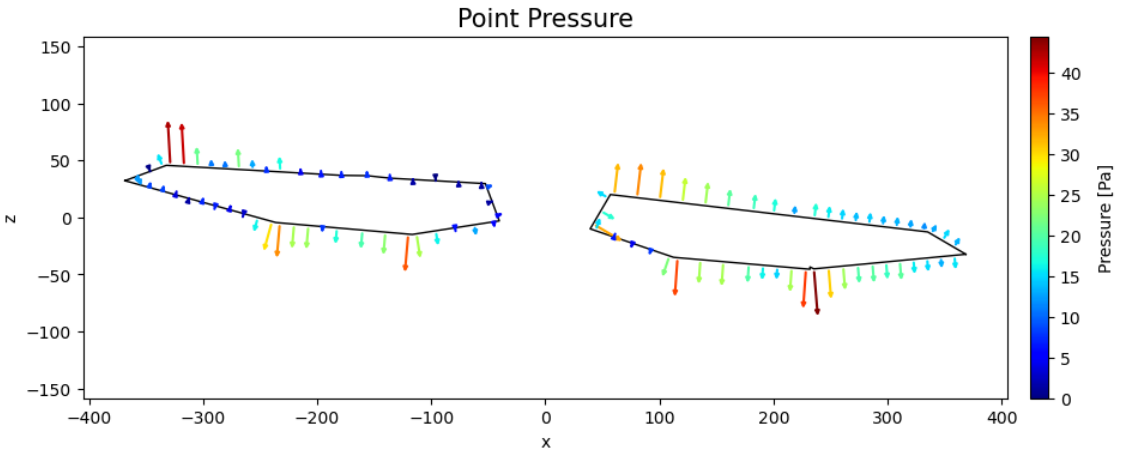
(b) With railings

Figure 6.26: Distributed pressure with still open grid, RPM = 330, $V \approx 9m/s$ and $\alpha = 2°$.



(a) Without railings

Figure 6.27: Distributed pressure with still open grid, RPM = 330, $V \approx 9m/s$ and $\alpha = 5°$.



(b) With railings

Figure 6.27: Distributed pressure with still open grid, RPM = 330, $V \approx 9m/s$ and $\alpha = 5°$.

When comparing the pressure distribution from tests done with and without railings, the pressure distributions display some similarity except in the vicinity of the railings. It can be observed from Figure 6.25 that the high negative pressure on S2 on the upstream-box and S8 on the downstream-box decreases when the railings are attached to the girder. In addition, on the lower surface S6, the negative pressure shows a slight increase due to the railings' blockage effect. The high fluctuating pressure on the downstream-box can result from the vortex shed from the upstream-box. When the railings are attached on the leading edge, they can induce large flow separations that cause an increase in the distance between the upper and lower shear layers of the gap [47]. The interaction decreases due to the increased distance between the layers, and less motion-induced vortex shedding is formed in the gap. This implies that the railings will affect the VIV of the twin-box bridge. The railings reduces the negative pressure on the leading edge of S2 and produce a wave-like pressure distribution where the railings are attached.

The negative pressure were increased on the upper surfaces when the angle of attack increases. The railings showed to have a significant impact on the pressure distribution on the leading edge of both the upstream-box and the downstream-box when the angle of attack is 5°. It can be seen in Figure 6.27 that the negative pressure is almost reduced by half.

6.2.4 Comparison and Validation of Pressure Distribution

The pressure distributions presented in this thesis display an unexpected amount of negative pressure on the upstream-box at S6. This does not coincide with results obtained from previous studies and gives it a reason for questioning the fluctuating pressure on the upstream-box.

When comparing the pressure distribution with the result obtained in the experimental study by Wang et al. [3], it is a noticeable difference in the pressure on the bottom surface on the windward edge of the upstream-box. According to the study, the wind flow gives a positive pressure on the windward edge and slowly turns negative as it gets closer to the bottom surface. This also agrees with pressure distribution found in other studies and master's theses. A possible reason for this unexpected negative pressure may be that something is triggering the wind flow around this area. The correlation lines show approximately the same pressure distribution for all lines, as seen in Figure 6.17 and 6.24. This indicates that there are no local errors for correlation line one, since all the lines presents negative pressure on S6.

In addition, an underpressure in the wind tunnel could cause negative pressure on the cross-section. In that case, a tentative underpressure could be subtracted from the mean value of the pressure from each pressure hole. The measured negative pressure is relatively large, and therefore desires a large underpressure to obtain the same results as Wang et al. This would completely change the distributed pressure on the cross-section, which would not correspond with previous findings.

Another contribution to the unexpected negative pressure could be the sharp angle on the windward edge. The angle of S6 is relatively smaller compared to bridge cross-sections from other studies. Rocchi et al. [42] did an experimental study where he studied the pressure distribution with different angles of attack and global forces on a bridge deck. An illustration based on the pressure distribution found in the study for angle of attack of 0° and -9° is presented below.

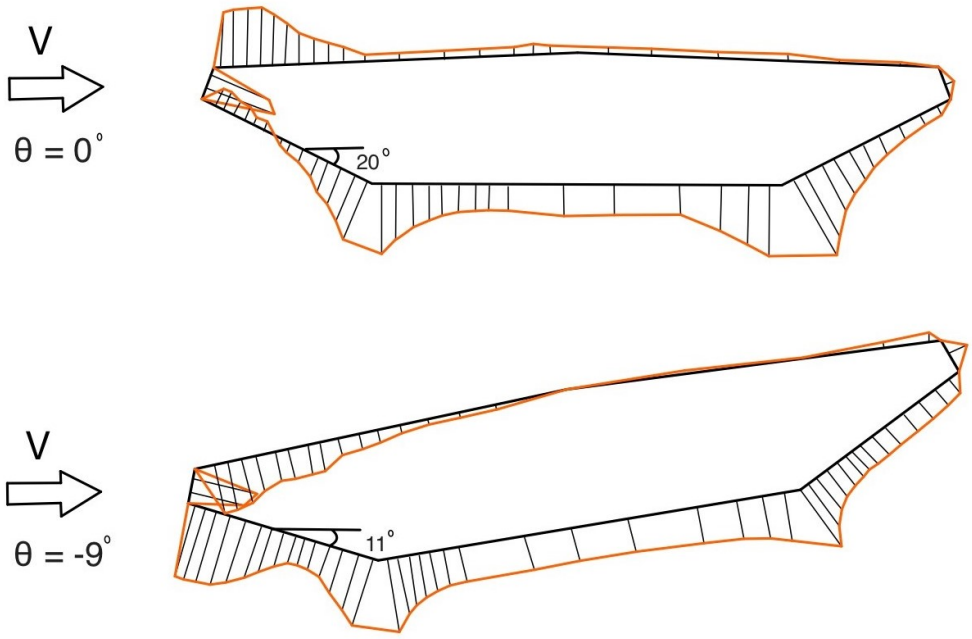


Figure 6.28: Illustration of pressure distribution with angle of attack 0° and -9° based on the study by Rocchi et al. [42].

An angle of attack of 0° shows an pressure distribution with positive pressure on the lower surface on the leading edge. The angle of this surface is 20°, which is larger than the angle for the cross-section used in this thesis. When the angle of attack is -9°, there is a complete separation, and the positive pressure turns to negative pressure. The surface angle on the leading edge becomes 11° as shown in Figure 6.28, which is approximately the same as the angle for surface S6 as shown in Figure 4.1. In that case, great similarity is shown when comparing the pressure distribution on the lower leading surface of the cross-section in this thesis and the cross-section in the study by Rocchi et al. This observation may confirm that the pressure on S6 at the leading edge may indeed become negative. If the angle of the leading edge is small enough, the expected positive pressure on the leading edge may become negative with an angle of attack of 0°.

6.3 Comparison of Forces

The forces measured by the pressure scanner are calculated using two different methods, the interpolated method and the piece-wise method, as described in Subsection 5.4.1. Table 6.4 presents the total static forces on the upstream-box and the downstream-box for wind velocity $V \approx 7m/s$ and $V \approx 9m/s$ with an angle of attack of 0°. In addition, the forces from the load cells are also presented in the table for comparison.

V [m/s]	**Interpolated load method**		**Piece-wise load method**		**Load Cells**	
	Upstream	Downstream	Upstream	Downstream	Upstream	Downstream
	Fx[N]					
7	0.24	0.88	0.29	0.86	1.16	1.58
9	0.48	1.61	0.58	1.61	1.83	2.47
	Fz[N]					
7	-9.04	-1.98	-8.81	-1.99	-7.84	-1.33
9	-14.82	-3.31	-14.35	-3.38	-12.09	-2.18
	My[Nm]					
7	-1.99	0.60	-1.89	0.65	-1.52	0.52
9	-3.31	0.97	-3.10	1.03	-2.35	0.84

Table 6.4: Forces obtained from the interpolated load method, the piece-wise method and the load cells.

The forces obtained by the pressure scanners are lower compared to the forces obtained from the load cells in the horizontal direction. The load cells include the friction forces in the measurements, which the pressure scanners are not able to measure. This results in lower forces and inaccurate measurements in the horizontal direction. When the wind velocity increases, the deviation between the forces measured by the pressure scanner and the load cells becomes even more remarkable. Further, the forces obtained by the pressure measurements in the vertical direction have a higher magnitude than the forces measured by the load cells for both wind velocities. This also applies to the moment forces. When comparing the forces, larger errors are observed on the upstream-box than on the downstream-box.

A slight deviation is observed when comparing the interpolated method and the piece-wise load method. The forces obtained by the piece-wise load method are slightly closer to the load cell forces than the forces obtained by the interpolated load method. In general, the interpolated load method presents forces with a higher magnitude, except for the forces in the vertical direction. Even though both methods show some differences compared with forces measured by the load cells, they can be used for further calculations.

6.4 Static Coefficients

The static coefficients can be estimated by using both forces measured by the load cells and forces measured by the pressure scanners. When the static coefficients are estimated by the load cells, the static tests described in Section 5.2.2 are used. The section model is rotated with an amplitude of ± 10 ° with wind velocities of approximately 6, 7, 9, and 10 m/s. The tests were executed with a still open grid, grid rotation of 7 Hz and 0.5 Hz. The measured forces are filtered with a Butterworth filter of order six and a cutoff frequency equal to 6 Hz. Equation 3.24 is used to calculate the static coefficients, and the whole width of the twin-box bridge, B, is used in the

calculations. The drag coefficients $C_D(\alpha)$, lift coefficients $C_L(\alpha)$, and moment coefficients $C_M(\alpha)$ are plotted as a function of α, which is the incline angle of the bridge decks.

Static Tests

The static coefficients for the total twin-box, the upstream-box and the downstream-box obtained from the static test with forces measured by the load cells, are presented in Figure 6.29, 6.30 and 6.31. The unfiltered raw data, which is the noise from the vibration response, is plotted together with the filtered forces. Since the bridge was rotated more than once, a third-degree polynomial function is fitted to the filtered data. This is used to extract the values for the static coefficients at different angles of attack.

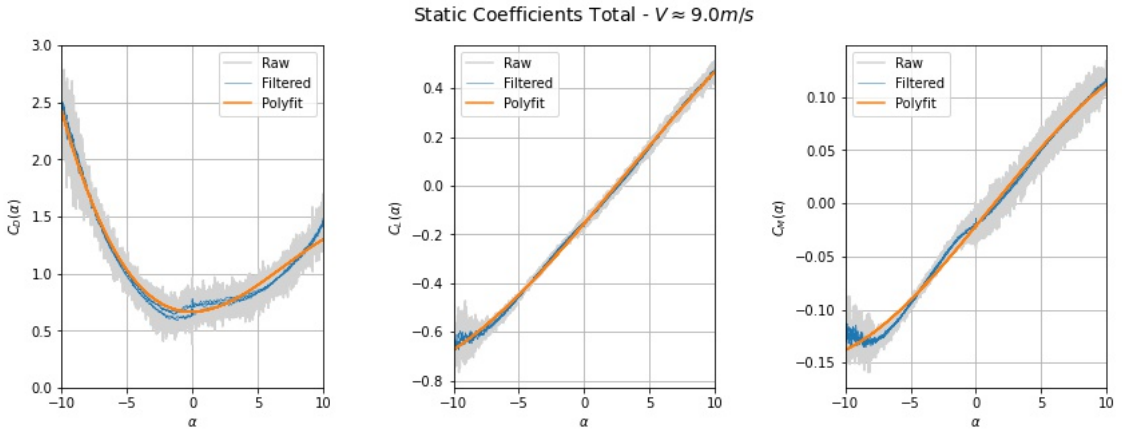


Figure 6.29: Total static coefficients for the bridge with still open grid and $V \approx 9m/s$.

Total Static Coefficients			
$\boldsymbol{\alpha[deg]}$	$\boldsymbol{C_D}$	$\boldsymbol{C_L}$	$\boldsymbol{C_M}$
-10	2.425	-0.6731	-0.138
-8	1.717	-0.5943	-0.1231
-6	1.214	-0.5005	-0.1028
-4	0.89	-0.3943	-0.0783
-2	0.717	-0.2785	-0.0508
0	0.667	-0.1556	-0.0214
2	0.713	-0.0288	-0.0087
4	0.826	0.0997	0.0382
6	0.981	0.2271	0.0661
8	1.148	0.3506	0.0912
10	1.300	0.4675	0.1122

Table 6.5: Static coefficients at different angles of attack.

The drag coefficient, $C_D(\alpha)$ in Figure 6.29 shows clear non-linearity and appear as a parabolic curve. As for the lift coefficient, $C_L(\alpha)$, and the moment coefficient $C_M(\alpha)$, a more typical linear

relationship is displayed. An exception of this linear relationship appears between the negative inclination $-5°$ and $-10°$ where some non-linearity is observed. This can be caused by the induced wind flow on top surfaces, creating a more turbulent flow around the cross-section for large negative angles. As a result of this, the bridge section may start to vibrate. Negative values of C_L are observed for approximately $\alpha < 3°$, which indicates that the vertical forces on the bridge section are acting in the downward direction. When the angle of attack decreases towards $-10°$, the negative pressure on the bottom surfaces gets a higher magnitude than the negative pressure on the top surface. This is due to flow accelerating over the streamlined bottom surfaces. Moreover, for $\alpha = 0°$, the desired value of C_M is close to zero to avoid rotation of the bridge cross-section when exposed to wind. The static coefficient C_M for this bridge section when the angle of attack is zero is approximately -0.021. This value is fairly close to zero and therefore indicates sufficient rational stability.

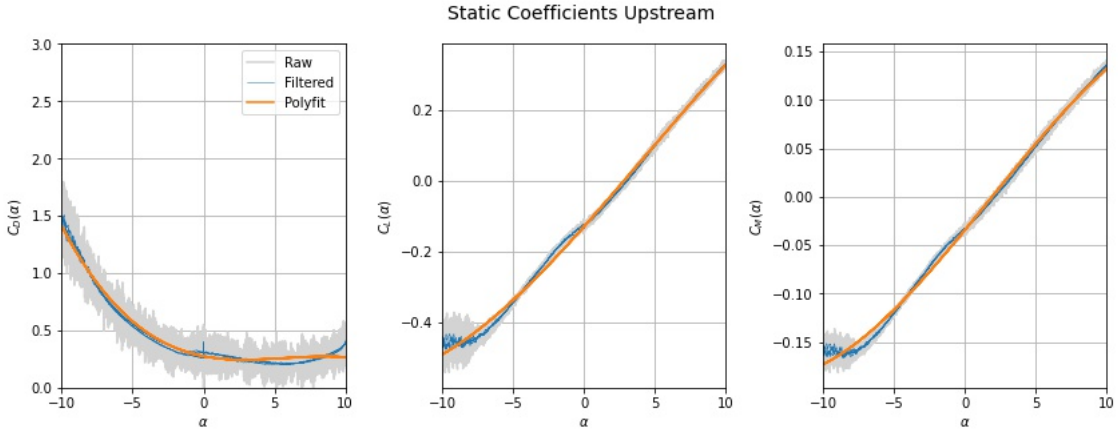


Figure 6.30: Static coefficients for the upstream-box, $V \approx 9m/s$, still open grid

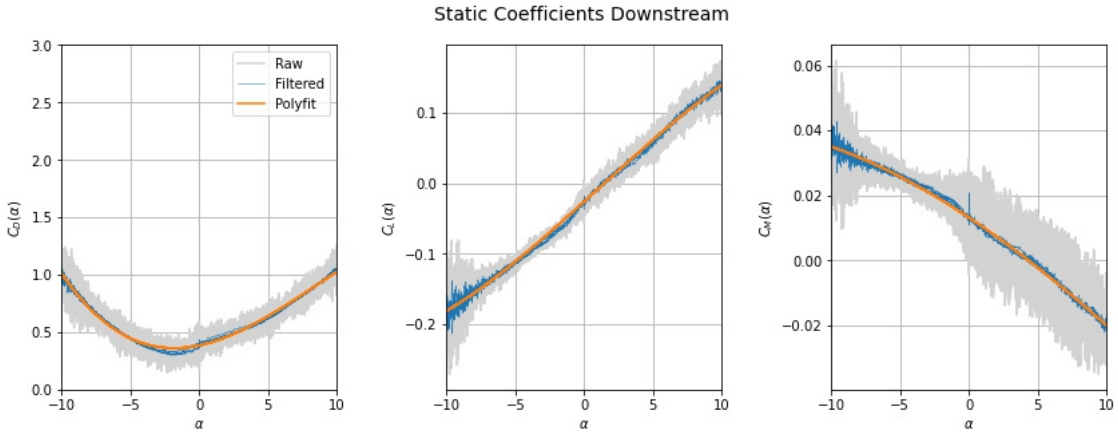


Figure 6.31: Static coefficients for the downstream-box, $V \approx 9m/s$, still open grid

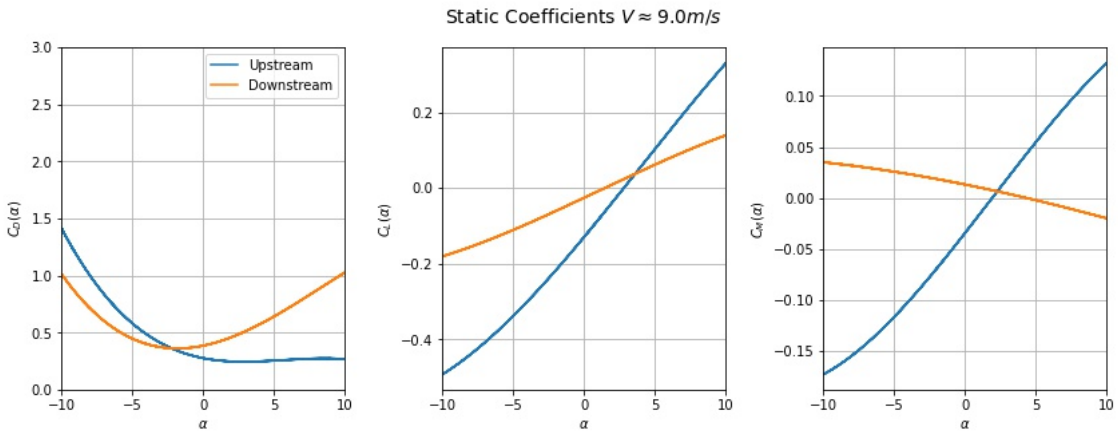


Figure 6.32: Static coefficients for upstream-box and downstream-box, $V \approx 9m/s$, still open grid

$\alpha[deg]$	Upstream-box			Downstream-box		
	C_D	C_L	C_M	C_D	C_L	C_M
-10	1.419	-0.4923	-0.1731	1.0128	-0.1808	-0.0351
-8	1.001	-0.4388	-0.1548	0.7152	-0.1555	0.0318
-6	0.6985	-0.3739	-0.1306	0.5159	-0.1265	0.0279
-4	0.4876	-0.2995	-0.1017	0.4029	-0.0948	0.0234
-2	0.353	-0.2174	-0.0693	0.3643	-0.0611	0.0185
0	0.2791	-01296	-0.0345	0.3882	-0.0261	0.0131
2	0.2501	-0.038	0.0014	0.4627	0.0092	0.0073
4	0.2505	0.0555	0.0371	0.576	0.0442	0.001
6	0.2645	0.1491	0.0717	0.7161	0.078	-0.0056
8	0.2764	0.2408	0.1038	0.8711	0.1098	-0.0126
10	0.2707	0.3287	0.1321	1.0292	0.1388	-0.0199

Table 6.6: Static coefficients at different α for the upstream-box and downstream-box.

Figure 6.32 presents the static coefficients for the upstream-box and the downstream-box. It can be observed from the plot of both C_L and C_M, that the main contribution derive from the upstream-box for all angles. For C_D the main contribution derive from the upstream-box for $-10° < \alpha < -2°$ and from the downstream box for $\alpha > -2°$. For an angle of attack, $\alpha = 0°$, it is expected that the main contribution is provided by the upstream-box. The results presented in Figure 6.32 and Table 6.6, contradict this. It is observed that the downstream-box is the main contribution to the drag coefficient for $\alpha = 0°$. From the forces presented in Section 6.3, it can be seen that forces in the horizontal direction obtained by the pressure scanner and the load cells are the largest for the downstream-box, which results in the downstream-box becoming the main contribution for the drag coefficient. The geometry of the cross-section has a great impact of how the wind flow interacts with the twin-box, and how the fluctuating pressure is distributed around the cross-section. The chosen cross-section, with a sharp angle in the leading edge of the upstream-box, may be the reason for this unexpected observation.

In order to investigate if the static coefficients are dependent on the wind velocity or turbulence flow, the response of the bridge section is plotted in Figure 6.33 with four different wind velocities and in Figure 6.34 with three different grid generations.

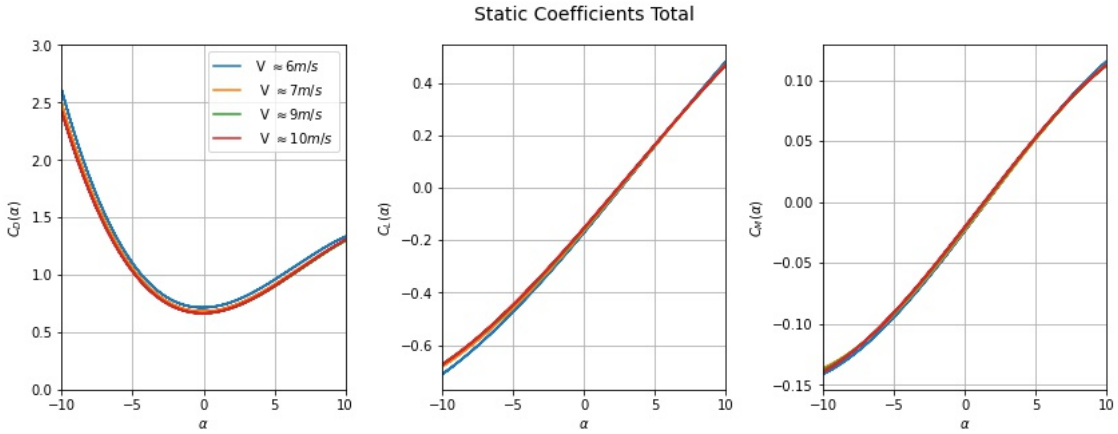


Figure 6.33: Static coefficients for different wind velocities, Still open grid

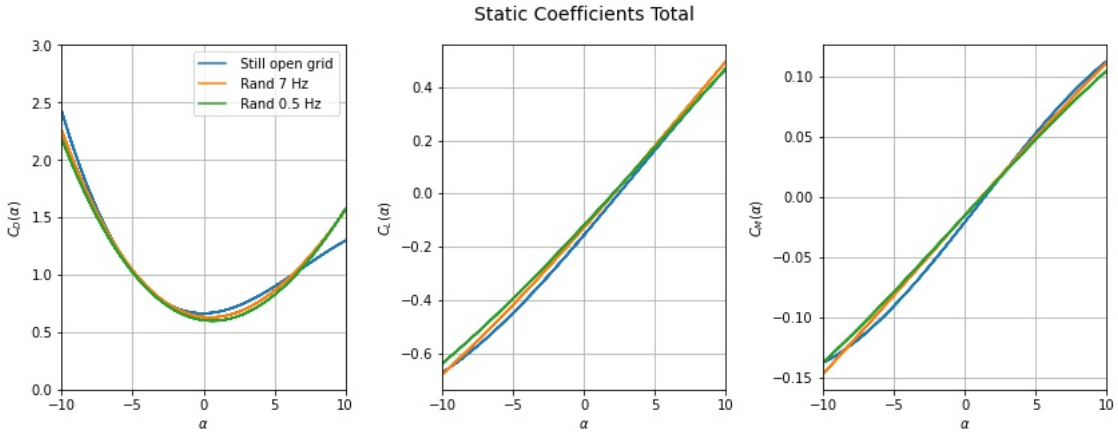


Figure 6.34: Static coefficients with different grid rotations, $V \approx 9m/s$.

The different curves in the plot present the response of the bridge section with different wind velocities and different grid generations. The curves show significant similarity when the wind velocity changes. The same is observed for the curves with different grid generations. The lack of offset between the curves when the wind flow changes indicates that the twin-box bridge has very low Reynolds dependency.

The drag, lift and moment slopes, $dC_D/d\alpha$, $dC_L/d\alpha$ and $dC_M/d\alpha$ are presented in Figure 6.35 for the upstream-box, the downstream-box and the total twin-box. They are defined as the rate of change of the static coefficients with respect to the angle of attack. When calculating the slopes, the angle is measured in radians.

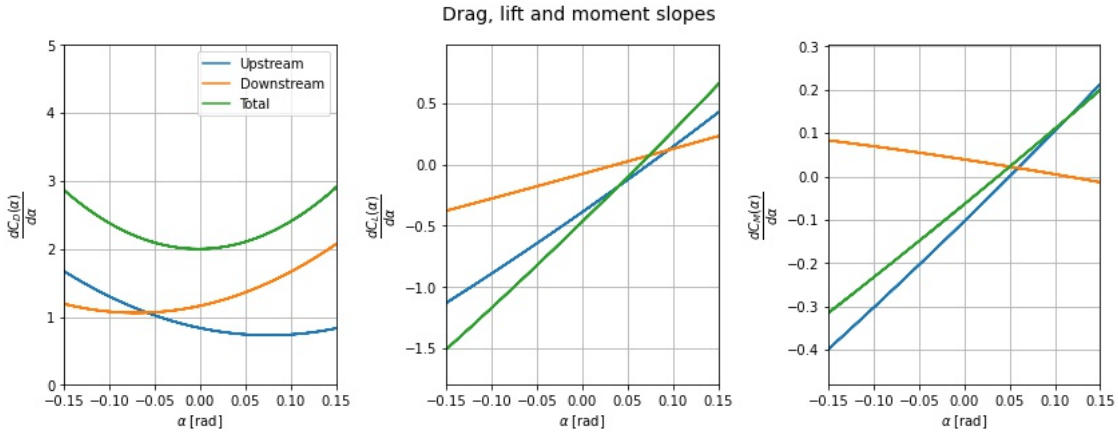


Figure 6.35: Drag, lift and moment slopes for the upstream-box, the downstream-box and the total twin-box.

Total			
$\boldsymbol{\alpha[deg]}$	$\boldsymbol{C'_D}$	$\boldsymbol{C'_L}$	$\boldsymbol{C'_M}$
-10	3.173	-1.660	-0.3547
-8	2.748	-1.436	-0.2980
-6	2.418	-1.20	-0.2406
-4	2.183	-0.960	-0.1825
-2	2.048	-0.7158	-0.1237
0	2.002	-0.4673	-0.0642
2	2.054	-0.2146	-0.004
4	2.202	0.0423	0.0568
6	2.446	0.3035	0.1183
8	2.786	0.5690	0.1806
10	3.221	0.8387	0.2434

Table 6.7: Derivative of static coefficients for the twin-box.

$\alpha[deg]$	Upstream-box			Downstream-box		
	C'_D	C'_L	C'_L	C'_D	C'_L	C'_L
-10	1.8876	-1.2412	-0.4448	1.2853	-0.4261	0.090
-8	1.5876	-1.0784	-0.3788	1.1599	-0.3574	0.0808
-6	1.3325	-0.9118	-0.3116	1.0851	-0.2883	0.0711
-4	1.1225	-0.7413	-0.2434	1.0161	-0.2187	0.609
-2	0.9574	-0.567	-0.174	1.0874	-0.1488	0.0503
0	0.8372	-0.3889	-0.1035	1.1645	-0.0784	0.0393
2	0.7621	-0.207	-0.03187	1.2922	-0.0077	0.0279
4	0.7319	-0.0212	0.0409	1.4706	0.0635	0.0159
6	0.7467	0.1684	0.1147	1.6995	0.1351	0.0036
8	0.8065	0.3618	0.1897	1.9791	0.2071	-0.0092
10	0.9112	0.5591	0.2658	2.309	0.2796	-0.0224

Table 6.8: Derivative of static coefficients for the upstream-box and the downstream-box.

Pressure Measurements

The static coefficients based on the measurements obtained from the pressure scanners are calculated for five different angles of attack. Table 6.9 presents the static coefficients obtained from the pressure measurements together with the static coefficients obtained from the load cells for comparison. The static coefficients for the upstream-box and the downstream-box is presented in Table 6.10 and 6.11.

$\alpha[deg]$	C_D		C_L		C_M	
	MPS	**Load Cells**	**MPS**	**Load Cells**	**MPS**	**Load Cells**
-5	0.6168	1.032	-0.5036	-0.4487	-0.1086	-0.0910
-2	0.3841	0.717	-0.309	-0.2785	-0.0574	-0.0508
0	0.3444	0.667	-0.188	-0.1558	-0.030	-0.0214
2	0.4166	0.713	-0.0651	-0.0288	-0.0045	0.0087
5	0.6223	0.900	0.1525	0.1637	0.0405	0.0525

Table 6.9: Static coefficients obtained from the pressure scanners and the load cells.

Upstream-box						
$\alpha[deg]$	C_D		C_L		C_M	
	MPS	**Load Cells**	**MPS**	**Load Cells**	**MPS**	**Load Cells**
-5	0.3177	0.5825	-0.3773	-0.3378	-0.1358	-0.1167
-2	0.1373	0.353	-0.2357	-0.2174	-0.0783	-0.0693
0	0.091	0.2791	-0.1521	-0.1296	-0.0444	-0.0345
2	0.0895	0.2501	-0.0697	-0.038	-0.0113	0.0014
5	0.1479	0.2568	0.0857	0.1024	0.0447	0.0547

Table 6.10: Static coefficients for the upstream-box.

Downstream-box						
$\alpha[deg]$	C_D		C_L		C_M	
	MPS	**Load Cells**	**MPS**	**Load Cells**	**MPS**	**Load Cells**
-5	0.2991	0.4493	-0.1263	-0.111	0.0273	0.0257
-2	0.2468	0.3643	-0.0733	-0.0611	0.0209	0.0185
0	0.2533	0.3882	-0.0358	-0.0261	0.0147	0.0131
2	0.327	0.4627	0.0045	0.0092	0.0068	0.0073
5	0.4759	0.6434	0.0668	0.0613	-0.0042	-0.0022

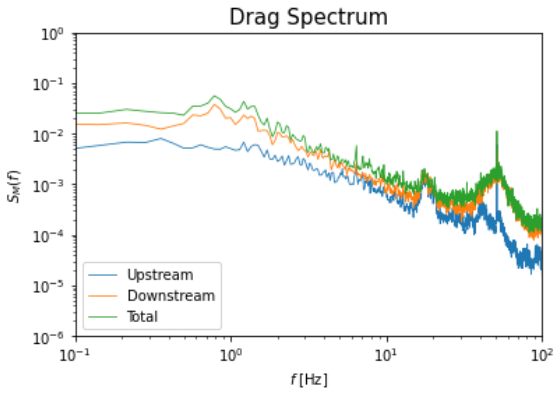
Table 6.11: Static coefficients for the downstream-box.

It can be observed that the drag coefficients, C_D obtained by the pressure scanner are generally lower than the drag coefficient from the load cells. As explained in Section 6.3, the pressure scanners are not able to measure the friction forces. The drag coefficients obtained by the measurements from pressure scanners are therefore lower. Hence, the drag coefficient measured by the load cells is more accurate. For the lift and moment coefficient C_L and C_M, the deviation is lower, and they display more similarity than the drag coefficients. It can be observed that for C_L and C_M, the static coefficients for the upstream-box is larger than for the downstream-box. This implies that the largest contribution to the static coefficients C_L and C_M provided by the upstream-box. For the drag coefficient C_D, the largest contribution comes from the downstream-box. In addition, the deviation is smaller when the magnitude of the static coefficients is lower.

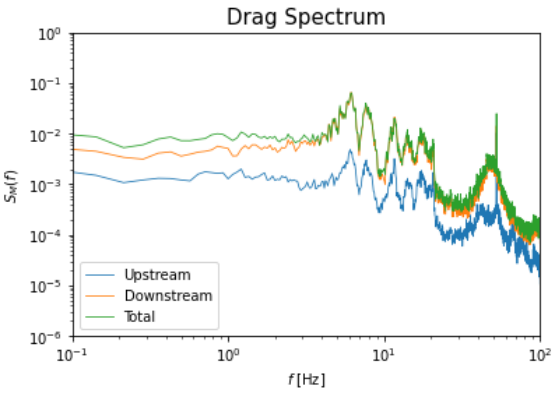
6.5 Force Spectra

The buffeting forces are calculated based on the measurements obtained from the pressure scanners. Forces from the piece-wise load method are used when calculating the buffeting forces and force spectra. The piece-wise load method and the interpolated load method would obtain the same results, but the piece-wise load method performs the calculations faster. Therefore, the piece-wise load method is favourable.

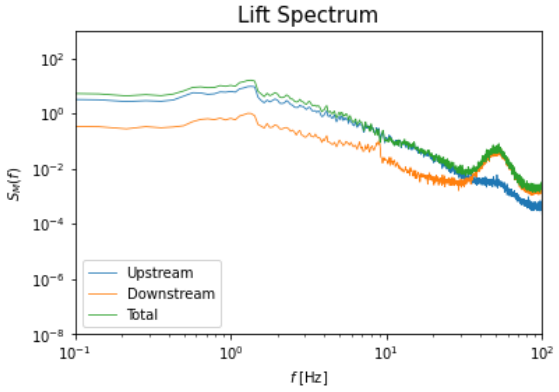
The buffeting force spectra for turbulent flow with grid rotations of 0.5 Hz and 7 Hz is presented in Figure 6.36.



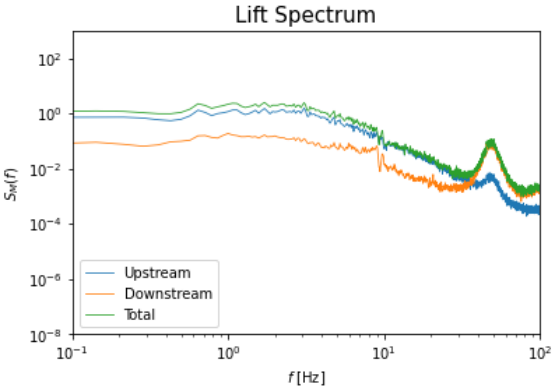
(a) Drag Spectrum, grid rotation of 0.5 Hz



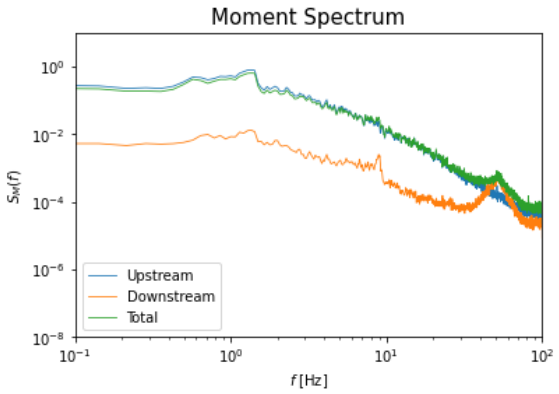
(b) Drag Spectrum, grid rotation of 7 Hz



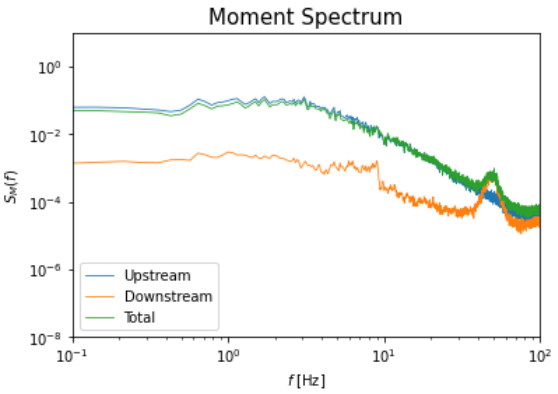
(c) Lift Spectrum, grid rotation of 0.5 Hz



(d) Lift Spectrum, grid rotation of 7 Hz



(e) Moment Spectrum, grid rotation of 0.5 Hz



(f) Moment Spectrum, grid rotation of 7 Hz

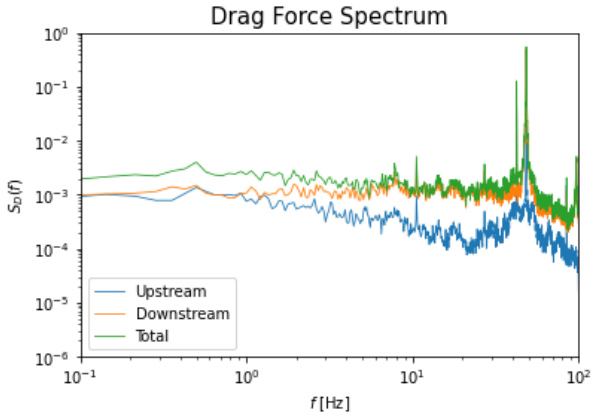
Figure 6.36: Buffeting Force Spectra, grid rotation of 0.5 Hz (left) and 7 Hz (right), $V \approx 9m/s$.

For the drag force spectra with grid-rotation of 0.5 Hz, two distinct peaks at the frequencies 18 Hz and 50 Hz are observed. For the lift and moment force spectra only one distinct peak at frequency 50 Hz is observed. The thin peaks shown on the drag spectra at the frequency of 50 Hz may be caused by electrical noise, which often occurs at this frequency. For grid-rotation of 7 Hz, the force spectra for drag, lift, and moment also has a distinct peak at 50 Hz. The buffeting force spectra for the two different turbulent flows displays similarity, except for some deviations for the drag force spectra as seen in Figure 6.36b. In the interval 5-20 Hz, several peaks occur. They are mainly caused by the downstream-box but are also observed on the upstream-box. Vibrations of the cross-section could cause the peaks. However, it is difficult to determine the exact reason

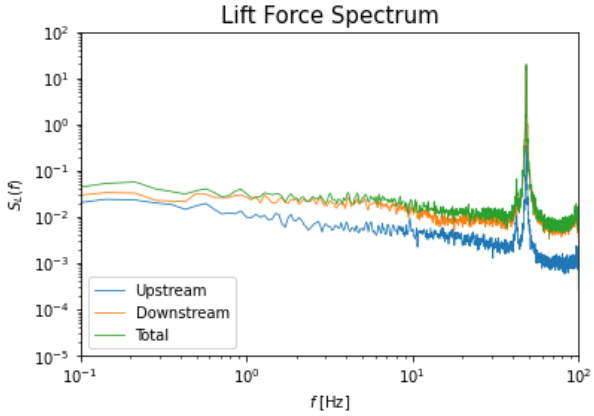
why several peaks occur in this interval for grid-rotation of 7 Hz and not 0.5 Hz. In order to verify the plots, the buffeting force spectra calculated with the measurements from the load cells were checked. The results obtained from the load cells also display distinct peaks for the same frequencies as those obtained from the pressure scanners. This confirms that there is no error in the calculations of the buffeting force spectra.

Similarity is shown when studying the buffeting force spectra and comparing it to previous studies. In the experimental study by Wang et al. [23], distinct peaks were observed for the lift and moment force spectra around a frequency of 25 Hz. The downstream-box is the main contribution to the peaks, which most likely is caused by the vortex shedding effect. The peaks in the force spectra obtained in this thesis occur at a much higher frequency, 50 Hz, which most likely is caused by vortex shedding from the trailing edge of the upstream-box. The main contribution of the drag force spectra in turbulent flow is from the downstream-box for both grid-generations. For the lift and moment force spectra, the main contribution is provided by the upstream-box. The vortex shedding from the upstream-box will affect the downstream-box and, according to this, change the lift on the downstream-box.

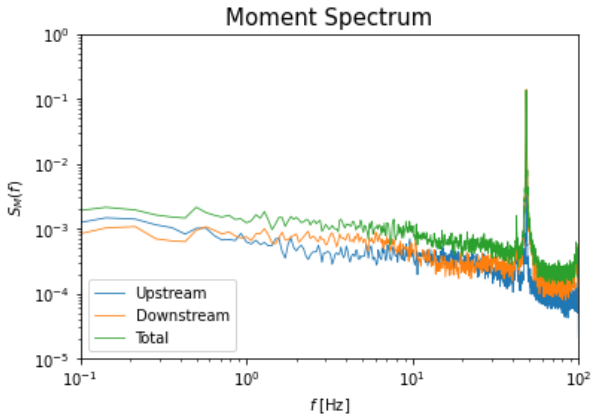
Further, the buffeting force spectra for a still open grid are presented in Figure 6.37. It can be observed from the figure that for all three force spectra, the main contribution is provided by the downstream box. This indicates that the grid-generated turbulence increases the lift and moment forces on the upstream-box, when compared to the downstream-box.



(a) Drag Force Spectra



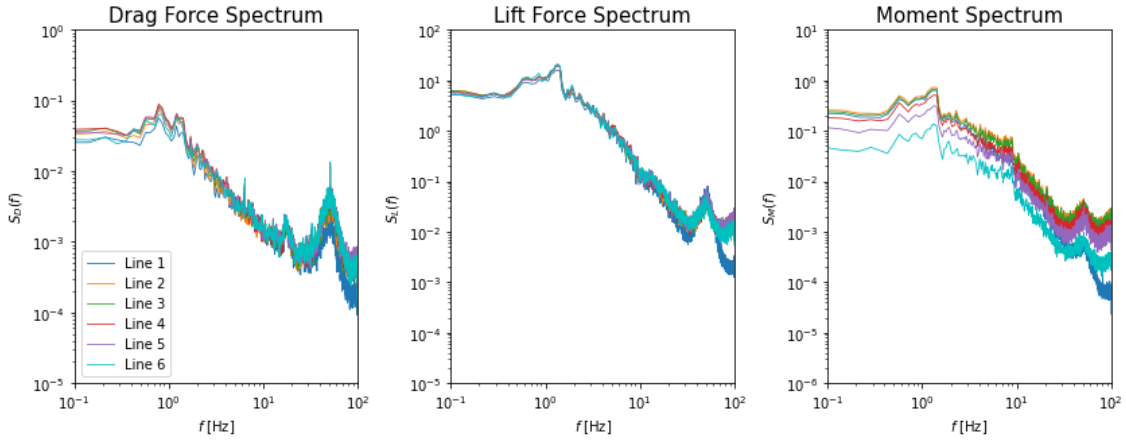
(b) Lift Force Spectra



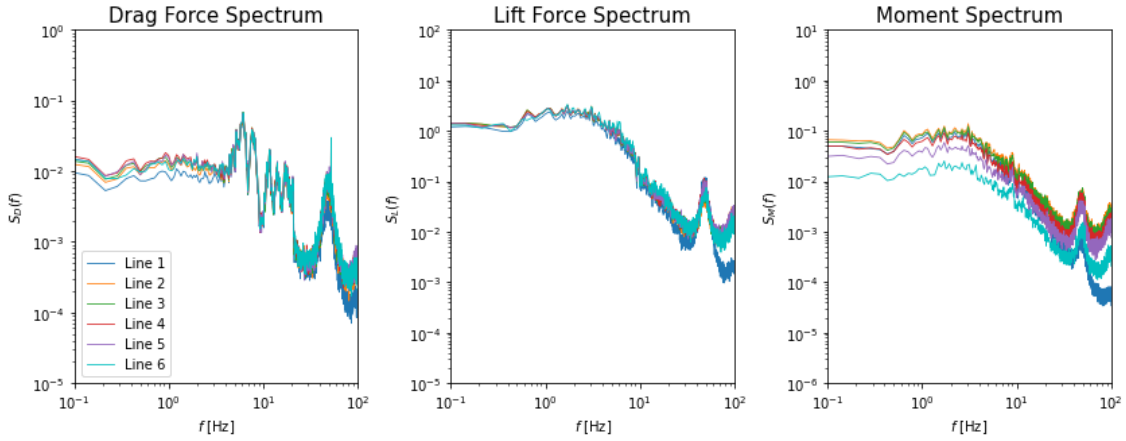
(c) Moment Force Spectra

Figure 6.37: Buffeting Force Spectra, still open grid, $V \approx 9m/s$

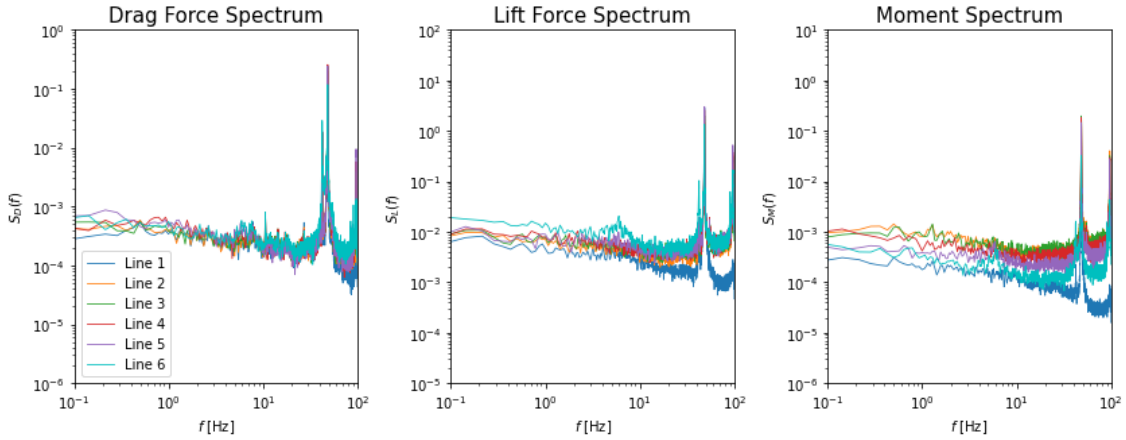
Figure 6.38 presents the buffeting force spectra of drag, lift and moment on the six correlation lines. Due to the high similarity between the correlation lines, except for some deviations for the moment, the two-dimensionality of the mean flow is confirmed. In addition, it confirms the consistency of the measurements.



(a) Force Spectra, Grid rotation of 0.5 Hz



(b) Force Spectra, grid rotation of 7 Hz



(c) Force Spectra, still open grid

Figure 6.38: Buffeting Force Spectra for different grid-generated turbulence's, $V \approx 9m/s$

6.6 Coherence

To describe the spatial distribution characteristics of the buffeting forces acting on the cross-section, the spanwise coherence is estimated. This has been done with the measured drag, lift, and moment forces from the piece-wise load method. The estimation is based on the cross-spectra between the forces in two different correlation strips with distance, Δy, and corresponding auto-spectra for the strips, according to Equation 3.80. Figure 6.39 and 6.40 illustrates the spanwise coherence between correlation strip one and the five other correlations strips with the distances $\Delta y = 10, 25, 55, 105, 220mm$. The coherence is found for the upstream-box, the downstream-box, and the total model for a grid-generated turbulence with a mean wind velocity of $V \approx 7m/s$ and $V \approx 9m/$, respectively.

The spanwise coherence at $f = 0$ decreases to a lower value with increasing separation distances for all the buffeting forces. It can be observed that the coherence approaches or are close to 1.0 at zero frequency for the nearest correlation strip. Even though the spanwise coherence is significantly lower for the distance, $\Delta y = 220$, coherence is still observed. Comparing the two figures with different wind velocity, it can be observed that the influence of the wind velocity is slight. This also applies to the angle of attack, which has been investigated for several tests. The results obtained from a study by Zhou et al.[51] emphasize the small impact of change in wind velocity and angle of attack.

Comparing the spanwise coherence for lift and moment on the upstream-box and the downstream-box, it can be seen that the coherence is much higher for the upstream-box. There is a particularly high variation between the downstream-box and the upstream-box for low frequencies. In addition, the spanwise coherence for the downstream box decay more rapidly in this frequency area. The overall spanwise coherence is, therefore, mainly affected by the upstream-box. The experimental study by Wang et al.[23] presented in Section 3.5 where the spanwise coherence of a twin-box girder and a closed-box girder were compared, also showed lower coherence for the downstream-box. This affected the overall spanwise coherence of the twin-box girder and was suggested to be the main factor for significantly less coherence on the twin-box girder compared to the closed-box girder.

Another remark on the spanwise coherence for lift and moment is that the downstream-box has higher coherence at around 40 Hz for $V \approx 7m/s$ and 50 Hz for $V \approx 9m/s$, where a peak is observed. This may be because of vortex shedding from the rear edge of the upstream-box. A study by Xia et al. [52] also observed that the coherence was affected by vortex shedding in the high-frequency region. For higher frequencies, the coherence is enhanced by vortex shedding effects, such that the turbulence intensity is decisive. Therefore, before the periodical vortices get suppressed by high turbulence intensity, the coherence is higher for the downstream-box.

In terms of the spanwise coherence of drag, the upstream-box has higher coherence than the downstream-box. However, the total coherence tends to pursue the downstream-box, which indicates a stronger influence. This has also been observed in the drag coefficients and the drag force spectra.

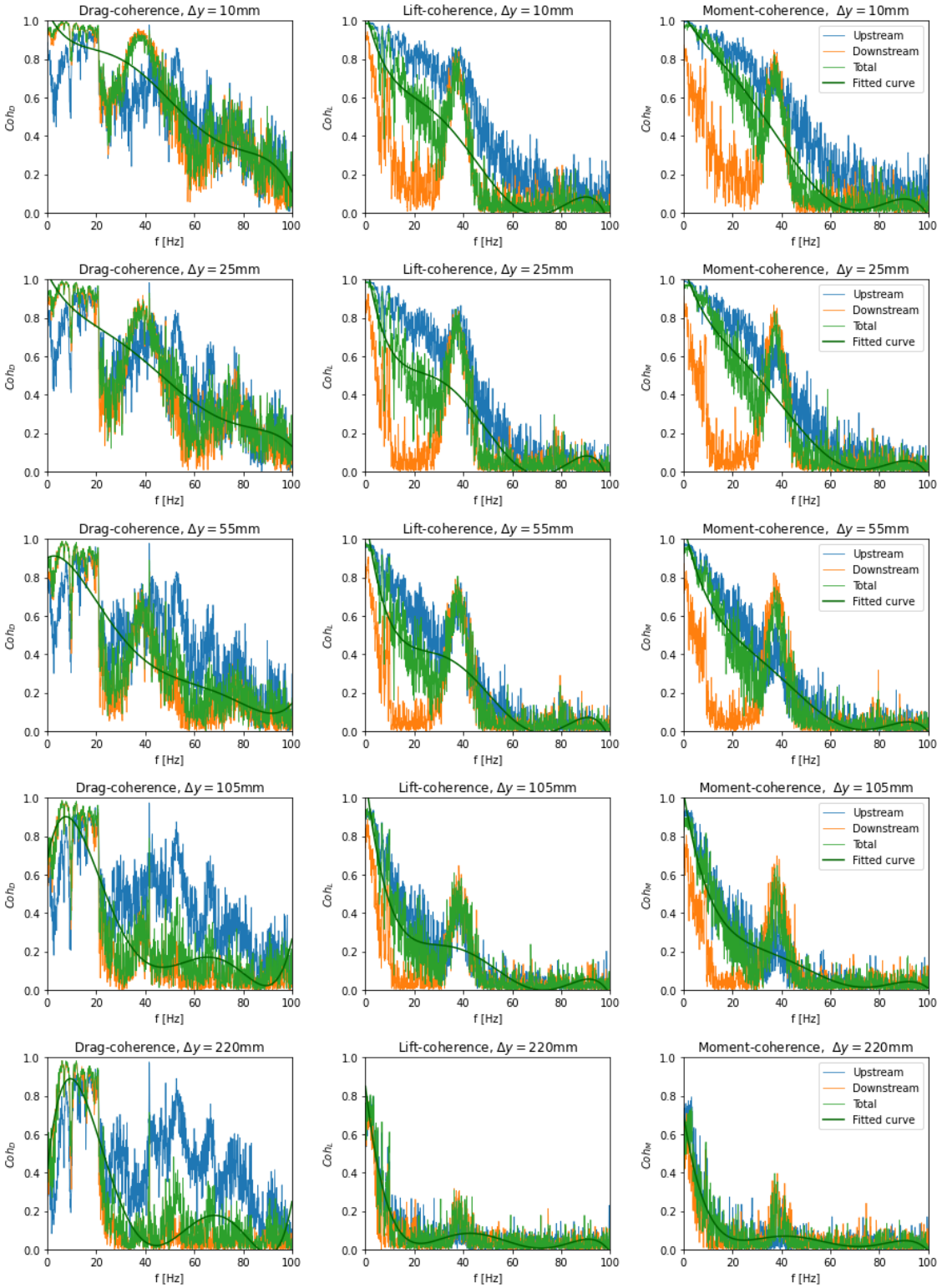


Figure 6.39: Spanwise coherence at $V \approx 7m/s$, grid rotation of 7 Hz

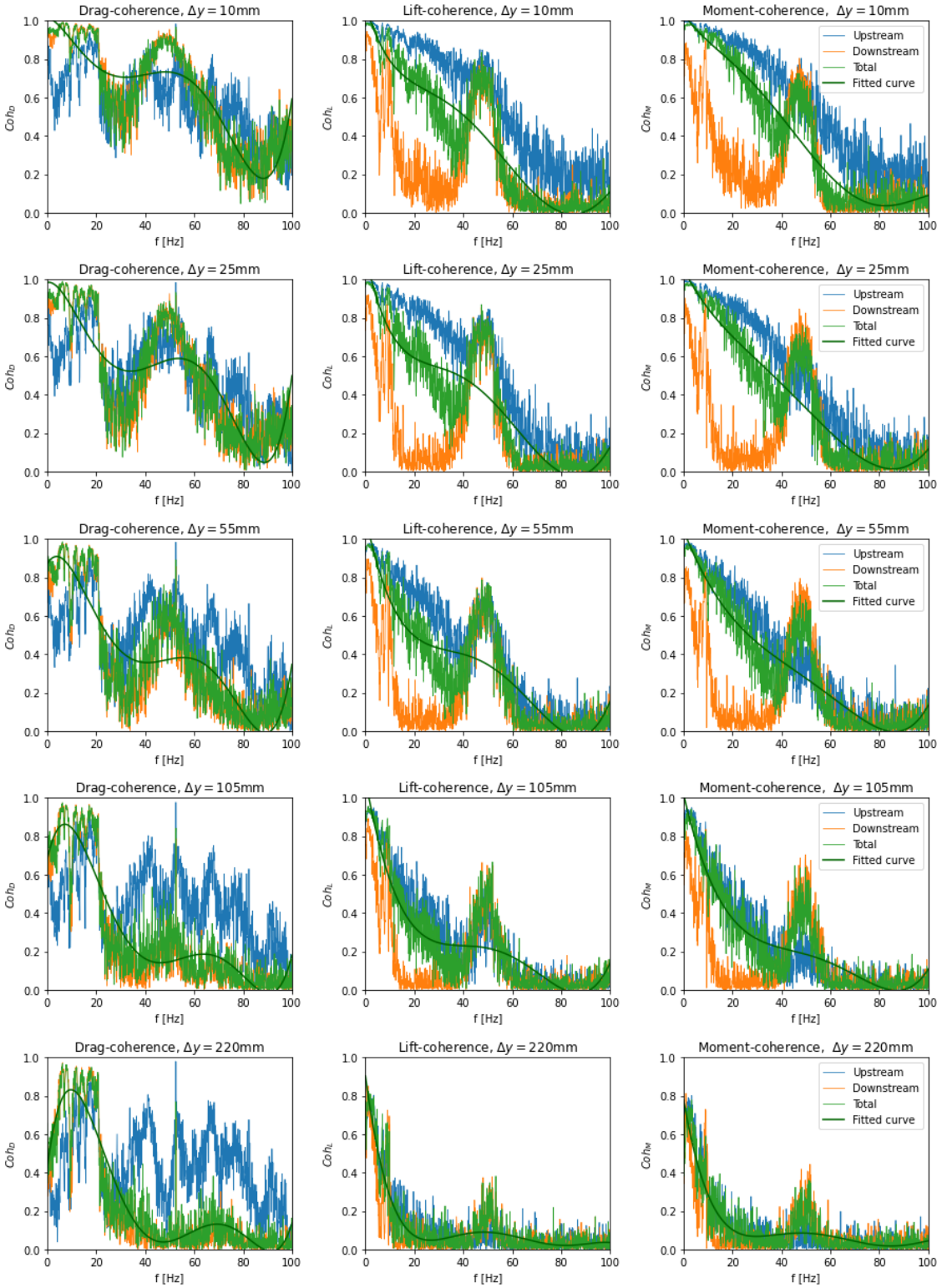


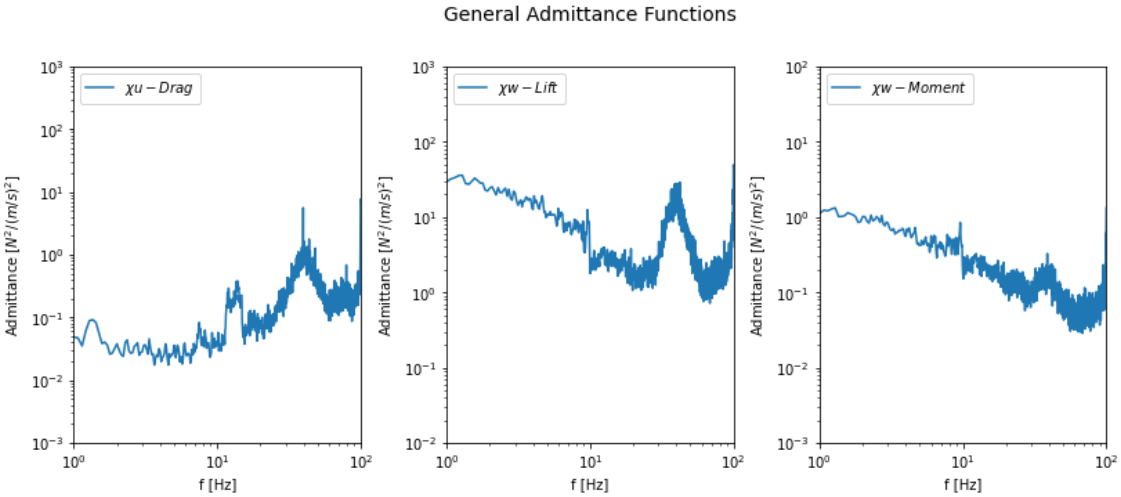
Figure 6.40: Spanwise coherence at $V \approx 9m/s$, grid rotation of 7 Hz

6.7 Aerodynamic Admittance Functions

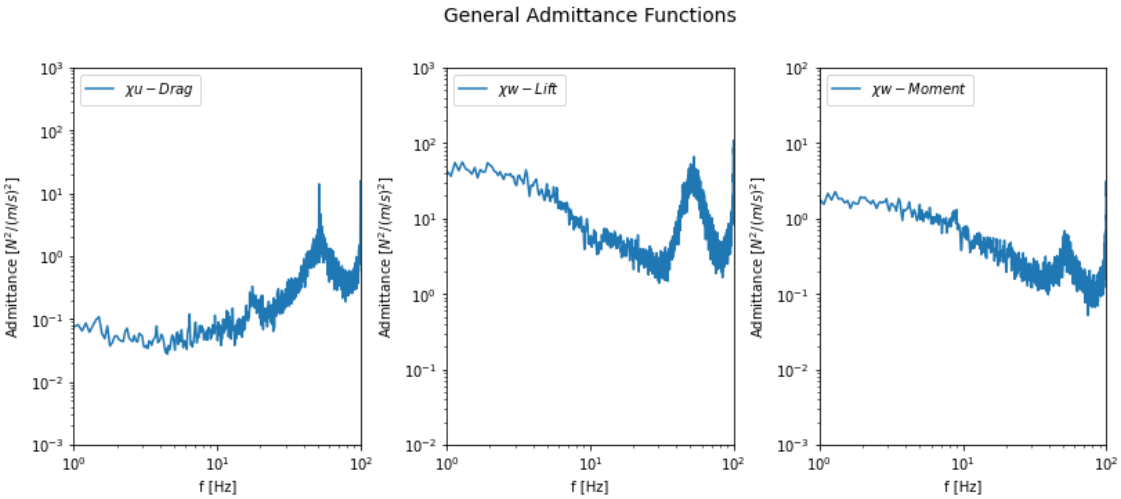
The aerodynamic admittance function is, as mentioned earlier, an important transfer function that transfers the turbulent wind to buffeting forces to estimate the buffeting response. The methods described in Section 3.6 are used to estimate the admittance functions. The first method presented is the general admittance method, where the admittance function is found by a transfer function between one force spectra and one turbulence spectra, as seen in Equation 3.81. The next method used is the auto-spectral method (the equivalent method), given in Equation 3.83, where the force spectra is related to both turbulence components u and w. Last, the cross-spectral method is used to estimate the admittance functions, as given in Equation 3.86. This method is based on the measured cross-spectra between the fluctuating force coefficients $C_F, (F = D, L, M)$ and each turbulence component, u and w. The auto-spectral and cross-spectral admittance functions are presented in the same figure for comparison.

6.7.1 General Admittance Functions

Figure 6.41 and 6.42 presents the general admittance function with the two different grid rotations of 0.5 Hz and 7 Hz and two different mean wind velocities.

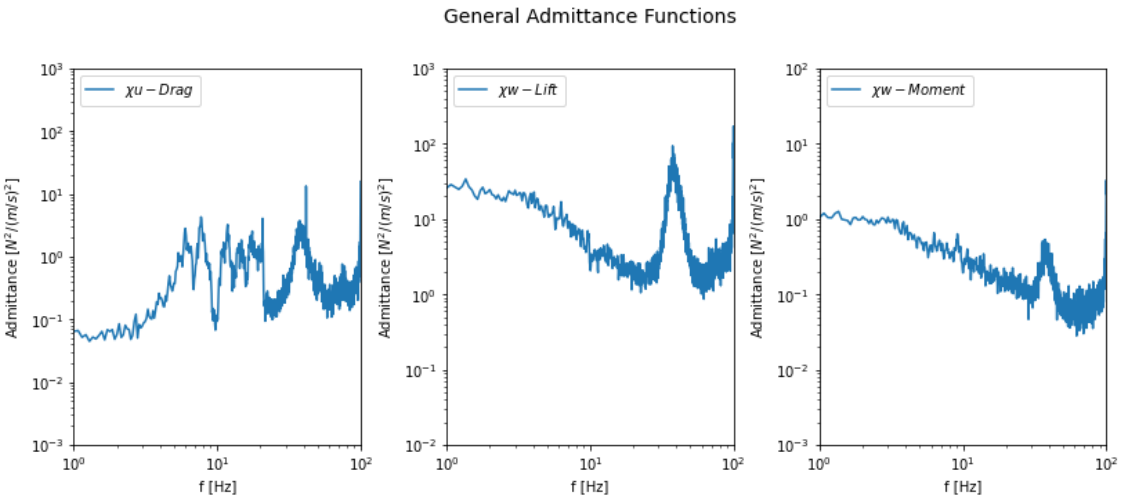


(a) Grid rotation of 0.5 Hz, V≈ 7 m/s

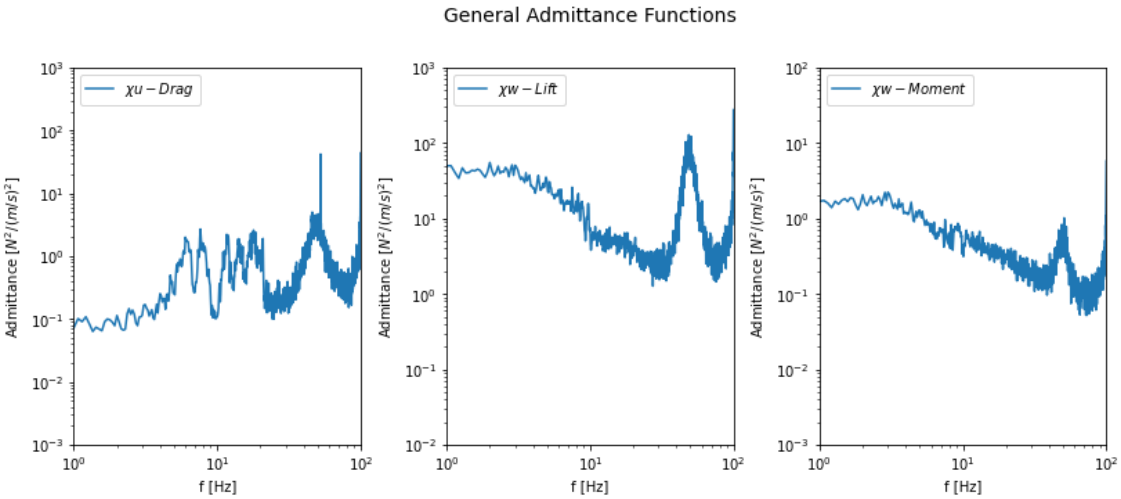


(b) Grid rotation of 0.5 Hz, V ≈ 9 m/s

Figure 6.41: General admittance functions, grid rotation of 0.5 Hz



(a) Grid rotation of 7 Hz, V ≈ 7 m/s



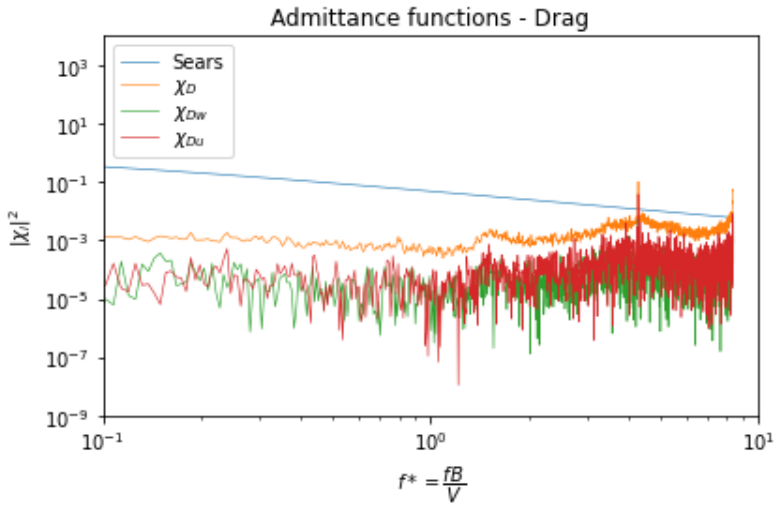
(b) Grid rotation of 7 Hz, V≈ 9 m/s

Figure 6.42: General admittance functions, grid rotation of 7 Hz

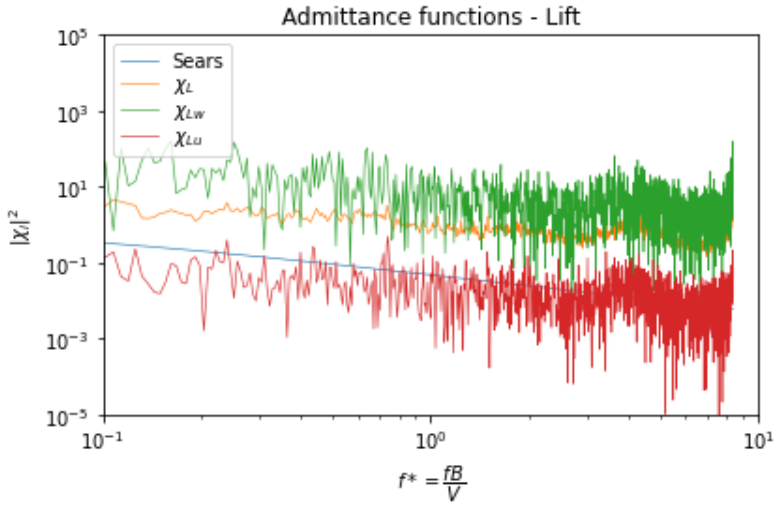
The plots of the general admittance functions for different wind velocity displays similarity, but vary some when the wind velocity increases. Since the general admittance function transfers velocity spectra to force spectra, the admittance functions should be close to similar for different velocities. In general, the deviation between the admittance function for higher wind velocity becomes smaller, and therefore it may be more accurate to present the admittance functions for higher wind velocities. This may explain why the deviation between the plots in Figure 6.41 and 6.42 is small for the wind velocities of 7 m/s and 9 m/s. Another observation is that the appearing peaks in the plots occur at approximately the same frequency as the peaks in the force spectra, discussed in Section 6.5. The general admittance function for lift is significantly higher than for drag and moment. Overall, the general admittance functions presented seem acceptable based on the observations.

6.7.2 Auto-spectral and Cross-spectral Admittance Functions

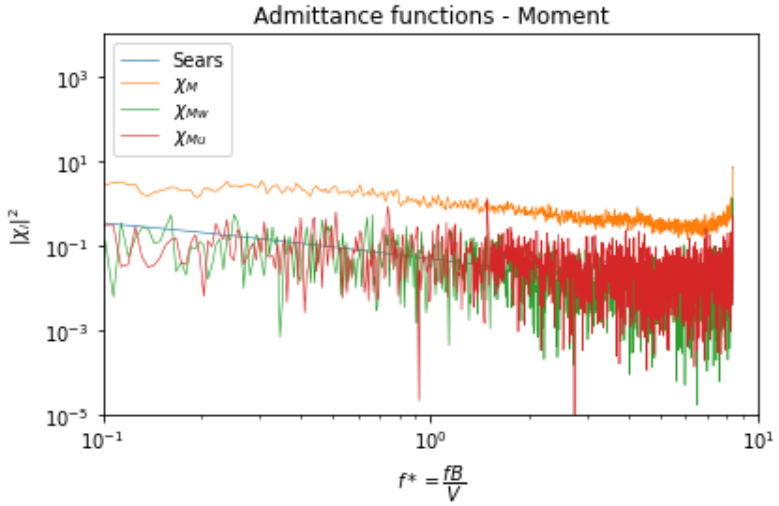
The admittance functions estimated by the auto-spectral method $|\chi_F|^2$ and the cross-spectral method $|\chi_{Fu}|^2$ and $|\chi_{Fw}|^2$, $(F = D, L, M)$ is presented in this section . To compare the identified admittance functions with the different approaches, the square of the AAF is presented in the figures together with the Sears function. The admittance functions are plotted in turbulent flow with grid rotation of 0.5 Hz, with wind velocity 9 m/s and three different angles of attack, -5°, -2°, 0°, 2°, and 5°. The admittance functions for the upstream-box and the downstream-box with the auto-spectral method are also presented to study the contribution from each deck.



(a) Admittance functions, drag

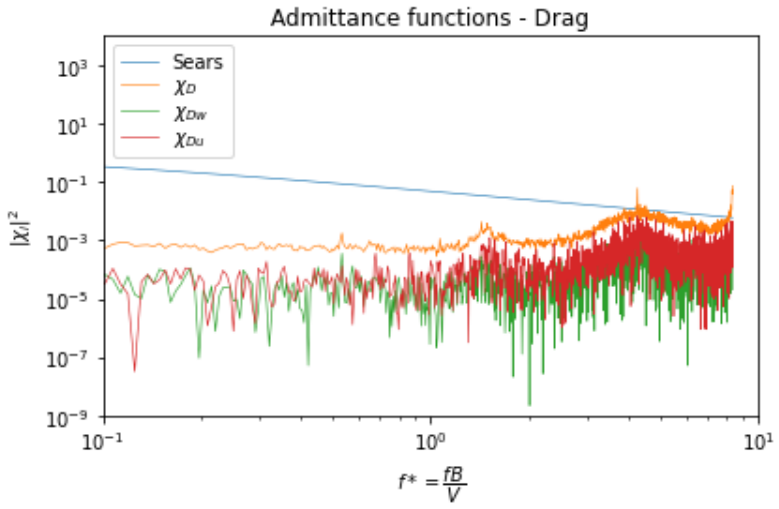


(b) Admittance functions, lift

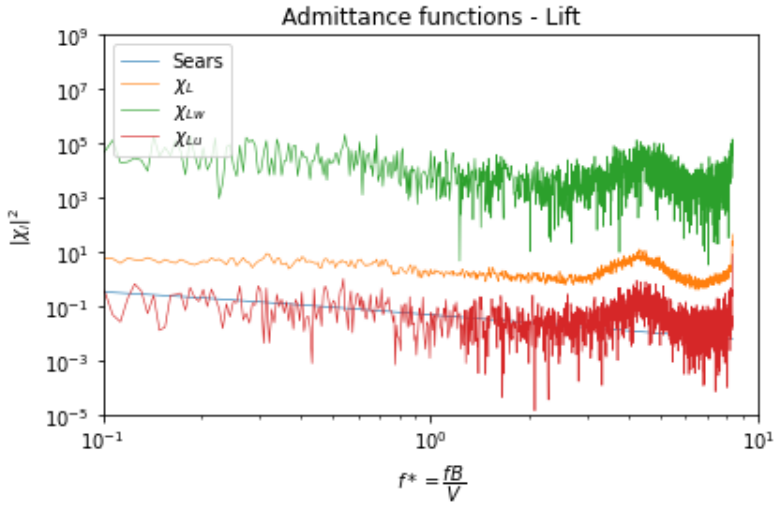


(c) Admittance functions, moment

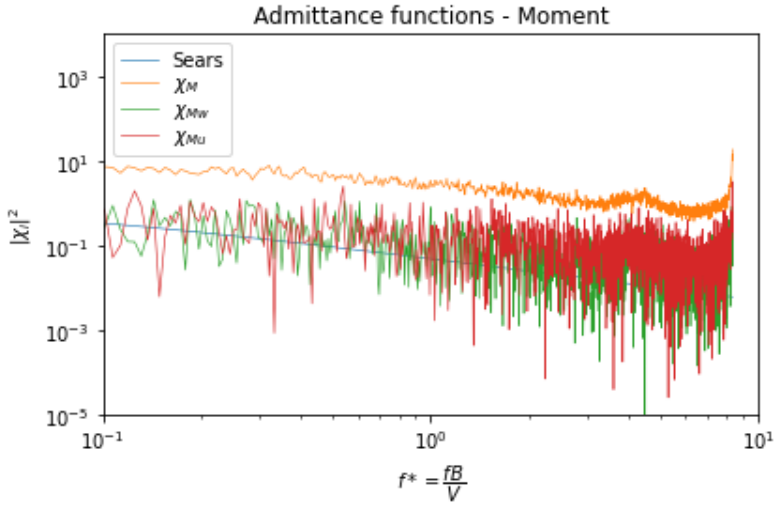
Figure 6.43: Admittance functions for drag, lift and moment, estimated by auto-spectral method and cross-spectral method, $\alpha = -5^\circ$ and $V \approx 9$.



(a) Admittance functions, drag

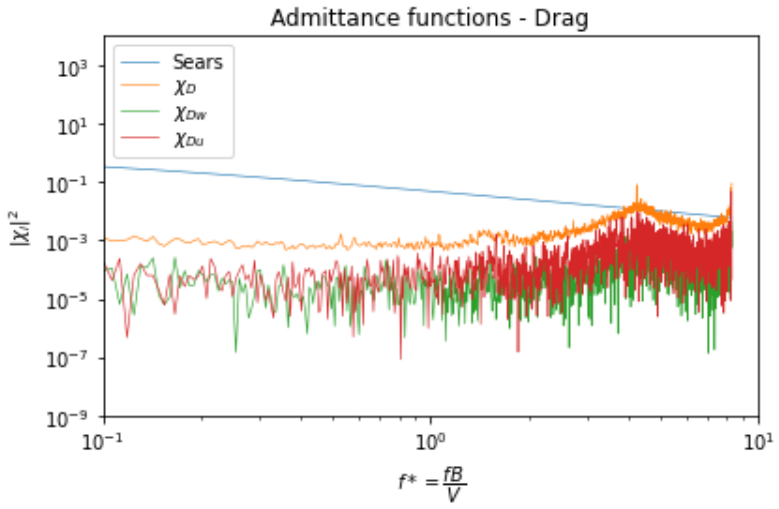


(b) Admittance functions, lift

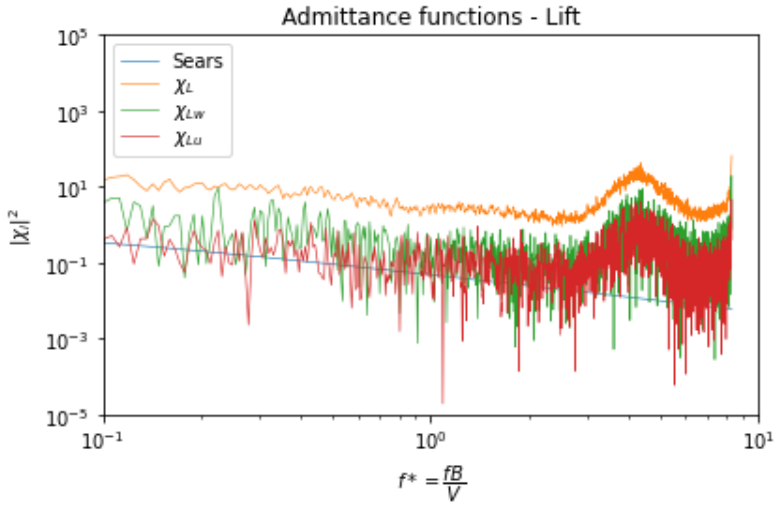


(c) Admittance functions, moment

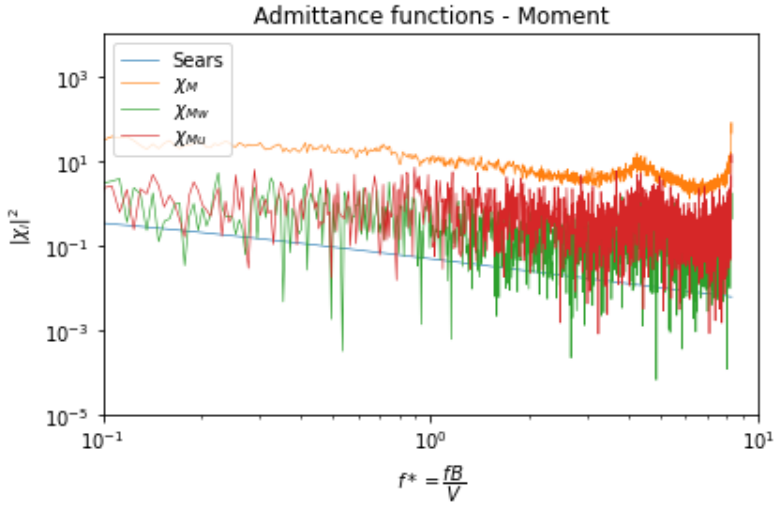
Figure 6.44: Admittance functions for drag, lift and moment, estimated by auto-spectral method and cross-spectral method, $\alpha = -2^\circ$ and $V \approx 9$.



(a) Admittance functions, drag

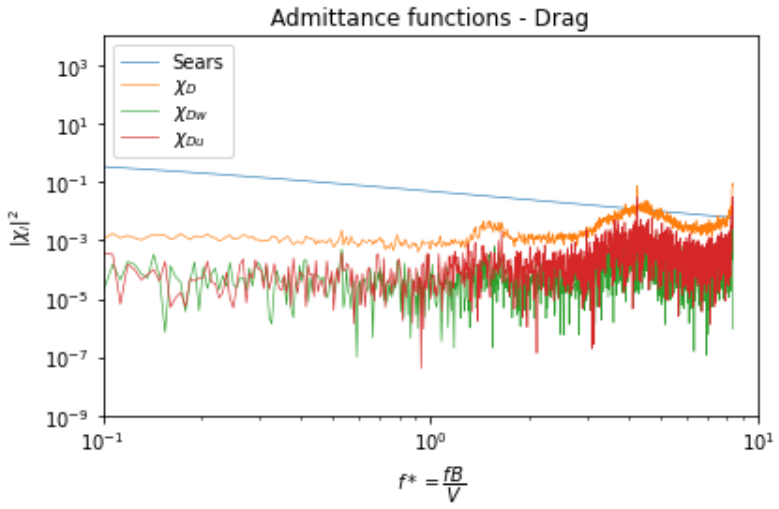


(b) Admittance functions, lift

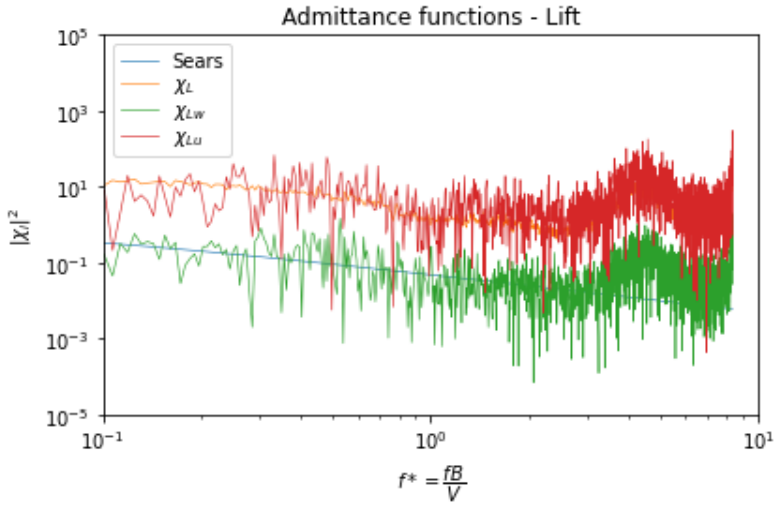


(c) Admittance functions, moment

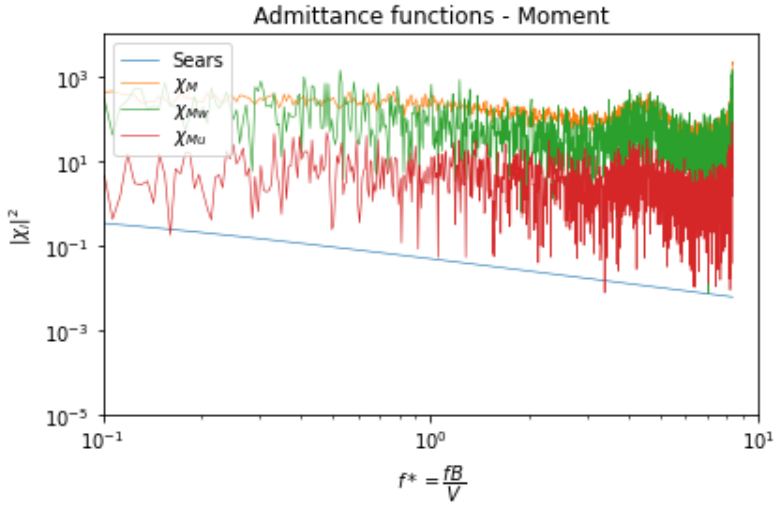
Figure 6.45: Admittance functions for drag, lift and moment, estimated by auto-spectral method and cross-spectral method, $\alpha = 0°$ and $V \approx 9$.



(a) Admittance functions, drag

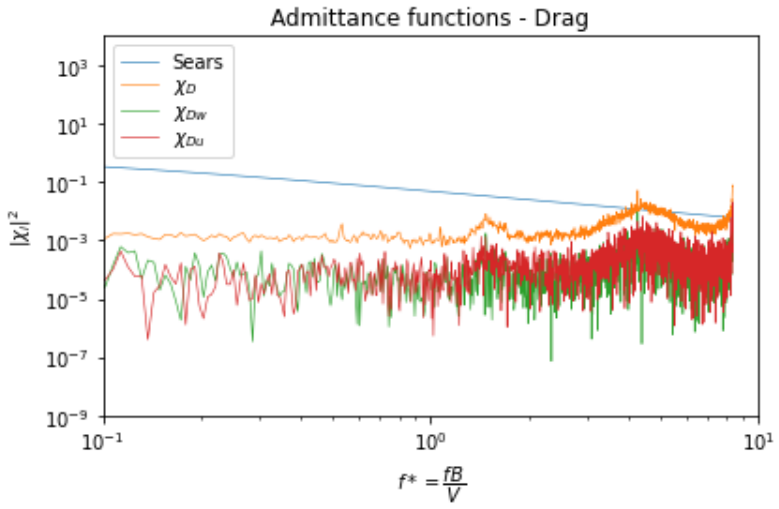


(b) Admittance functions, lift

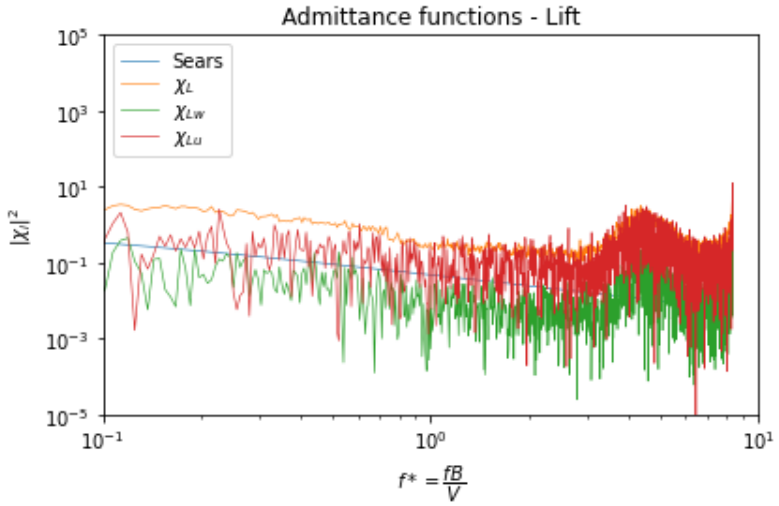


(c) Admittance functions, moment

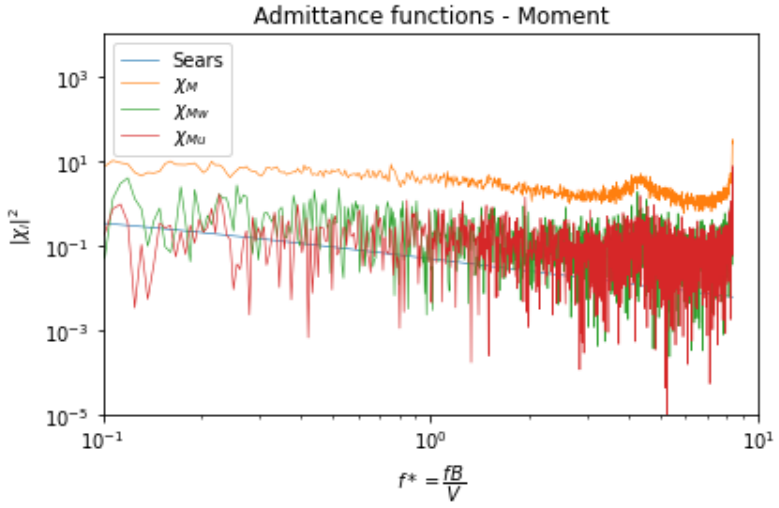
Figure 6.46: Admittance functions for drag, lift and moment, estimated by auto-spectral method and cross-spectral method, $\alpha = 2^\circ$ and $V \approx 9$.



(a) Admittance functions, drag



(b) Admittance functions, lift



(c) Admittance functions, moment

Figure 6.47: Admittance functions for drag, lift and moment, estimated by auto-spectral method and cross-spectral method, $\alpha = 5^\circ$ and $V \approx 9$.

Figure 6.43-6.47 presents the admittance functions for drag, lift and moment for five different angles of attack. A peak is observed for most admittance functions at all angles. The peak occurs approximately at the same frequency as the peak observed in the force spectra. Moreover, the admittance functions for drag display steady behaviour, and no noticeable change is observed for the different angles. The admittance functions for drag are located significantly lower than the admittance functions for lift and moment. This is mainly because the drag coefficients have a higher value than the coefficients for lift and moment. Further, the admittance functions $|\chi_{Du}|^2$ and $|\chi_{Dw}|^2$ are very similar to each other and both are located below $|\chi_D|^2$. This may imply that the auto-spectral method overestimates the admittance function for drag. $|\chi_{Du}|^2$ is slightly closer to $|\chi_D|^2$, which is common because the horizontal component, u, provides the main contribution to the drag admittance. When comparing the identified admittance functions for drag with the Sears function, it is observed that the Sears function deviates significantly from the identified admittance functions for drag as it overestimates the admittance function. The Sears function underestimates when it comes to the estimation of the admittance function for lift and moment.

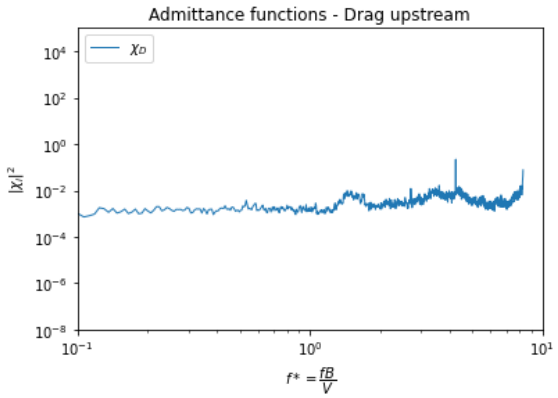
The estimated functions for lift display more variation when the angle of attack changes. For the angles $-5°$ and $-2°$, the auto-spectral admittance function $|\chi_L|^2$ lays between $|\chi_{Lw}|^2$ and $|\chi_{Lu}|^2$, where $|\chi_{Lw}|^2$ is higher. The deviation between $|\chi_{Lw}|^2$ and $|\chi_{Lu}|^2$ increases as the angle changes to $-2°$. According to the static coefficients, this seems reasonable. When the angle of attack is changed to $0°$, the cross-spectral admittance functions $|\chi_{Lw}|^2$ and $|\chi_{Lu}|^2$ becomes almost identical, and $|\chi_L|^2$ is now above both functions. Deviations between $|\chi_{Lw}|^2$ and $|\chi_{Lu}|^2$ is observed again when the angle is increased to $2°$ and $5°$, where $|\chi_{Lu}|^2$ and $|\chi_L|^2$ are is laying at approximately the same level, above $|\chi_{Lw}|^2$. The auto-spectral admittance function $|\chi_L|^2$ is a weighted average of $|\chi_{Lw}|^2$ and $|\chi_{Lu}|^2$ and is often close to $|\chi_{Lw}|^2$. This is only observed for negative angles of attack, which may indicate that the auto-spectral method underestimate the admittance functions for lift for positive angles of attacks.

As for the estimated admittance functions for moment, the identified moment admittance functions for an angle of attack of $2°$ deviate compared to the other angles. Both admittance functions from the auto-spectral and cross-spectral method are located at a significantly higher level for an angle of attack of $2°$. This is because the static coefficient for the moment is very close to zero for this angle, which causes the admittance functions to lay at a much higher level. Further, it is observed that the auto-spectral function $|\chi_M|$ is above both $|\chi_{Dw}|^2$ and $|\chi_{Du}|^2$ for all angles. For the angle of attack $2°$, it is observed that $|\chi_{Dw}|^2$ is closer to $|\chi_M$, since the static coefficient C'_M is smaller than C_M.

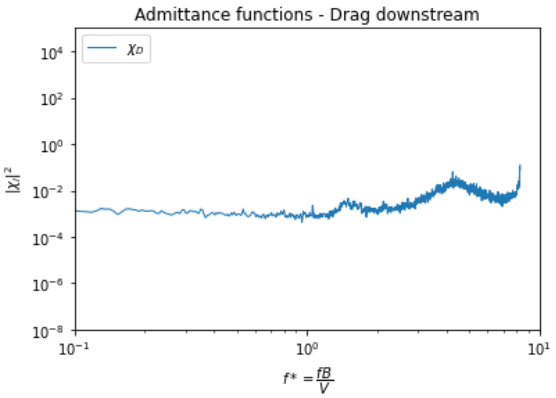
The admittance functions were also compared with different wind velocities. No significant deviations in the shapes were observed, and the admittance functions showed great similarity. Therefore, the admittance functions were only studied with a wind velocity of 9 m/s.

The upstream-box and the downstream-box

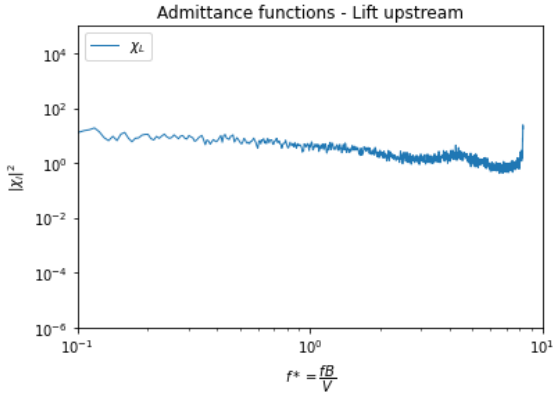
The auto-spectral admittance functions for the upstream-box and the downstream-box with an angle of attack 0° and wind velocity 7 m/s and 9 m/s is shown in Figure 6.48.



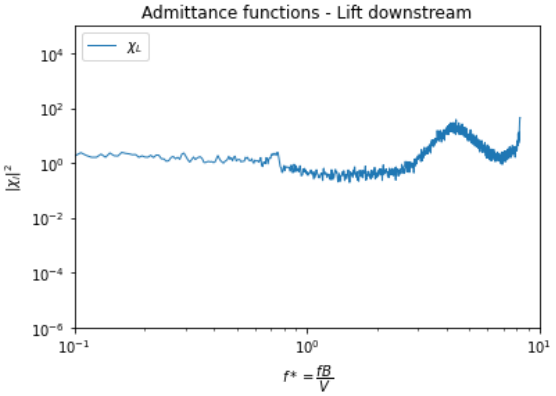
(a) Admittance function drag, V ≈ 7 m/s



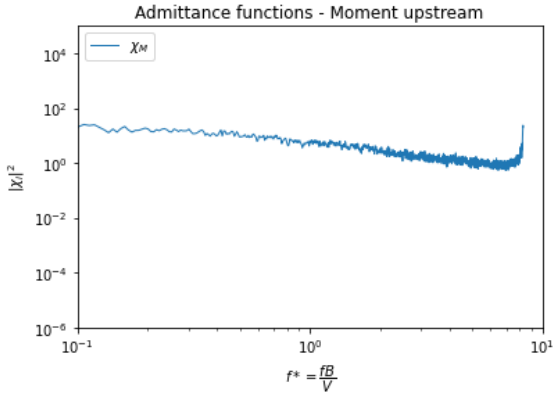
(b) Admittance function drag, V ≈ 9 m/s



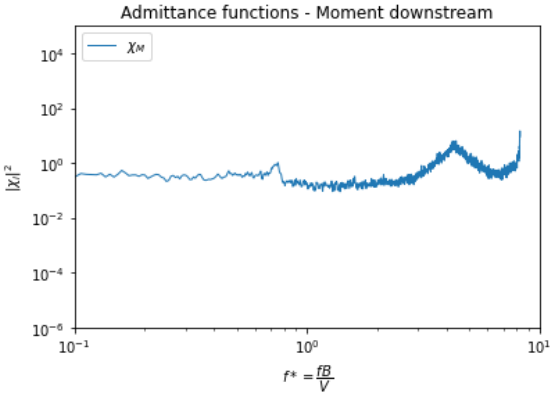
(c) Admittance function lift, V ≈ 7 m/s



(d) Admittance function lift, V ≈ 9 m/s



(e) Admittance function moment, V ≈ 7 m/s

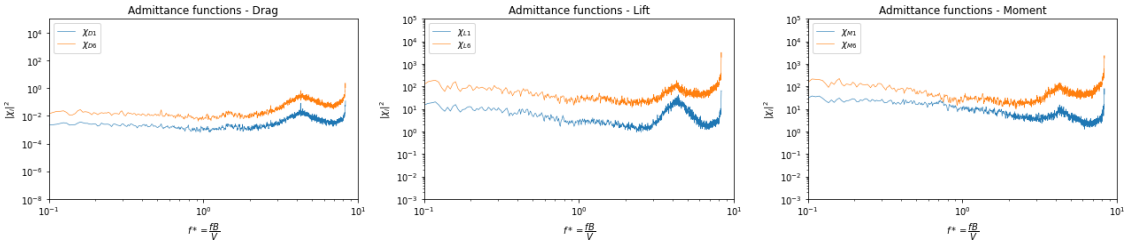


(f) Admittance function moment V ≈ 9 m/s

Figure 6.48: Admittance functions for drag, lift and moment for the upstream-box and the downstream-box, estimated with auto-spectral, $\alpha = 0°$.

The distinct peak observed in the $|\chi_F|^2$ is mainly caused by the downstream-box. This peak occurs due to vortex shedding from the trailing edge of the upstream-box, as previously discussed in Section 6.5. The shape of the admittance functions shows similarity for all buffeting forces, but the magnitude for the estimated $|\chi_D|^2$ for both the upstream-box and downstream-box differs significantly compared to $|\chi_L|^2$ and $|\chi_M|^2$. This is expected, since the same trend is observed in the drag force spectra.

As mentioned previously, the admittance functions presented in this thesis are estimated with the assumption of 2-dimensional turbulence. Hence, the span-wise correlation is neglected. In order to investigate this assumption further, the admittance functions are studied for correlation line one and six, see Figure 6.49. A deviation between the admittance functions for the two correlation lines is observed, which indicates that the assumption of 2-dimensional turbulence is incorrect. Therefore, when estimating the admittance functions, the 3-dimensional effects should be considered.



(a) Admittance functions, drag (b) Admittance functions, lift (c) Admittance functions, moment

Figure 6.49: Admittance functions for drag, lift and moment, at correlation line one and six, $\alpha = 0°$ and $V \approx 9$.

Validation of the Estimated Aerodynamic Admittance Functions

The estimated admittance functions obtained by the auto-spectral method and the cross-spectral method identifies different admittance functions. For some angles of attack, the deviation between the admittance function for the horizontal and vertical turbulence components is more evident, which underlines the importance of identifying the admittance functions related to both turbulence components, u and w, for each buffeting force. The auto-spectral method assumes that the χ_{iu} and χ_{iw} are equal to each other, which can result in inaccurate results. It should be mentioned that the cross-spectral method does not include the auto-spectral equations of buffeting forces. This could cause a low accuracy of the identified admittance functions because the correlation between the buffeting forces and fluctuating wind velocities is relatively weak [30].

When comparing the Sears function with the identified admittance functions, it was detected that the Sears function overestimates the admittance functions for drag and underestimates for lift and moment. It should be mentioned that the admittance functions obtained by the cross-spectral method are slightly closer to the Sears function for lift and moment and share the same changing rate. The Sears function is theoretically derived for vertical components for the admittance functions χ_{Lw} and χ_{Mw} of an airfoil. This explains why the admittance functions for drag deviate significantly from the Sears function and why the Sears function does not apply to the twin-box bridge.

Another approach for estimating the admittance functions is the colligated residue least square method of auto and cross spectra (CRLSMACS), as described in Section 3.4.3. In a study by Yan et al.[30], the two methods used in this thesis and the CRLSMACS method were used to estimate

the aerodynamic admittance functions for lift and moment. The study showed that, χ_{Fw} estimated by the cross-spectral method is much smaller than the corresponding ones estimated by the auto-spectral method and CRLSMACS. This observation coincides with the results obtained in this thesis for most of the angles of attack. Further, the results showed that the auto-spectral method is very close to the χ_{Fw} obtained by the CRLSMACS method. The main difference between the results obtained by Yan et al. and the results obtained in this thesis is that the auto-spectral admittance is located above the cross-spectral admittance for most angles. In contrast, in the results obtained by Yan et al. where the auto-spectral admittance is located in the middle of χ_{Fu} and χ_{Fw}. The results from the study by Yan et al. are obtained for a closed-box deck with a different cross-section, which affect the measurements. A twin-box bridge, like the one studied in this thesis, can be more complex due to the gap between the girders and the geometry. Therefore, it may be inaccurate to compare the results for a closed-box with the results for a twin-box due to these differences.

Many factors will affect the result in the end. In this thesis, the admittance functions are identified for several angles of attack. A previous master's thesis [44] estimated the admittance functions for a twin-box bridge for 0° angle of attack. The admittance functions obtained in this thesis show some similarity in regards to the placement of $|\chi_F|^2$ when compared to the results from the previous master's thesis. Moreover, studies on estimating aerodynamic admittance functions for a twin box bridge are limited, and few estimate the admittance functions for drag. Many studies only estimate the admittance functions for lift and moment and only focus on the angle of attack of 0°. Therefore, it makes it difficult to compare and validate the results. Nevertheless, based on the results obtained in this master's thesis, as the force spectra and static coefficients, the estimated admittance functions seems reasonable and may be a valid representation of the buffeting forces acting on the twin-box.

Chapter 7

Conclusion and Further Work

7.1 Conclusion

This thesis aimed to investigate the pressure distribution of a twin-box bridge and estimate the aerodynamic admittance functions. A section model of a twin-box bridge was built and tested in a wind tunnel for different wind velocities, angles of attack, and active grid-generated turbulence. The mid-section for the placement of 256 pressure tubes was designed in SolidWorks and 3D-printed at the Department of Manufacturing and Civil Engineering at NTNU Gjøvik. The pressure tubes were distributed equally on the two boxes along six correlation lines.

The pressure distribution on the twin-box was investigated and concluded to be reasonable. The pressure distribution on the upstream-box changed from having the maximum negative pressure on the upper surface of the leading edge to the lower surface as the angle of attack decreased. For the downstream-box, the pressure distribution showed similarities for the different angles of attack. The negative pressure on the leading edge caused by vortex-shedding decreased slowly as the angle of attack decreased. However, an unexpected negative pressure was observed on the upstream-box at edge S6. The sharp angle on the leading edge of the upstream-box could be the main reason for this. Another contribution may be due to negative pressure in the wind tunnel. When comparing the effect of railings on the pressure distribution, there were observed similarities, except in the vicinity of the railings.

The turbulence spectra were studied and concluded to be a reasonable representation of the turbulence wind field. Further, the static forces from the pressure measurements were estimated using two different methods; the piece-wise load method and the interpolated load method. A slight difference was observed when comparing the methods. It was also observed that the drag forces obtained from the pressure measurements were significantly lower than those obtained from the load cells, and the results deviated more than for the lift force and moment. An explanation for this could be that the pressure tubes cannot measure the friction forces.

Moreover, the static load coefficients were estimated using the loads from both the pressure measurements and the load cells. The coefficients for lift and moment showed similarities, but the drag coefficients deviated. However, this was expected since the drag forces from the pressure measurements were lower. The most significant contribution to the lift coefficient, C_L and moment coefficient, C_M came from the upstream-box for all angles of attack. However, the downstream box had the highest contribution to the drag coefficient for angles $\alpha > -2°$, which was somewhat unexpected. This was observed in the coefficients obtained from the pressure measurements

and the load cells. Large negative pressure caused by vortex-shedding on the leading edge of the downstream-box could be the explanation.

The force spectra based on the loads from the piece-wise load method were examined. Several peaks were observed at a frequency of 50 Hz, which is most likely caused by electric noise and vortex shedding. The buffeting force spectra for the different correlation lines were compared and confirmed the consistency of the measurements. The spanwise coherence was also investigated and showed that the upstream-box mainly affected the overall spanwise coherence for lift and moment.

Further, the aerodynamic admittance functions were estimated by using the three different methods; the general, the auto-spectral, and the cross-spectral method. The methods displayed a peak at approximately 50 Hz, mainly caused by the downstream-box due to vortex shedding. The change in the admittance functions was not very noticeable when the wind velocity increased, indicating that the admittance functions present more accurate results with higher wind velocities. The drag admittance function was displayed at a lower magnitude than the admittance functions for lift and moment for all methods.

The results shows that the admittance functions obtained by the cross-spectral method deviated compared to the admittance functions from the auto-spectral method. The admittance functions $|\chi_{Du}|^2$ and $|\chi_{Dw}|^2$ obtained by the cross-spectral method were very similar to each other and were located below $|\chi_D|^2$, which could imply that the auto-spectral overestimated the admittance function for drag. This was also observed for the admittance functions for moment. For the lift admittance functions, the results varied for the different angles of attack. This seems reasonable due to the value of the static coefficients. The auto-spectral admittance function is a weighted average of the cross-spectral function and is often close to $|\chi_{Fw}|^2$. This was only observed for negative angles of attack, implying that the auto-spectral method underestimates the admittance functions for lift for positive angles of attack. Further, the identified admittance functions for moment for the angle of attack of 2° deviated compared to the other angles, which could be explained by the very low value of the static coefficient. The Sears function was presented together with the identified admittance functions for comparison. It was detected that the Sears function overestimated the admittance functions for drag and underestimated for moment and lift. In addition, it was observed that cross-spectral functions for lift and moment were slightly closer to the Sears function because the Sears function is theoretically derived for the vertical components for the admittance functions $|\chi_{Lw}|^2$ $|\chi_{Mw}|^2$ of airfoil.

Based on the observation, the auto-spectral method seems to estimate inaccurate admittance functions. This underlines the importance of identifying the admittance functions related to turbulence components, u, and w. Moreover, the cross-spectral method does not include the auto-spectral equations of buffeting forces, which could cause low accuracy of the identified admittance functions. Therefore, further investigation of the six admittance functions could be done by the using the CRLSMACS method.

Nevertheless, based on the results obtained in this master's thesis, as the force spectra and static coefficients, the estimated admittance functions seems reasonable and may be a valid representation of the buffeting forces acting on the twin-box.

7.2 Further work

The pressure distribution and wind data contains much information and for future work it is therefore recommended to evaluate this further:

- The pressure distribution showed an unexpected large amount of negative pressure on the upstream-box for angle of attack = 0°, which could be caused by the sharp angle on the leading edge. Therefore, different cross-sections could be further investigated to determine to what extent the pressure distribution is effected by the angle.
- Investigate the influence of turbulence length scales on the fluctuating wind effect and pressure distribution on the twin-box.
- Investigate the coherence between the pressure distribution on the two boxes. Studying the coherence between a tube on the upstream-box and a tube on the downstream-box will, in addition, give a better understanding of the turbulent flow.
- Study the effect of guide vanes on the pressure distribution.
- Investigate the effect of different gap widths on the pressure distribution and the vortex shedding.

In addition, the estimation of the aerodynamic admittance functions should be investigated further. The auto-spectral method tends to overestimate the admittance functions compared to the cross-spectral method, while the cross-spectra method may not be an accurate representation of the six admittance functions. Hence, further research are suggested:

- The CRLSMACS method should be conducted for identifying the six aerodynamic admittance function and for comparison with the auto-spectral and cross-spectral method.
- Investigate the 3-dimensional effect of turbulence (3D AAF) and adopt the flow tests into the calculations.

Moreover, as mentioned in Section 5.3, two tubes were squeezed due to being too close to the aluminium pipe and the 3D-printed part. Therefore, another suggestion for future work is to optimize the 3D-part and the placement of the pressure tubes further.

Bibliography

[1] H. Tanaka. 'Similitude and modelling in wind tunnel testing of bridges'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 33 (1990), pp. 283–300.

[2] G. Diana et al. 'Wind tunnel tests and numerical approach for long span bridges: The Messina bridge'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 122 (2013), pp. 38–49. ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2013.07.012.

[3] J. Wang et al. 'Experimental study on aerodynamic admittance of twin-box bridge decks'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 198 (2020), p. 104080. ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2019.104080.

[4] B. Wu and S. Laima. 'Experimental study on characteristics of vortex-induced vibration of a twin-box girder and damping effects'. In: *Journal of Fluids and Structures* 103 (2021), p. 103282. ISSN: 0889-9746. DOI: https://doi.org/10.1016/j.jfluidstructs.2021.103282.

[5] Statens vegvesen. *Ferjefri E39*. Jan. 2022. URL: https://www.vegvesen.no/vegprosjekter/europaveg/ferjefrie39/.

[6] Statens vegvesen. *Vegvesenet har nå valgt hvem som skal utarbeide forprosjekt for bruer over Halsafjorden og Sulafjorden*. May 2021. URL: https://www.vegvesen.no/vegprosjekter/europaveg/e39halsafjorden/nyhetsarkiv/vegvesenet-har-na-valgt-hvem-som-skal-utarbeide-forprosjekt-for-bruer-over-halsafjorden-og-sulafjorden/.

[7] E. Strømmen. *Theory of Bridge Aerodynamics*. 2nd ed. Springer, 2010.

[8] J.A. Jurado et al. *Bridge Aeroelasticity: Sensitivity Analysis and Optimal Design*. WIT Press, 2011.

[9] Y. Tamura and A. Kareem. *Advanced Structural Wind Engineering*. Springer, 2013.

[10] C.P.W. Geurts. 'The use of Wind tunnel experiments for wind loads on structures.' In: *TNO Built Environment and Geosciences, Delft, The Netherlands* (2005).

[11] G. L. Larose. *The Response of a Suspension Bridge Deck to Turbulent Wind: The Taut Strip Model Approach*. 1992.

[12] H. Tanka. 'Similitude and modelling in bridge aerodynamics'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 33 (1990), pp. 283–300.

[13] C. Geurts and van C. Bentum. *Wind Effects on Buildings and Design of Wind-Sensitive Structures*. Ed. by Ted Stathopoulos and Charalambos C. Baniotopoulos. Springer Vienna, 2007, pp. 31–65. ISBN: 978-3-211-73076-8. DOI: 10.1007/978-3-211-73076-8_2.

[14] K.C.S. Kwok et al. 'Wind-induced pressures around a sectional twin-deck bridge model: Effects of gap-width on the aerodynamic forces and vortex shedding mechanisms'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 110 (2012), pp. 50–61. ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2012.07.010.

[15] G. Schewe and A. Larsen. 'Reynolds number effects in the flow around a bluff bridge deck cross section'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 74-76 (1998), pp. 829–838. ISSN: 0167-6105. DOI: https://doi.org/10.1016/S0167-6105(98)00075-0. URL: https://www.sciencedirect.com/science/article/pii/S0167610598000750.

[16] C. W. Higgins et al. 'The Effect of Scale on the Applicability of Taylor's Frozen Turbulence Hypothesis in the Atmospheric Boundary Layer'. In: *Boundary-Layer Meteorol* 143 (2012), pp. 379–391. DOI: https://doi.org/10.1007/s10546-012-9701-1.

[17] G. I. Taylor. 'The Spectrum of Turbulence'. In: *The Royal Society* 164 (1938). ISSN: 2053-9169. DOI: http://doi.org/10.1098/rspa.1938.0032.

[18] L. Zhao et al. 'Revisiting aerodynamic admittance functions of bridge decks'. In: *Journal of Zhejiang University-SCIENCE A* 21 (July 2020), pp. 535–552. DOI: 10.1631/jzus.A1900353.

[19] F. Tubino. 'Relationships among aerodynamic admittance functions, flutter derivatives and static coefficients for long-span bridges'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 93.12 (2005), pp. 929–950. ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2005.09.002.

[20] Y. Su et al. 'Buffeting Response Prediction of Long-Span Bridges Based on Different Wind Tunnel Test Techniques'. In: *Applied Sciences* 12.6 (2022). DOI: 10.3390/app12063171.

[21] G. L. Larose. *The dynamic action of gusty winds on long-span bridges*. 1997.

[22] M. Li et al. 'Direct measurement of the Sears function in turbulent flow'. In: *Journal of Fluid Mechanics* 847 (July 2018), pp. 768–785.

[23] J. Wang et al. 'Experimental study on aerodynamic admittance of twin-box bridge decks'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 198 (2020). ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2019.104080.

[24] S. Li et al. 'The lift on an aerofoil in grid-generated turbulence'. In: *Journal of Fluid Mechanics* 771 (2015), pp. 16–35. DOI: ishttps://doi.org/10.1017/jfm.2015.162.

[25] H. Liu et al. 'Identification and Application of the Aerodynamic Admittance Functions of a Double-Deck Truss Girder'. In: *Applied Sciences* 9.9 (2019). ISSN: 2076-3417. URL: https://www.mdpi.com/2076-3417/9/9/1818.

[26] T.T. Ma et al. 'Investigations of aerodynamic effects on streamlined box girder using two-dimensional actively-controlled oncoming flow'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 122 (2013), pp. 118–129. ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2013.07.011.

[27] L. Zhao and Y Ge. 'Cross-spectral recognition method of bridge deck aerodynamic admittance function'. In: *Earthquake engineering and engineering vibration* 14.4 (2015), pp. 595–609. DOI: https://doi.org/10.1007/s11803-015-0048-8.

[28] L. Yan et al. 'Identification of aerodynamic admittance functions of a flat closed-box deck in different grid-generated turbulent wind fields'. In: *Advances in Structural Engineering* 21.3 (2018), pp. 380–395. DOI: 10.1177/1369433217718985.

[29] Y. Han et al. 'New estimation methodology of six complex aerodynamic admittance functions'. In: *Wind and Structures An International Journal* 13 (May 2010). DOI: 10.12989/was.2010.13.3.293.

[30] L. Zhu et al. 'Identification and application of six-component aerodynamic admittance functions of a closed-box bridge deck'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 172 (2018), pp. 268–279. ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2017.11.002.

[31] J. Wang et al. 'Influence of gap width on buffeting force spatial correlation and aerodynamic admittance of twin-box bridge deck'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 207 (2020). ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2020.104392.

[32] H. Liu et al. 'Identification and application of the aerodynamic admittance functions of a double-deck truss girder'. In: *Applied Sciences* 9.9 (2019), p. 1818. DOI: https://doi.org/10.3390/app9091818.

[33] S. E. Horg and S. B. Aas. *Wind Tunnel Testing of Bridge Decks*. 2016. URL: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2407302.

[34] Y. Kubo et al. 'Effects of end plates and blockage of structural members on drag forces'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 32.3 (1989), pp. 329–342. ISSN: 0167-6105. DOI: https://doi.org/10.1016/0167-6105(89)90006-8.

[35] L. Liu et al. 'Numerical and Experimental Studies on Grid-Generated Turbulence in Wind Tunnel'. In: *Journal of Engineering Science and Technology Rewiew* 10.3 (2017), pp. 159–169. ISSN: 1791-2377. DOI: 10.25103/jestr.103.21.

[36] R. Tresso et al. 'Homogeneous, Isotropic Flow in Grid Generated Turbulence '. In: *Journal of Fluids Engineering* 122.1 (Nov. 1999), pp. 51–56. ISSN: 0098-2202. DOI: 10.1115/1.483226.

[37] X. Wang et al. 'Effects of tube system and data correction for fluctuating pressure test in wind tunnel'. In: *Chinese Journal of Aeronautics* 31.4 (2018), pp. 710–718. ISSN: 1000-9361. DOI: https://doi.org/10.1016/j.cja.2018.01.021.

[38] Y.C. He et al. 'Accurate estimation of tube-induced distortion effects on wind pressure measurements'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 188 (2019), pp. 260–268. ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2019.02.017.

[39] Statens Vegvesen. *N400 Bruprosjektering*. Jan. 2022.

[40] M. Shinozuka and G. Deodatis. 'Simulation of Stochastic Processes by Spectral Representation'. In: *Applied Mechanics Reviews* 44.4 (Apr. 1991), pp. 191–204. ISSN: 0003-6900. DOI: 10.1115/1.3119501.

[41] E. A. Sivertsen and H. S. H. Strehl. *Flutter Analysis of Twin-Deck Configurations for a Suspension Bridge Crossing the Sulafjord*. 2021. URL: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2824697.

[42] D. Rocchi et al. 'Pressure distribution and global forces on a bridge deck section: experimental and CFD analysis of static aerodynamic forces'. In: *Journal of Bridge Engineering* 20.9 (2015). DOI: 10.1061/(ASCE)BE.1943-5592.0000695.

[43] Dassault Systèmes SolidWorks. *Introducing SolidWorks*. Dassault Systèmes SolidWorks Corporation. n.d. URL: https://files.solidworks.com/pdf/introsw.pdf.

[44] S. R. Haldorsen and T. Jahren. *Estimation of Aerodynamic Admittance Functions for a Twin-box Bridge*. 2020. URL: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2780047.

[45] J. R. Grongstad and O. Kildal. *Wind Tunnel Testing of bridge decks*. 2018. URL: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2575853.

[46] B. Siedziako and O. Øiseth. 'On the importance of cross-sectional details in the wind tunnel testing of bridge deck section models'. In: *Procedia Engineering* 199 (2017). X International Conference on Structural Dynamics, EURODYN 2017, pp. 3145–3151. ISSN: 1877-7058. DOI: https://doi.org/10.1016/j.proeng.2017.09.573.

[47] S. Laima et al. 'Effects of attachments on aerodynamic characteristics and vortex-induced vibration of twin-box girder'. In: *Journal of Fluids and Structures* 77 (2018), pp. 115–133. ISSN: 0889-9746. DOI: https://doi.org/10.1016/j.jfluidstructs.2017.12.005.

[48] B. Siedziako et al. 'An enhanced forced vibration rig for wind tunnel testing of bridge deck section models in arbitrary motion'. In: *Journal of Wind Engineering and Industrial Aerodynamics* 164 (2017), pp. 152–163. ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2017.02.011. URL: https://www.sciencedirect.com/science/article/pii/S0167610516307498.

[49] Scanivalve. *Mps4264 hardware software manual*. Scanivalve. Feb. 2022. URL: https://scanivalve.com/products/pressure-measurement/miniature-ethernet-pressure-scanners/mps4264/.

[50] Turbuelnt Flow Instruments. *Cobra Probe*. Turbuelnt Flow Instruments. n.d. URL: https://www.turbulentflow.com.au/Products/CobraProbe/CobraProbe.php.

[51] Q. Zhou et al. 'Investigation on spanwise coherence of buffeting forces acting on bridges with bluff body decks'. In: *Wind and Structures* 30.2 (Feb. 2020), pp. 181–198.

[52] J. Xia et al. 'Span-wise coherence of fluctuating forces on twin bridge decks and the turbulence effect'. In: *Advances in Structural Engineering* 22.15 (2019), pp. 3207–3221. DOI: 10.1177/1369433219859467.

Appendix A

Tube to Scanner Channel

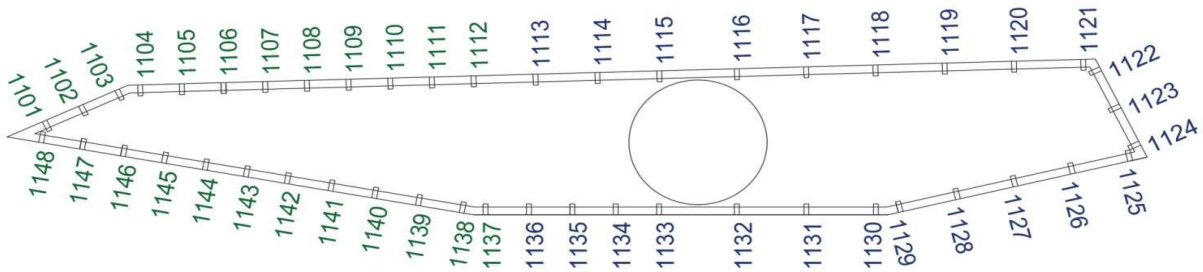
This appendix contains an overview of which pressure tube is connected to which channel and the associated scanner.

UPSTREAM BOX

SCANNER 1 - 179							
Channel	**Tube nr.**	**Channel**	**Tube nr.**	**Channel**	**Tube nr.**	**Channel**	**Tube nr.**
1	1101	2	1102	33	1301	34	1302
3	1103	4	1104	35	1303	36	1304
5	1105	6	1106	37	1313	38	1314
7	1107	8	1108	39	1315	40	1316
9	1109	10	1110	41	1401	42	1402
11	1111	12	1112	43	1403	44	1404
13	1137	14	1138	45	1413	46	1414
15	1139	16	1140	47	1415	48	1416
17	1141	18	1142	49	1501	50	1502
19	1143	20	1144	51	1503	52	1504
21	1145	22	1146	53	1513	54	1514
23	1147	24	1148	55	1515	56	1516
25	1201	26	1202	57	1601	58	1602
27	1203	28	1204	59	1603	60	1604
29	1213	30	1214	61	1613	62	1614
31	1215	32	1216	63	1615	64	1616

SCANNER 2 - 180							
Channel	**Tube nr.**	**Channel**	**Tube nr.**	**Channel**	**Tube nr.**	**Channel**	**Tube nr.**
1	1113	2	1114	33	1305	34	1306
3	1115	4	1116	35	1307	36	1308
5	1117	6	1118	37	1309	38	1310
7	1119	8	1120	39	1311	40	1312
9	1121	10	1122	41	1405	42	1406
11	1123	12	1124	43	1407	44	1408
13	1125	14	1126	45	1409	46	1410
15	1127	16	1128	47	1411	48	1412
17	1129	18	1130	49	1505	50	1506
19	1131	20	1132	51	1507	52	1508
21	1133	22	1134	53	1509	54	1510
23	1135	24	1136	55	1511	56	1512
25	1205	26	1206	57	1605	58	1606
27	1207	28	1208	59	1607	60	1608
29	1209	30	1210	61	1609	62	1610
31	1211	32	1212	63	1611	64	1612

Example, numbering line 1 (upstream):

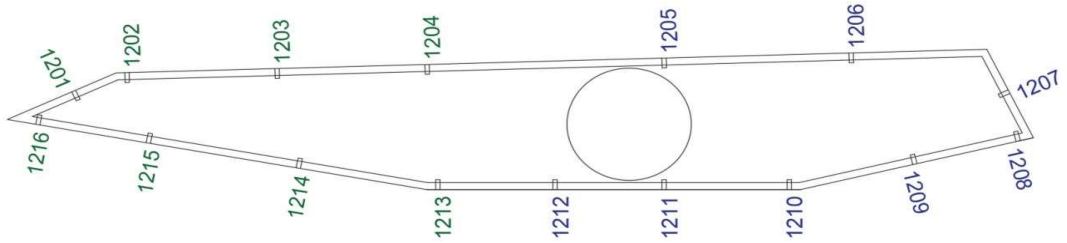


DOWNSTREAM BOX

SCANNER 3 - 181							
Channel	**Tube nr.**	**Channel**	**Tube nr.**	**Channel**	**Tube nr.**	**Channel**	**Tube nr.**
1	2113	2	2114	33	2305	34	2306
3	2115	4	2116	35	2307	36	2308
5	2117	6	2118	37	2309	38	2310
7	2119	8	2120	39	2311	40	2312
9	2121	10	2122	41	2405	42	2406
11	2123	12	2124	43	2407	44	2408
13	2125	14	2126	45	2409	46	2410
15	2127	16	2128	47	2411	48	2412
17	2129	18	2130	49	2505	50	2506
19	2131	20	2132	51	2507	52	2508
21	2133	22	2134	53	2509	54	2510
23	2135	24	2136	55	2511	56	2512
25	2205	26	2206	57	2605	58	2606
27	2207	28	2208	59	2607	60	2608
29	2209	30	2210	61	2609	62	2610
31	2211	32	2212	63	2611	64	2612

SCANNER 4 - 182							
Channel	**Tube nr.**	**Channel**	**Tube nr.**	**Channel**	**Tube nr.**	**Channel**	**Tube nr.**
1	2101	2	2102	33	2301	34	2302
3	2103	4	2104	35	2303	36	2304
5	2105	6	2106	37	2313	38	2314
7	2107	8	2108	39	2315	40	2316
9	2109	10	2110	41	2401	42	2402
11	2111	12	2112	43	2403	44	2404
13	2137	14	2138	45	2413	46	2414
15	2139	16	2140	47	2415	48	2416
17	2141	18	2142	49	2501	50	2502
19	2143	20	2144	51	2503	52	2504
21	2145	22	2146	53	2513	54	2514
23	2147	24	2148	55	2515	56	2516
25	2201	26	2202	57	2601	58	2602
27	2203	28	2204	59	2603	60	2604
29	2213	30	2214	61	2613	62	2614
31	2215	32	2216	63	2615	64	2616

Example, numbering line 2 (upstream):



Appendix B

Python Script for Estimation of Pressure and Load Distribution

This appendix contains the Python scripts for estimation of the pressure distribution and loads with the piece-wise load method and the interpolated load method. In addition, several functions used in both methods for the calculation is attached.

B.1 The Piece-wise Load Method

```python
# -*- coding: utf-8 -*-
"""
Created on Tue Mar  1 13:10:41 2022

@author: marth

"""
# Piece-wise load method - Calculate pressure to loads + plots

from CoordinatesAndAreaFunc import CoordinatesAndArea
import presscan
from SortPressure import SortPressure
from Surface16Taps import Area16Taps

from matplotlib import pyplot as plt
import numpy as np
from numpy.linalg import norm
import matplotlib.cm as cmx
import matplotlib as mpl

#%% Importing pressure
folder = 'C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/ScannerFiler/ScannerFiler/05_03_2022_12_59_54_PM'

filename = '191_30_90_179.dat'
(t_frame, t_trig, pres1, temp, scan_data) = presscan.readdatfile(filename,folder)

filename = '191_30_90_180.dat'
(t_frame, t_trig, pres2, temp, scan_data) = presscan.readdatfile(filename,folder)

filename = '191_30_90_181.dat'
(t_frame, t_trig, pres3, temp, scan_data) = presscan.readdatfile(filename,folder)

filename = '191_30_90_182.dat'
(t_frame, t_trig, pres4, temp, scan_data) = presscan.readdatfile(filename,folder)

#%% Channel 3 on scanner 2 and 3 --> mean value of neighbouring tubes
for i in range(len(pres2)):
    pres2[i,2] = np.mean([pres2[i,1], pres2[i,3]])
    pres3[i,2] = np.mean([pres3[i,1], pres3[i,3]])


#Remove first and last 10 sec --> 200 per sec --> 2000
pres1 = pres1[2000:58000,:]
pres2 = pres2[2000:58000,:]
pres3 = pres3[2000:58000,:]
pres4 = pres4[2000:58000,:]

#%% Sort pressure after correlation line

Pres_up11, Pres_up21, Pres_up31, Pres_up41, Pres_up51, Pres_up61 = SortPressure(pres1, scanner = 1)
Pres_up12, Pres_up22, Pres_up32, Pres_up42, Pres_up52, Pres_up62 = SortPressure(pres2, scanner = 2)

Pres_up1 = Pres_up11 + Pres_up12 #Correlation line 1, upstream-box
Pres_up2 = Pres_up21 + Pres_up22
Pres_up3 = Pres_up31 + Pres_up32
Pres_up4 = Pres_up41 + Pres_up42
Pres_up5 = Pres_up51 + Pres_up52
Pres_up6 = Pres_up61 + Pres_up62

Pres_down11, Pres_down21, Pres_down31, Pres_down41, Pres_down51, Pres_down61 = SortPressure(pres3, scanner = 3)
Pres_down12, Pres_down22, Pres_down32, Pres_down42, Pres_down52, Pres_down62 = SortPressure(pres4, scanner = 4)

Pres_down1 = Pres_down11 + Pres_down12 #Correlation line 1, downstream-box
Pres_down2 = Pres_down21 + Pres_down22
Pres_down3 = Pres_down31 + Pres_down32
Pres_down4 = Pres_down41 + Pres_down42
Pres_down5 = Pres_down51 + Pres_down52
Pres_down6 = Pres_down61 + Pres_down62

#%%Import coordinates

deg = 0

(x_taps_up, y_taps_up, x_taps_down, y_taps_down, x_coord_up, y_coord_up, x_coord_down,y_coord_down, x_taps_up_16,
 y_taps_up_16, x_taps_down_16, y_taps_down_16, x_coord_up_16, y_coord_up_16, x_coord_down_16, y_coord_down_16,
 Surface, A, angle_up, angle_down) = CoordinatesAndArea(deg)
```

```
#%% Normal vectors on the surface of the cross-section

#Upstream box
v_up = np.zeros((7,2))

v_up[0,:] = [x_coord_up[0],y_coord_up[0]]
v_up[1,:] = [x_coord_up[5],y_coord_up[5]]
v_up[2,:] = [x_coord_up[25],y_coord_up[25]]
v_up[3,:] = [x_coord_up[30],y_coord_up[30]]
v_up[4,:] = [x_coord_up[37],y_coord_up[37]]
v_up[5,:] = [x_coord_up[47],y_coord_up[47]]
v_up[6,:] = [x_coord_up[59],y_coord_up[59]]

#Downstream box
v_down = np.zeros((7,2))

v_down[0,:] = [x_coord_down[0],y_coord_down[0]]
v_down[1,:] = [x_coord_down[5],y_coord_down[5]]
v_down[2,:] = [x_coord_down[25],y_coord_down[25]]
v_down[3,:] = [x_coord_down[30],y_coord_down[30]]
v_down[4,:] = [x_coord_down[37],y_coord_down[37]]
v_down[5,:] = [x_coord_down[47],y_coord_down[47]]
v_down[6,:] = [x_coord_down[59],y_coord_down[59]]

#Distance between edges

#Upstream box
Distance_up = np.zeros((6,2))

Distance_up[0,:] = abs(v_up[1,:]-v_up[0,:])
Distance_up[1,:] = abs(v_up[2,:]-v_up[1,:])
Distance_up[2,:] = abs(v_up[3,:]-v_up[2,:])
Distance_up[3,:] = abs(v_up[4,:]-v_up[3,:])
Distance_up[4,:] = abs(v_up[5,:]-v_up[4,:])
Distance_up[5,:] = abs(v_up[6,:]-v_up[5,:])

#Downstream box
Distance_down = np.zeros((6,2))

Distance_down[0,:] = abs(v_down[1,:]-v_down[0,:])
Distance_down[1,:] = abs(v_down[2,:]-v_down[1,:])
Distance_down[2,:] = abs(v_down[3,:]-v_down[2,:])
Distance_down[3,:] = abs(v_down[4,:]-v_down[3,:])
Distance_down[4,:] = abs(v_down[5,:]-v_down[4,:])
Distance_down[5,:] = abs(v_down[6,:]-v_down[5,:])

#Normal vectors

#Upstream box
nv_up_1 = Distance_up[0,:]/norm(Distance_up[0,:])
nv_up_2 = Distance_up[1,:]/norm(Distance_up[1,:])
nv_up_3 = Distance_up[2,:]/norm(Distance_up[2,:])
nv_up_4 = Distance_up[3,:]/norm(Distance_up[3,:])
nv_up_5 = Distance_up[4,:]/norm(Distance_up[4,:])
nv_up_6 = Distance_up[5,:]/norm(Distance_up[5,:])

#Downstream box
nv_down_1 = Distance_down[0,:]/norm(Distance_down[0,:])
nv_down_2 = Distance_down[1,:]/norm(Distance_down[1,:])
nv_down_3 = Distance_down[2,:]/norm(Distance_down[2,:])
nv_down_4 = Distance_down[3,:]/norm(Distance_down[3,:])
nv_down_5 = Distance_down[4,:]/norm(Distance_down[4,:])
nv_down_6 = Distance_down[5,:]/norm(Distance_down[5,:])


#%% Line 1 - Decompose the pressure

#Upstream box
Pres_up_x = np.zeros((len(pres1), 48))
Pres_up_y = np.zeros((len(pres1),48))

#Side 1 - 3 taps
Pres_up_x[:,0:3] = Pres_up1[:,0:3]*nv_up_1[1]
Pres_up_y[:,0:3] = Pres_up1[:,0:3]*-(nv_up_1[0])
```

```
#Side 2 - 18 taps
Pres_up_x[:,3:21] = Pres_up1[:,3:21]*nv_up_2[1]
Pres_up_y[:,3:21] = Pres_up1[:,3:21]*-nv_up_2[0]

#Side 3 - 3 taps
Pres_up_x[:,21:24] = Pres_up1[:,21:24]*-(nv_up_3[1])
Pres_up_y[:,21:24] = Pres_up1[:,21:24]*-(nv_up_3[0])

#Side 4 - 5 taps
Pres_up_x[:,24:29] = Pres_up1[:,24:29]*-nv_up_4[1]
Pres_up_y[:,24:29] = Pres_up1[:,24:29]*nv_up_4[0]

#Side 5 - 8 taps
Pres_up_x[:,29:37] = Pres_up1[:,29:37]*nv_up_5[1]
Pres_up_y[:,29:37] = Pres_up1[:,29:37]*nv_up_5[0]

#Side 6 - 9 taps
Pres_up_x[:,37:48] = Pres_up1[:,37:48]*nv_up_6[1]
Pres_up_y[:,37:48] = Pres_up1[:,37:48]*nv_up_6[0]


#Downstream box
Pres_down_x = np.zeros((len(pres1),48))
Pres_down_y = np.zeros((len(pres1),48))

#Side 7 - 3 taps
Pres_down_x[:,0:3] = Pres_down1[:,0:3]*-nv_down_1[1]
Pres_down_y[:,0:3] = Pres_down1[:,0:3]*-nv_down_1[0]

#Side 8 - 18 taps
Pres_down_x[:,3:21] = Pres_down1[:,3:21]*-nv_down_2[1]
Pres_down_y[:,3:21] = Pres_down1[:,3:21]*-nv_down_2[0]

#Side 9 - 3 taps
Pres_down_x[:,21:24] = Pres_down1[:,21:24]*nv_down_3[1]
Pres_down_y[:,21:24] = Pres_down1[:,21:24]*-nv_down_3[0]

#Side 10 - 5 taps
Pres_down_x[:,24:29] = Pres_down1[:,24:29]*nv_down_4[1]
Pres_down_y[:,24:29] = Pres_down1[:,24:29]*nv_down_4[0]

#Side 11 - 8 taps
Pres_down_x[:,29:37] = Pres_down1[:,29:37]*nv_down_5[1]
Pres_down_y[:,29:37] = Pres_down1[:,29:37]*nv_down_5[0]

#Side 12 - 9 taps
Pres_down_x[:,37:48] = Pres_down1[:,37:48]*-nv_down_6[1]
Pres_down_y[:,37:48] = Pres_down1[:,37:48]*nv_down_6[0]

#%% Line 1 - Plotting pressure
file = folder.split('C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/ScannerFiler/ScannerFiler/')
file = file[1]
savename1 = file + '_PointLoad_Line1'

MeanPres_up_x = Pres_up_x.mean(0)
MeanPres_up_y = Pres_up_y.mean(0)
MeanPres_down_x = Pres_down_x.mean(0)
MeanPres_down_y = Pres_down_y.mean(0)

xnew_up = x_taps_up + MeanPres_up_x
ynew_up = y_taps_up + MeanPres_up_y
xnew_down = x_taps_down + MeanPres_down_x
ynew_down = y_taps_down + MeanPres_down_y

xy = np.zeros((2*48,2))
new_xy = np.zeros((2*48,2))

for i in range(0,48):
    xy[i] = [x_taps_up[i],y_taps_up[i]]
    xy[i+48] = [x_taps_down[i], y_taps_down[i]]


for i in range(0,48):
    new_xy[i] = [xnew_up[i],ynew_up[i]]
    new_xy[i+48] = [xnew_down[i], ynew_down[i]]
```

```
dx = new_xy[:,0] - xy[:,0]
dy = new_xy[:,1] - xy[:,1]


fig = plt.figure(figsize = (12, 4), dpi = 100)
plt.plot(x_coord_up, y_coord_up, color = 'black', linewidth = 1.0)
plt.plot(x_coord_down, y_coord_down,color = 'black', linewidth = 1.0)
plt.axis('equal')

ax = fig.add_subplot(111)
cmap = plt.get_cmap('jet')

#Normalizer
D = np.sqrt(dx**2 + dy**2)
norm = mpl.colors.Normalize(0, max(D))
scalarMap = cmx.ScalarMappable(norm=norm,cmap=cmap)

for i in range(2*48):
    colorVal = scalarMap.to_rgba(D[i])
    ax.annotate("", xy = (new_xy[i,0],new_xy[i,1]), xycoords='data',
                xytext=(xy[i,0], xy[i,1]), textcoords='data',
                arrowprops=dict(arrowstyle = "-|>, head_width = 0.1, head_length = 0.2", connectionstyle="arc3",
                color=colorVal, linewidth = 1.5))

#Creating ScalarMappable
sm = plt.cm.ScalarMappable(cmap = cmap, norm = norm)
sm.set_array([])
ticks = np.arange(0,max(D),5)

clb = plt.colorbar(sm, ticks = ticks, pad = 0.02)
clb.set_label('Pressure [Pa]', labelpad = 10)
plt.title('Point Pressure', fontsize = 15)
plt.xlabel('x')
plt.ylabel('z')
plt.savefig(savename1 + '.png', bbox_inches='tight')
plt.show()


#%% Line 1 - Pressure to loads
width = 1 #m
Area = A*0.001*width #m2

Load_up_x = np.zeros((len(Pres_up1[:,0]),len(Pres_up1[0,:])))
Load_up_y = np.zeros((len(Pres_up1[:,0]),len(Pres_up1[0,:])))
Load_down_x =  np.zeros((len(Pres_up1[:,0]),len(Pres_up1[0,:])))
Load_down_y = np.zeros((len(Pres_up1[:,0]),len(Pres_up1[0,:])))

for i in range(48):
    #Upstream
    Load_up_x[:,i] = Pres_up_x[:,i]*Area[i]
    Load_up_y[:,i] = Pres_up_y[:,i]*Area[i]
    #Downstream
    Load_down_x[:,i] = Pres_down_x[:,i]*Area[i]
    Load_down_y[:,i] = Pres_down_y[:,i]*Area[i]


#%% Plotting the loads on correlation line 1

scale = 100

MeanLoad_up_x = Load_up_x.mean(0)
MeanLoad_up_y = Load_up_y.mean(0)
MeanLoad_down_x = Load_down_x.mean(0)
MeanLoad_down_y = Load_down_y.mean(0)

xnew_up = x_taps_up + MeanLoad_up_x*scale
ynew_up = y_taps_up + MeanLoad_up_y*scale
xnew_down = x_taps_down + MeanLoad_down_x*scale
ynew_down = y_taps_down + MeanLoad_down_y*scale

xy = np.zeros((2*48,2))
new_xy = np.zeros((2*48,2))

for i in range(0,48):
    xy[i] = [x_taps_up[i],y_taps_up[i]]
```

```
    xy[i+48] = [x_taps_down[i], y_taps_down[i]]


for i in range(0,48):
    new_xy[i] = [xnew_up[i],ynew_up[i]]
    new_xy[i+48] = [xnew_down[i], ynew_down[i]]


dx = new_xy[:,0] - xy[:,0]
dy = new_xy[:,1] - xy[:,1]


fig = plt.figure(figsize = (12, 4), dpi = 100)
plt.plot(x_coord_up, y_coord_up, color = 'black', linewidth = 1.0)
plt.plot(x_coord_down, y_coord_down,color = 'black', linewidth = 1.0)
plt.axis('equal')

ax = fig.add_subplot(111)
cmap = plt.get_cmap('jet') #viridis

#Normalizer
D = np.sqrt(dx**2 + dy**2)

norm = mpl.colors.Normalize(0, max(D))
scalarMap = cmx.ScalarMappable(norm=norm,cmap=cmap)


for i in range(2*48):
    colorVal = scalarMap.to_rgba(D[i])
    ax.annotate("", xy = (new_xy[i,0],new_xy[i,1]), xycoords='data',
                xytext=(xy[i,0], xy[i,1]), textcoords='data',
                arrowprops=dict(arrowstyle = "-|>, head_width = 0.1, head_length = 0.2", connectionstyle="arc3",
                color=colorVal, linewidth = 1.5))

#Creating ScalarMappable
sm = plt.cm.ScalarMappable(cmap = cmap, norm = norm)
sm.set_array([])
ticks = np.arange(0,max(D),10)

clb = plt.colorbar(sm, ticks = ticks, pad = 0.02)
clb.set_label('Force [N] $\cdot 10^{-2}$', labelpad = 10)
plt.title('Point Load', fontsize = 15)
plt.xlabel('x')
plt.ylabel('z')
plt.savefig(savename1 + 'PointLoad.png', bbox_inches='tight')
plt.show()

#%% Line 1 - Transforming pressure to moment

dist_up_x = abs(x_taps_up*10**(-3))
dist_up_y = abs(y_taps_up*10**(-3))
dist_down_x = abs(x_taps_down*10**(-3))
dist_down_y = abs(y_taps_down*10**(-3))

center = [0,0]
Moment_up = np.zeros((len(Load_up_x),48))
Moment_down = np.zeros((len(Load_down_x),48))

#Upstream box
for i in range(len(x_taps_up)):
    if x_taps_up[i] <= center[0] and y_taps_up[i] >= center[1]: #Tube nr. 1-23
        if 0 <= i <= 20: #Tube nr. 1-21
            Moment_up[:,i] = Load_up_x[:,i]*dist_up_y[i] + Load_up_y[:,i]*dist_up_x[i]
        else: #Tube nr. 22-23
            Moment_up[:,i] = Load_up_x[:,i]*dist_up_y[i] + Load_up_y[:,i]*dist_up_x[i]
    else: #Tube nr. 24-48
        if i == 23:
            Moment_up[:,i] = - Load_up_x[:,i]*dist_up_y[i] + Load_up_y[:,i]*dist_up_x[i]
        elif 24 <= i <= 28: #Tube nr. 25-29
            Moment_up[:,i] = - Load_up_x[:,i]*dist_up_y[i] + Load_up_y[:,i]*dist_up_x[i]
        elif 29 <= i <= 36: #Tube nr. 30-37
            Moment_up[:,i] = Load_up_y[:,i]*dist_up_x[i]
        else: #Tube nr. 38-48
            Moment_up[:,i] = - Load_up_x[:,i]*dist_up_y[i] + Load_up_y[:,i]*dist_up_x[i]
```

```
#Downstream box
for i in range(len(x_taps_down)):
    if x_taps_down[i] <= center[0] and y_taps_down[i] >= center[1]: #Tube nr. 1-23
        if 0 <= i <= 20: #Tube nr. 1-21
            Moment_down[:,i] = Load_down_x[:,i]*dist_down_y[i] - Load_down_y[:,i]*dist_down_x[i]
        else: #Tube nr. 22-23
            Moment_down[:,i] = Load_down_x[:,i]*dist_down_y[i] - Load_down_y[:,i]*dist_down_x[i]
    else: #Tube nr. 24-48
        if i == 23:
            Moment_down[:,i] = - Load_down_x[:,i]*dist_down_y[i] - Load_down_y[:,i]*dist_down_x[i]
        elif 24 <= i <= 28: #Tube nr. 25-29
            Moment_down[:,i] = - Load_down_x[:,i]*dist_down_y[i] - Load_down_y[:,i]*dist_down_x[i]
        elif 29 <= i <= 36: #Tube nr. 30-37
            Moment_down[:,i] = - Load_down_y[:,i]*dist_down_x[i]
        else: #Tube nr. 38-48
            Moment_down[:,i] = - Load_down_x[:,i]*dist_down_y[i] - Load_down_y[:,i]*dist_down_x[i]



#Save loads and moment from correlation line 1
file = folder.split('C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/ScannerFiler/ScannerFiler/')
file = file[1]
savename1 = file + '_PointLoad_Line1'
savename1 = savename1 + '.npz'

np.savez_compressed(savename1,Load_up_x = Load_up_x, Load_up_y = Load_up_y,Load_down_x = Load_down_x,
                    Load_down_y = Load_down_y, Moment_up = Moment_up, Moment_down = Moment_down)


#%% Correlation line 2-6
file = folder.split('C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/ScannerFiler/ScannerFiler/')
file = file[1]
savename2 = file

#Decompose the pressure
Pres_up = np.zeros((len(pres1[:,0]),16,5))
Pres_down = np.zeros((len(pres1[:,0]),16,5))

Pres_up[:,:,0] = Pres_up2
Pres_up[:,:,1] = Pres_up3
Pres_up[:,:,2] = Pres_up4
Pres_up[:,:,3] = Pres_up5
Pres_up[:,:,4] = Pres_up6

Pres_down[:,:,0] = Pres_down2
Pres_down[:,:,1] = Pres_down3
Pres_down[:,:,2] = Pres_down4
Pres_down[:,:,3] = Pres_down5
Pres_down[:,:,4] = Pres_down6


Pres_up_x_cl = np.zeros((len(pres1[:,0]),16,5))
Pres_up_y_cl = np.zeros((len(pres1[:,0]),16,5))

Pres_down_x_cl = np.zeros((len(pres1[:,0]),16,5))
Pres_down_y_cl = np.zeros((len(pres1[:,0]),16,5))


for i in range(5):

    # Upstream box
    # Side 1

    #Side 1 - 1 tap
    Pres_up_x_cl[:,0,i] = Pres_up[:,0,i]*nv_up_1[1]
    Pres_up_y_cl[:,0,i] = Pres_up[:,0,i]*-nv_up_1[0]

    #Side 2 - 5 taps
    Pres_up_x_cl[:,1:6,i] = Pres_up[:,1:6,i]*nv_up_2[1]
    Pres_up_y_cl[:,1:6,i] = Pres_up[:,1:6,i]*-nv_up_2[0]

    #Side 3 - 1 tap
    Pres_up_x_cl[:,6,i] = Pres_up[:,6,i]*-nv_up_3[1]
    Pres_up_y_cl[:,6,i] = Pres_up[:,6,i]*-nv_up_3[0]

    #Side 4 - 2 taps
```

```
    Pres_up_x_cl[:,7:9,i] = Pres_up[:,7:9,i]*-nv_up_4[1]
    Pres_up_y_cl[:,7:9,i] = Pres_up[:,7:9,i]*nv_up_4[0]

    #Side 5 - 4 taps
    Pres_up_x_cl[:,9:13,i] = Pres_up[:,9:13,i]*nv_up_5[1]
    Pres_up_y_cl[:,9:13,i] = Pres_up[:,9:13,i]*nv_up_5[0]

    #Side 6 - 3 taps
    Pres_up_x_cl[:,13:16,i] = Pres_up[:,13:16,i]*nv_up_6[1]
    Pres_up_y_cl[:,13:16,i] = Pres_up[:,13:16,i]*nv_up_6[0]


    #Downstream box
    #Side 7 - 1 tap
    Pres_down_x_cl[:,0,i] = Pres_down[:,0,i]*-nv_down_1[1]
    Pres_down_y_cl[:,0,i] = Pres_down[:,0,i]*-nv_down_1[0]

    #Side 8 - 5 taps
    Pres_down_x_cl[:,1:6,i] = Pres_down[:,1:6,i]*-nv_down_2[1]
    Pres_down_y_cl[:,1:6,i] = Pres_down[:,1:6,i]*-nv_down_2[0]

    #Side 9 - 1 tap
    Pres_down_x_cl[:,6,i] = Pres_down[:,6,i]*nv_down_3[1]
    Pres_down_y_cl[:,6,i] = Pres_down[:,6,i]*-nv_down_3[0]

    #Side 10 - 2 taps
    Pres_down_x_cl[:,7:9,i] = Pres_down[:,7:9,i]*nv_down_4[1]
    Pres_down_y_cl[:,7:9,i] = Pres_down[:,7:9,i]*nv_down_4[0]

    #Side 11 - 4 taps
    Pres_down_x_cl[:,9:13,i] = Pres_down[:,9:13,i]*nv_down_5[1]
    Pres_down_y_cl[:,9:13,i] = Pres_down[:,9:13,i]*nv_down_5[0]

    #Side 12 - 3 taps
    Pres_down_x_cl[:,13:16,i] = Pres_down[:,13:16,i]*-nv_down_6[1]
    Pres_down_y_cl[:,13:16,i] = Pres_down[:,13:16,i]*nv_down_6[0]


xy_16 = np.zeros((2*16,2,5))
new_xy_16 = np.zeros((2*16,2,5))

MeanPres_up_x_cl = np.zeros((16,5))
MeanPres_up_y_cl = np.zeros((16,5))
MeanPres_down_x_cl = np.zeros((16,5))
MeanPres_down_y_cl = np.zeros((16,5))

for i in range(5):
    for j in range(16):
        MeanPres_up_x_cl[j,i] = np.mean(Pres_up_x_cl[:,j,i])
        MeanPres_up_y_cl[j,i] = np.mean(Pres_up_y_cl[:,j,i])
        MeanPres_down_x_cl[j,i] = np.mean(Pres_down_x_cl[:,j,i])
        MeanPres_down_y_cl[j,i] = np.mean(Pres_down_y_cl[:,j,i])

#Plotting the pressure on correlation lines 2-6

xnew_up = np.zeros((16,5))
ynew_up = np.zeros((16,5))
xnew_down = np.zeros((16,5))
ynew_down = np.zeros((16,5))

for i in range(5):
    xnew_up[:,i] = x_taps_up_16 + MeanPres_up_x_cl[:,i]
    ynew_up[:,i] = y_taps_up_16 + MeanPres_up_y_cl[:,i]
    xnew_down[:,i] = x_taps_down_16 + MeanPres_down_x_cl[:,i]
    ynew_down[:,i] = y_taps_down_16 + MeanPres_down_y_cl[:,i]


for i in range(5):
    for j in range(0,16):

        xy_16[j,:,i] = [x_taps_up_16[j],y_taps_up_16[j]]
        xy_16[j+16,:,i] = [x_taps_down_16[j], y_taps_down_16[j]]

        new_xy_16[j,:,i] = [xnew_up[j,i],ynew_up[j,i]]
        new_xy_16[j+16,:,i] = [xnew_down[j,i], ynew_down[j,i]]
```

```
dx_16 = np.zeros((2*16,5))
dy_16 = np.zeros((2*16,5))

for i in range(5):
    dx_16[:,i] = xy_16[:,0,i] - new_xy_16[:,0,i]
    dy_16[:,i] = xy_16[:,1,i] - new_xy_16[:,1,i]


#Plotting the cross-section
fig = plt.figure(figsize = (12, 4), dpi = 100)
plt.plot(x_coord_up, y_coord_up, color = 'black', linewidth = 1.0)
plt.plot(x_coord_down, y_coord_down,color = 'black', linewidth = 1.0)
plt.axis('equal')


ax = fig.add_subplot(111)
cmap = plt.get_cmap('jet')

#Normalizer
D = np.sqrt(dx_16**2 + dy_16**2)

norm = mpl.colors.Normalize(0, max(D[:,0]))
scalarMap = cmx.ScalarMappable(norm=norm,cmap=cmap)


#Correlation line 2 --> index = 0
for i in range(2*16):
    colorVal = scalarMap.to_rgba(D[i,0])
    ax.annotate("", xy = (new_xy_16[i,0,0],new_xy_16[i,1,0]), xycoords='data',
                xytext=(xy_16[i,0,0], xy_16[i,1,0]), textcoords='data',
                arrowprops=dict(arrowstyle = "-|>, head_width = 0.1, head_length = 0.2", connectionstyle="arc3",
                color=colorVal, linewidth = 1.5))

#Creating ScalarMappable
sm = plt.cm.ScalarMappable(cmap = cmap, norm = norm)
sm.set_array([])
ticks = np.arange(0,max(D[:,0]),10)

clb = plt.colorbar(sm, ticks = ticks, pad = 0.02)
clb.set_label('Pressure [Pa]', labelpad = 10)
plt.title('Point Pressure', fontsize = 15)
plt.xlabel('x')
plt.ylabel('z')
plt.savefig(savename2 +'_Line2.png', bbox_inches='tight')

#%% Transforming pressure to loads and moment, correlation line 2-6

A16 = Area16Taps()
Width = 1 #m
Area16 = A16*Width*10**(-3) #m2

Load_up_x_cl = np.zeros((len(pres1[:,0]),16,5))
Load_up_y_cl = np.zeros((len(pres1[:,0]),16,5))
Load_down_x_cl =  np.zeros((len(pres1[:,0]),16,5))
Load_down_y_cl = np.zeros((len(pres1[:,0]),16,5))

for i in range(5):
    for j in range(16):

        #Upstream box
        Load_up_x_cl[:,j,i] = Pres_up_x_cl[:,j,i]*Area16[j]
        Load_up_y_cl[:,j,i] = Pres_up_y_cl[:,j,i]*Area16[j]

        #Downstream box
        Load_down_x_cl[:,j,i] = Pres_down_x_cl[:,j,i]*Area16[j]
        Load_down_y_cl[:,j,i] = Pres_down_y_cl[:,j,i]*Area16[j]

#Transforming to moments
dist_up_x = abs(x_taps_up_16*10**(-3))
dist_up_y = abs(y_taps_up_16*10**(-3))
dist_down_x = abs(x_taps_down_16*10**(-3))
dist_down_y = abs(y_taps_down_16*10**(-3))

center = [0,0]
Moment_up_cl = np.zeros((len(Load_up_x),16,5))
```

```
Moment_down_cl = np.zeros((len(Load_down_x),16,5))

#Upstream box
for i in range(5):
    for j in range(16):
        if x_taps_up_16[i] <= center[0] and y_taps_up_16[i] >= center[1]: #Tube nr.1-7
            if 0 <= i <= 5: #Tube nr.1-6
                Moment_up_cl[:,j,i] = Load_up_x_cl[:,j,i]*dist_up_y[i] + Load_up_y_cl[:,j,i]*dist_up_x[i]
            else: #Tube nr.7
                Moment_up_cl[:,j,i] =  Load_up_x_cl[:,j,i]*dist_up_y[i] + Load_up_y_cl[:,j,i]*dist_up_x[i]

        else: #Tube nr. 8-16
            if i == 7 or i == 8: #Tube nr. 8-9
                Moment_up_cl[:,j,i] = - Load_up_x_cl[:,j,i]*dist_up_y[i] + Load_up_y_cl[:,j,i]*dist_up_x[i]
            elif 9 <= i <= 12: #Tube nr. 10-13
                Moment_up_cl[:,j,i] = Load_up_y_cl[:,j,i]*dist_up_x[i]
            else: #Tube nr. 14-16
                Moment_up_cl[:,j,i] = - Load_up_x_cl[:,j,i]*dist_up_y[i] + Load_up_y_cl[:,j,i]*dist_up_x[i]

#Downstream box
for i in range(5):
    for j in range(16):
        if x_taps_down_16[i] <= center[0] and y_taps_down_16[i] >= center[1]: #Tube nr.1-7
            if 0 <= i <= 5: #Tube nr.1-6
                Moment_down_cl[:,j,i] =  Load_down_x_cl[:,j,i]*dist_down_y[i] - Load_down_y_cl[:,j,i]*dist_down_x[i]
            else: #Tube nr.7
                Moment_down_cl[:,j,i] = Load_down_x_cl[:,j,i]*dist_down_y[i] - Load_down_y_cl[:,j,i]*dist_down_x[i]

        else: #Tube nr. 8-16
            if i == 7 or i == 8: #Tube nr. 8-9
                Moment_down_cl[:,j,i] = - Load_down_x_cl[:,j,i]*dist_down_y[i] - Load_down_y_cl[:,j,i]*dist_down_x[i]
            elif 9 <= i <= 12: #Tube nr. 10-13
                Moment_down_cl[:,j,i] = - Load_down_y_cl[:,j,i]*dist_down_x[i]
            else: #Tube nr. 14-16
                Moment_down_cl[:,j,i] = - Load_down_x_cl[:,j,i]*dist_down_y[i] - Load_down_y_cl[:,j,i]*dist_down_x[i]


#Save loads and moments
savename2 = savename2 + '.npz'
np.savez_compressed(savename2, Load_up_x = Load_up_x_cl, Load_up_y = Load_up_y_cl,Load_down_x = Load_down_x_cl,
                    Load_down_y = Load_down_y_cl, Moment_up = Moment_up_cl, Moment_down = Moment_down_cl )
```

B.2 The Interpolated Load Method

```python
# -*- coding: utf-8 -*-
"""
Created on Wed Mar  9 14:23:18 2022

@author: marth
"""

from CoordinatesAndAreaFunc import CoordinatesAndArea
import presscan
from CoordinatesEqualSpacing import CoordinatesEqualSpacing
from SortPressure import SortPressure

import numpy as np
from numpy.linalg import norm
from scipy.interpolate import CubicSpline
from matplotlib import pyplot as plt
import matplotlib.cm as cmx
import matplotlib as mpl

#%% Importing pressure and sorting it
folder = 'C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/ScannerFiler/ScannerFiler/05_02_2022_4_08_09_PM'

filename = '191_30_90_179.dat'
(t_frame, t_trig, pres1, temp, scan_data) = presscan.readdatfile(filename,folder)

filename = '191_30_90_180.dat'
(t_frame, t_trig, pres2, temp, scan_data) = presscan.readdatfile(filename,folder)

filename = '191_30_90_181.dat'
(t_frame, t_trig, pres3, temp, scan_data) = presscan.readdatfile(filename,folder)

filename = '191_30_90_182.dat'
(t_frame, t_trig, pres4, temp, scan_data) = presscan.readdatfile(filename,folder)

#%% Channel 3 on scanner 2 and 3 --> mean value of neighbouring tubes

for i in range(len(pres2)):
    pres2[i,2] = np.mean([pres2[i,1], pres2[i,3]])
    pres3[i,2] = np.mean([pres3[i,1], pres3[i,3]])


#Remove first and last 10 sec --> 200 per sec --> 2000
pres1 = pres1[2000:58000,:]
pres2 = pres2[2000:58000,:]
pres3 = pres3[2000:58000,:]
pres4 = pres4[2000:58000,:]

#%% Sort pressure after correlation line

Pres_up11, Pres_up21, Pres_up31, Pres_up41, Pres_up51, Pres_up61 = SortPressure(pres1, scanner = 1)
Pres_up12, Pres_up22, Pres_up32, Pres_up42, Pres_up52, Pres_up62 = SortPressure(pres2, scanner = 2)

Pres_up1 = Pres_up11 + Pres_up12 #Correlation line 1, upstream-box
Pres_up2 = Pres_up21 + Pres_up22
Pres_up3 = Pres_up31 + Pres_up32
Pres_up4 = Pres_up41 + Pres_up42
Pres_up5 = Pres_up51 + Pres_up52
Pres_up6 = Pres_up61 + Pres_up62


Pres_down11, Pres_down21, Pres_down31, Pres_down41, Pres_down51, Pres_down61 = SortPressure(pres3, scanner = 3)
Pres_down12, Pres_down22, Pres_down32, Pres_down42, Pres_down52, Pres_down62 = SortPressure(pres4, scanner = 4)

Pres_down1 = Pres_down11 + Pres_down12 #Correlation line 1, downstream-box
Pres_down2 = Pres_down21 + Pres_down22
Pres_down3 = Pres_down31 + Pres_down32
Pres_down4 = Pres_down41 + Pres_down42
Pres_down5 = Pres_down51 + Pres_down52
Pres_down6 = Pres_down61 + Pres_down62

#%% Correlation line 1

Pres_up = np.zeros(48)
Pres_down = np.zeros(48)

for i in range(48):
    Pres_up[i] = np.mean(Pres_up1[:,i])
    Pres_down[i] = np.mean(Pres_down1[:,i])

#%%Import coordinates
deg = 0
(x_taps_up, y_taps_up, x_taps_down, y_taps_down, x_coord_up, y_coord_up, x_coord_down, y_coord_down, x_taps_up_16,
 y_taps_up_16, x_taps_down_16, y_taps_down_16, x_coord_up_16, y_coord_up_16, x_coord_down_16, y_coord_down_16,
 Surface, A, angle_up, angle_down) = CoordinatesAndArea(deg)

#%% New coordinates
N = [39, 281, 35, 78, 120, 138]
```

```
dt = 1 #Spacing, 1 mm

#Upstream box
x_start_up = [x_coord_up[0], x_coord_up[5], x_coord_up[25], x_coord_up[30], x_coord_up[37], x_coord_up[47]]
y_start_up = [y_coord_up[0], y_coord_up[5], y_coord_up[25], y_coord_up[30], y_coord_up[37], y_coord_up[47]]

#X-coordinates
(X_S1, y) = CoordinatesEqualSpacing(x_start_up[0], y_start_up[0],dt, 180 - angle_up[0], N[0])
(X_S2, y) = CoordinatesEqualSpacing(x_start_up[1], y_start_up[1],dt, 180 - angle_up[1], N[1])
(X_S3, y) = CoordinatesEqualSpacing(x_start_up[2], y_start_up[2],dt, 180 - angle_up[2], N[2])
(X_S4, y) = CoordinatesEqualSpacing(x_start_up[3], y_start_up[3],dt, angle_up[3], N[3])
(X_S5, y) = CoordinatesEqualSpacing(x_start_up[4], y_start_up[4],dt, angle_up[4], N[4])
(X_S6, y) = CoordinatesEqualSpacing(x_start_up[5], y_start_up[5],dt, angle_up[5], N[5])

X_coord_up = np.concatenate((X_S1, X_S2, X_S3, X_S4, X_S5, X_S6))

#Y-coordinates
(x, Y_S1) = CoordinatesEqualSpacing(x_start_up[0], y_start_up[0],dt, 180 + angle_up[0], N[0])
(x, Y_S2) = CoordinatesEqualSpacing(x_start_up[1], y_start_up[1],dt, 180 + angle_up[1], N[1])
(x, Y_S3) = CoordinatesEqualSpacing(x_start_up[2], y_start_up[2],dt, angle_up[2], N[2])
(x, Y_S4) = CoordinatesEqualSpacing(x_start_up[3], y_start_up[3],dt, angle_up[3], N[3])
(x, Y_S5) = CoordinatesEqualSpacing(x_start_up[4], y_start_up[4],dt, 180 + angle_up[4], N[4])
(x, Y_S6) = CoordinatesEqualSpacing(x_start_up[5], y_start_up[5],dt, 180 + angle_up[5], N[5])

Y_coord_up = np.concatenate((Y_S1, Y_S2, Y_S3, Y_S4, Y_S5, Y_S6))


#Downstream box
x_start_down = [x_coord_down[0], x_coord_down[5], x_coord_down[25], x_coord_down[30], x_coord_down[37], x_coord_down[47]]
y_start_down = [y_coord_down[0], y_coord_down[5], y_coord_down[25], y_coord_down[30], y_coord_down[37], y_coord_down[47]]

#X-coordinates
(X_S7, y) = CoordinatesEqualSpacing(x_start_down[0], y_start_down[0],dt, angle_down[0], N[0])
(X_S8, y) = CoordinatesEqualSpacing(x_start_down[1], y_start_down[1],dt, angle_down[1], N[1])
(X_S9, y) = CoordinatesEqualSpacing(x_start_down[2], y_start_down[2],dt, angle_down[2], N[2])
(X_S10, y) = CoordinatesEqualSpacing(x_start_down[3], y_start_down[3],dt, 180+angle_down[3], N[3])
(X_S11, y) = CoordinatesEqualSpacing(x_start_down[4], y_start_down[4],dt, 180+angle_down[4], N[4])
(X_S12, y) = CoordinatesEqualSpacing(x_start_down[5], y_start_down[5],dt, 180+angle_down[5], N[5])

X_coord_down = np.concatenate((X_S7, X_S8 , X_S9 , X_S10 , X_S11 , X_S12))

#Y-coordinates
(x, Y_S7) = CoordinatesEqualSpacing(x_start_down[0], y_start_down[0],dt, 180 + angle_down[0], N[0])
(x, Y_S8) = CoordinatesEqualSpacing(x_start_down[1], y_start_down[1],dt, 180 + angle_down[1], N[1])
(x, Y_S9) = CoordinatesEqualSpacing(x_start_down[2], y_start_down[2],dt, angle_down[2], N[2])
(x, Y_S10) = CoordinatesEqualSpacing(x_start_down[3], y_start_down[3],dt, angle_down[3], N[3])
(x, Y_S11) = CoordinatesEqualSpacing(x_start_down[4], y_start_down[4],dt, angle_down[4], N[4])
(x, Y_S12) = CoordinatesEqualSpacing(x_start_down[5], y_start_down[5],dt, 180  + angle_down[5], N[5])

Y_coord_down = np.concatenate((Y_S7, Y_S8, Y_S9, Y_S10, Y_S11, Y_S12))


#%% Converting point loads to distributed loads

for i in range(1,13): #Number of sides (both upstream and downstream box)

    #Upstream box
    if i == 1:  #Side 1
        S_x1 = x_coord_up[0:5]
        S_x = S_x1
        S_y1 = y_coord_up[0:5]
        S_y = S_y1
        S_taps1 = Pres_up[0:3]
        S_taps = S_taps1

        S_L = np.zeros((len(S_x)))

        for i in range(1,len(S_x)):
            S_L[i] = S_L[i-1] + np.sqrt((S_x[i]-S_x[i-1])**2 + (S_y[i]-S_y[i-1])**2)

        X_s = S_L[1:-1]
        xx_s = np.arange(S_L[0], S_L[-1], dt)
        s = CubicSpline(X_s, S_taps)
        yy_s = s(xx_s)

        P1 = yy_s
        S_L1 = S_L


    elif i == 2: #Side 2
        S_x2 = x_coord_up[5:25]
        S_x = S_x2
        S_y2 = y_coord_up[5:25]
        S_y = S_y2
        S_taps2 = Pres_up[3:21]
        S_taps = S_taps2
```

```
    S_L = np.zeros((len(S_x)))

    for i in range(1,len(S_x)):
        S_L[i] = S_L[i-1] + np.sqrt((S_x[i]-S_x[i-1])**2 + (S_y[i]-S_y[i-1])**2)

    X_s = S_L[1:-1]
    xx_s = np.arange(S_L[0], S_L[-1], dt)
    s = CubicSpline(X_s, S_taps)
    yy_s = s(xx_s)

    P2 = yy_s
    S_L2 = S_L


elif i == 3: #Side 3
    S_x3 = x_coord_up[25:30]
    S_x = S_x3
    S_y3 = y_coord_up[25:30]
    S_y = S_y3
    S_taps3 = Pres_up[21:24]
    S_taps = S_taps3

    S_L = np.zeros((len(S_x)))
    for i in range(1,len(S_x)):
        S_L[i] = S_L[i-1] + np.sqrt((S_x[i]-S_x[i-1])**2 + (S_y[i]-S_y[i-1])**2)

    X_s = S_L[1:-1]
    xx_s = np.arange(S_L[0], S_L[-1], dt)
    s = CubicSpline(X_s, S_taps)
    yy_s = s(xx_s)

    P3 = yy_s
    S_L3 = S_L

elif i == 4: #Side 4
    S_x4 = x_coord_up[30:37]
    S_x = S_x4
    S_y4 = y_coord_up[30:37]
    S_y = S_y4
    S_taps4 = Pres_up[24:29]
    S_taps = S_taps4

    S_L = np.zeros((len(S_x)))
    for i in range(1,len(S_x)):
        S_L[i] = S_L[i-1] + np.sqrt((S_x[i]-S_x[i-1])**2 + (S_y[i]-S_y[i-1])**2)

    X_s = S_L[1:-1]
    xx_s = np.arange(S_L[0], S_L[-1], dt)
    s = CubicSpline(X_s, S_taps)
    yy_s = s(xx_s)

    P4 = yy_s
    S_L4 = S_L

elif i == 5: #Side 5
    S_x5= x_coord_up[37:47]
    S_x = S_x5
    S_y5= y_coord_up[37:47]
    S_y = S_y5
    S_taps5 = Pres_up[29:37]
    S_taps = S_taps5

    S_L = np.zeros((len(S_x)))
    for i in range(1,len(S_x)):
        S_L[i] = S_L[i-1] + np.sqrt((S_x[i]-S_x[i-1])**2 + (S_y[i]-S_y[i-1])**2)

    X_s = S_L[1:-1]
    xx_s = np.arange(S_L[0], S_L[-1], dt)
    s = CubicSpline(X_s, S_taps)
    yy_s = s(xx_s)

    P5 = yy_s
    S_L5 = S_L

elif i == 6: #Side 6
    S_x6= x_coord_up[47:60]
    S_x = S_x6
    S_y6 = y_coord_up[47:60]
    S_y = S_y6
    S_taps6 = Pres_up[37:48]
    S_taps = S_taps6

    S_L = np.zeros((len(S_x)))
    for i in range(1,len(S_x)):
        S_L[i] = S_L[i-1] + np.sqrt((S_x[i]-S_x[i-1])**2 + (S_y[i]-S_y[i-1])**2)

    X_s = S_L[1:-1]
```

```
    xx_s = np.arange(S_L[0], S_L[-1], dt)
    s = CubicSpline(X_s, S_taps)
    yy_s = s(xx_s)

    P6 = yy_s
    S_L6 = S_L

#Downstream box
elif i == 7:  #Side 7
    S_x7 = x_coord_down[0:5]
    S_x = S_x7
    S_y7 = y_coord_down[0:5]
    S_y = S_y7
    S_taps7 = Pres_down[0:3]
    S_taps = S_taps7

    S_L = np.zeros((len(S_x)))
    for i in range(1,len(S_x)):
        S_L[i] = S_L[i-1] + np.sqrt((S_x[i]-S_x[i-1])**2 + (S_y[i]-S_y[i-1])**2)

    X_s = S_L[1:-1]
    xx_s = np.arange(S_L[0], S_L[-1], dt)
    s = CubicSpline(X_s, S_taps)
    yy_s = s(xx_s)

    P7 = yy_s
    S_L7 = S_L

elif i == 8: #Side 8
    S_x8 = x_coord_down[5:25]
    S_x = S_x8
    S_y8 = y_coord_down[5:25]
    S_y = S_y8
    S_taps8 = Pres_down[3:21]
    S_taps = S_taps8

    S_L = np.zeros((len(S_x)))
    for i in range(1,len(S_x)):
        S_L[i] = S_L[i-1] + np.sqrt((S_x[i]-S_x[i-1])**2 + (S_y[i]-S_y[i-1])**2)

    X_s = S_L[1:-1]
    xx_s = np.arange(S_L[0], S_L[-1], dt)
    s = CubicSpline(X_s, S_taps)
    yy_s = s(xx_s)

    P8 = yy_s
    S_L8 = S_L

elif i == 9: #Side 9
    S_x9 = x_coord_down[25:30]
    S_x = S_x9
    S_y9 = y_coord_down[25:30]
    S_y = S_y9
    S_taps9 = Pres_down[21:24]
    S_taps = S_taps9

    S_L = np.zeros((len(S_x)))
    for i in range(1,len(S_x)):
        S_L[i] = S_L[i-1] + np.sqrt((S_x[i]-S_x[i-1])**2 + (S_y[i]-S_y[i-1])**2)

    X_s = S_L[1:-1]
    xx_s = np.arange(S_L[0], S_L[-1], dt)
    s = CubicSpline(X_s, S_taps)
    yy_s = s(xx_s)

    P9 = yy_s
    S_L9 = S_L

elif i == 10: #Side 10
    S_x10 = x_coord_down[30:37]
    S_x = S_x10
    S_y10 = y_coord_down[30:37]
    S_y = S_y10
    S_taps10 = Pres_down[24:29]
    S_taps = S_taps10

    S_L = np.zeros((len(S_x)))
    for i in range(1,len(S_x)):
        S_L[i] = S_L[i-1] + np.sqrt((S_x[i]-S_x[i-1])**2 + (S_y[i]-S_y[i-1])**2)

    X_s = S_L[1:-1]
    xx_s = np.arange(S_L[0], S_L[-1], dt)
    s = CubicSpline(X_s, S_taps)
    yy_s = s(xx_s)

    P10 = yy_s
    S_L10 = S_L
```

```
    elif i == 11: #Side 11
        S_x11 = x_coord_down[37:47]
        S_x = S_x11
        S_y11 = y_coord_down[37:47]
        S_y = S_y11
        S_taps11 = Pres_down[29:37]
        S_taps = S_taps11

        S_L = np.zeros((len(S_x)))
        for i in range(1,len(S_x)):
            S_L[i] = S_L[i-1] + np.sqrt((S_x[i]-S_x[i-1])**2 + (S_y[i]-S_y[i-1])**2)

        X_s = S_L[1:-1]
        xx_s = np.arange(S_L[0], S_L[-1], dt)
        s = CubicSpline(X_s, S_taps)
        yy_s = s(xx_s)

        P11 = yy_s
        S_L11 = S_L

    elif i == 12: #Side 12
        S_x12 = x_coord_down[47:60]
        S_x = S_x12
        S_y12 = y_coord_down[47:60]
        S_y = S_y12
        S_taps12 = Pres_down[37:48]
        S_taps = S_taps12

        S_L = np.zeros((len(S_x)))
        for i in range(1,len(S_x)):
            S_L[i] = S_L[i-1] + np.sqrt((S_x[i]-S_x[i-1])**2 + (S_y[i]-S_y[i-1])**2)

        X_s = S_L[1:-1]
        xx_s = np.arange(S_L[0], S_L[-1], dt)
        s = CubicSpline(X_s, S_taps)
        yy_s = s(xx_s)

        P12 = yy_s
        S_L12 = S_L

#%% Normal vectors on the surface of the cross-section

#Edges on the cross section
#Upstream box
edge_up = np.zeros((7,2))

edge_up[0,:] = [x_coord_up[0],y_coord_up[0]]
edge_up[1,:] = [x_coord_up[5],y_coord_up[5]]
edge_up[2,:] = [x_coord_up[25],y_coord_up[25]]
edge_up[3,:] = [x_coord_up[30],y_coord_up[30]]
edge_up[4,:] = [x_coord_up[37],y_coord_up[37]]
edge_up[5,:] = [x_coord_up[47],y_coord_up[47]]
edge_up[6,:] = [x_coord_up[59],y_coord_up[59]]

#Downstream box
edge_down = np.zeros((7,2))

edge_down[0,:] = [x_coord_down[0],y_coord_down[0]]
edge_down[1,:] = [x_coord_down[5],y_coord_down[5]]
edge_down[2,:] = [x_coord_down[25],y_coord_down[25]]
edge_down[3,:] = [x_coord_down[30],y_coord_down[30]]
edge_down[4,:] = [x_coord_down[37],y_coord_down[37]]
edge_down[5,:] = [x_coord_down[47],y_coord_down[47]]
edge_down[6,:] = [x_coord_down[59],y_coord_down[59]]

#Distance between edges

#Upstream box
Distance_up = np.zeros((6,2))

Distance_up[0,:] = abs(edge_up[1,:]-edge_up[0,:])
Distance_up[1,:] = abs(edge_up[2,:]-edge_up[1,:])
Distance_up[2,:] = abs(edge_up[3,:]-edge_up[2,:])
Distance_up[3,:] = abs(edge_up[4,:]-edge_up[3,:])
Distance_up[4,:] = abs(edge_up[5,:]-edge_up[4,:])
Distance_up[5,:] = abs(edge_up[6,:]-edge_up[5,:])

#Downstream box
Distance_down = np.zeros((6,2))

Distance_down[0,:] = abs(edge_down[1,:]-edge_down[0,:])
Distance_down[1,:] = abs(edge_down[2,:]-edge_down[1,:])
Distance_down[2,:] = abs(edge_down[3,:]-edge_down[2,:])
Distance_down[3,:] = abs(edge_down[4,:]-edge_down[3,:])
Distance_down[4,:] = abs(edge_down[5,:]-edge_down[4,:])
Distance_down[5,:] = abs(edge_down[6,:]-edge_down[5,:])
```

```
#Normal vectors

#Upstream box
nv_up_1 = Distance_up[0,:]/norm(Distance_up[0,:])
nv_up_2 = Distance_up[1,:]/norm(Distance_up[1,:])
nv_up_3 = Distance_up[2,:]/norm(Distance_up[2,:])
nv_up_4 = Distance_up[3,:]/norm(Distance_up[3,:])
nv_up_5 = Distance_up[4,:]/norm(Distance_up[4,:])
nv_up_6 = Distance_up[5,:]/norm(Distance_up[5,:])

#Downstream box
nv_down_1 = Distance_down[0,:]/norm(Distance_down[0,:])
nv_down_2 = Distance_down[1,:]/norm(Distance_down[1,:])
nv_down_3 = Distance_down[2,:]/norm(Distance_down[2,:])
nv_down_4 = Distance_down[3,:]/norm(Distance_down[3,:])
nv_down_5 = Distance_down[4,:]/norm(Distance_down[4,:])
nv_down_6 = Distance_down[5,:]/norm(Distance_down[5,:])

#%%

#Upstream
#Side 1
Px = P1*nv_up_1[1]
Py = P1*-nv_up_1[0]
PS = P1
PS_x1 = Px
PS_y1 = Py

#Side 2
Px = P2*nv_up_2[1]
Py = P2*-nv_up_2[0]
PS = P2
PS_x2 = Px
PS_y2 = Py

#Side 3
Px = P3*-nv_up_3[1]
Py = P3*-nv_up_3[0]
PS = P3
PS_x3 = Px
PS_y3 = Py

#Side 4
Px = P4*-nv_up_4[1]
Py = P4*nv_up_4[0]
PS = P4
PS_x4 = Px
PS_y4 = Py

#Side 5
Px = P5*nv_up_5[1]
Py = P5*nv_up_5[0]
PS = P5
PS_x5 = Px
PS_y5 = Py

#Side 6 - 9 taps
Px = P6*nv_up_6[1]
Py = P6*nv_up_6[0]
PS = P6
PS_x6 = Px
PS_y6 = Py

#Downstream box
#Side 7
Px = P7*-nv_down_1[1]
Py = P7*-nv_down_1[0]
PS = P7
PS_x7 = Px
PS_y7 = Py

#Side 8
Px = P8*-nv_down_2[1]
Py = P8*-nv_down_2[0]
PS = P8
PS_x8 = Px
PS_y8 = Py

#Side 9
Px = P9*nv_down_3[1]
Py = P9*-nv_down_3[0]
PS = P9
PS_x9 = Px
PS_y9 = Py

#Side 10
```

```
Px = P10*nv_down_4[1]
Py = P10*nv_down_4[0]
PS = P10
PS_x10 = Px
PS_y10 = Py

#Side 11
Px = P11*nv_down_5[1]
Py = P11*nv_down_5[0]
PS = P11
PS_x11 = Px
PS_y11 = Py

#Side 12
Px = P12*-nv_down_6[1]
Py = P12*nv_down_6[0]
PS = P12
PS_x12 = Px
PS_y12 = Py

#%% Pressure distribution

#Upstream box
Px1 = X_S1 + PS_x1
Py1 = Y_S1 + PS_y1

Px2 = X_S2 + PS_x2
Py2 = Y_S2 + PS_y2

Px3 = X_S3 + PS_x3
Py3 = Y_S3 + PS_y3

Px4= X_S4 + PS_x4
Py4 = Y_S4 + PS_y4

Px5 = X_S5 + PS_x5
Py5 = Y_S5 + PS_y5

Px6 = X_S6 + PS_x6
Py6 = Y_S6 + PS_y6

#Downstream box
Px7 = X_S7 + PS_x7
Py7 = Y_S7 + PS_y7

Px8 = X_S8 + PS_x8
Py8 = Y_S8 + PS_y8

Px9 = X_S9 + PS_x9
Py9 = Y_S9 + PS_y9

Px10= X_S10 + PS_x10
Py10 = Y_S10 + PS_y10

Px11 = X_S11 + PS_x11
Py11 = Y_S11 + PS_y11

Px12 = X_S12 + PS_x12
Py12 = Y_S12 + PS_y12

Pres_up_x = np.concatenate((Px1, Px2, Px3, Px4, Px5, Px6))
Pres_up_y = np.concatenate((Py1, Py2, Py3, Py4, Py5, Py6))

Pres_down_x = np.concatenate((Px7, Px8, Px9, Px10, Px11, Px12))
Pres_down_y = np.concatenate((Py7, Py8, Py9, Py10, Py11, Py12))


#%% Plotting the distributed pressure
file = folder.split('C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/ScannerFiler/ScannerFiler/')
file = file[1]
savename1 = file + '_DistributedPressure_Line1'

xy = np.zeros((len(X_coord_up)*2, 2))
xy_pressure = np.zeros((len(X_coord_up)*2, 2))

for i in range(len(X_coord_up)):
    xy[i,:] = [X_coord_up[i], Y_coord_up[i]]
    xy[i+len(X_coord_up),:] = [X_coord_down[i], Y_coord_down[i]]

for i in range(len(X_coord_up)):
    xy_pressure[i,:] = [Pres_up_x[i], Pres_up_y[i]]
    xy_pressure[i+len(X_coord_up),:] = [Pres_down_x[i],Pres_down_y[i]]

dx = xy_pressure[:,0] - xy[:,0]
dy = xy_pressure[:,1] - xy[:,1]
```

```
fig = plt.figure(figsize = (12, 4), dpi = 100)
plt.plot(x_coord_up, y_coord_up, color = 'black', linewidth = 1.5)
plt.plot(x_coord_down, y_coord_down,color = 'black', linewidth = 1.5)
plt.axis('equal')

ax = fig.add_subplot(111)
cmap = plt.get_cmap('jet') #viridis

#Normalizer
D = np.sqrt(dx**2 + dy**2)

norm = mpl.colors.Normalize(0, max(D))
scalarMap = cmx.ScalarMappable(norm=norm,cmap=cmap)

for i in range(2*len(X_coord_up)):
    colorVal = scalarMap.to_rgba(D[i])
    ax.annotate("", xy = (xy_pressure[i,0],xy_pressure[i,1]), xycoords='data',
                xytext=(xy[i,0], xy[i,1]), textcoords='data',
                arrowprops=dict(arrowstyle = "-", connectionstyle="arc3",
                color=colorVal, linewidth = 1.5))

#Creating ScalarMappable
sm = plt.cm.ScalarMappable(cmap = cmap, norm = norm)
sm.set_array([])

ticks = np.arange(0,max(D),10)

clb = plt.colorbar(sm, ticks = ticks, pad = 0.02)
clb.set_label('Pressure [Pa]', labelpad = 10)
plt.title('Distributed Pressure', fontsize = 15)
plt.xlabel('x')
plt.ylabel('z')
plt.savefig(savename1 + '.png', bbox_inches='tight')
plt.show()

#%% Converting pressure to loads
w = 1.0
Dt = dt*10**(-3)

FS = np.zeros(1)
FS_y = np.zeros(1)
FS_x  = np.zeros(1)

for i in range(1,13):

    if i == 1:
        PS = P1
        Py = PS_y1
        Px = PS_x1

    elif i == 2:
        PS = P2
        Py = PS_y2
        Px = PS_x2

    elif i == 3:
        PS = P3
        Py = PS_y3
        Px = PS_x3

    elif i == 4:
        PS = P4
        Py = PS_y4
        Px = PS_x4

    elif i == 5:
        PS = P5
        Py = PS_y5
        Px = PS_x5

    elif i == 6:
        PS = P2
        Py = PS_y6
        Px = PS_x6

    elif i == 7:
        PS = P7
        Py = PS_y7
        Px = PS_x7

    elif i == 8:
        PS = P8
        Py = PS_y8
        Px = PS_x8

    elif i == 9:
        PS = P9
```

```
    Py = PS_y9
    Px = PS_x9

elif i == 10:
    PS = P10
    Py = PS_y10
    Px = PS_x10

elif i == 11:
    PS = P11
    Py = PS_y11
    Px = PS_x11

elif i == 12:
    PS = P12
    Py = PS_y12
    Px = PS_x12

F = np.zeros(len(PS))
Fy = np.zeros(len(Py))
Fx = np.zeros(len(Px))

F[1:-2] = PS[1:-2]*Dt*w
F[0] = PS[0]*Dt*w/2
F[-1] = PS[-1]*Dt*w/2

Fy[1:-2] = Py[1:-2]*Dt*w
Fy[0] = Py[0]*Dt*w/2
Fy[-1] = Py[-1]*Dt*w/2

Fx[1:-2] = Px[1:-2]*Dt*w
Fx[0] = Px[0]*Dt*w/2
Fx[-1] = Px[-1]*Dt*w/2

if i == 1:
    F1 = F
    Fy_1 = Fy
    Fx_1 = Fx

elif i == 2:
    F2 = F
    Fy_2 = Fy
    Fx_2 = Fx

elif i == 3:
    F3 = F
    Fy_3 = Fy
    Fx_3 = Fx

elif i == 4:
    F4 = F
    Fy_4 = Fy
    Fx_4 = Fx

elif i == 5:
    F5 = F
    Fy_5 = Fy
    Fx_5 = Fx

elif i == 6:
    F6 = F
    Fy_6 = Fy
    Fx_6 = Fx

elif i == 7:
    F7 = F
    Fy_7 = Fy
    Fx_7 = Fx

elif i == 8:
    F8 = F
    Fy_8 = Fy
    Fx_8 = Fx

elif i == 9:
    F9 = F
    Fy_9 = Fy
    Fx_9 = Fx

elif i == 10:
    F10 = F
    Fy_10 = Fy
    Fx_10 = Fx

elif i == 11:
    F11 = F
    Fy_11 = Fy
```

```
        Fx_11 = Fx

    elif i == 12:
        F12 = F
        Fy_12 = Fy
        Fx_12 = Fx

    FS = np.concatenate((FS,F))
    FS_y = np.concatenate((FS_y,Fy))
    FS_x = np.concatenate((FS_x,Fx))

Fx_up = np.concatenate((Fx_1, Fx_2, Fx_3, Fx_4, Fx_5, Fx_6))
Fy_up = np.concatenate((Fy_1, Fy_2, Fy_3, Fy_4, Fy_5, Fy_6))

Fx_down = np.concatenate((Fx_7, Fx_8, Fx_9, Fx_10, Fx_11, Fx_12))
Fy_down = np.concatenate((Fy_7, Fy_8, Fy_9, Fy_10, Fy_11, Fy_12))

Sum_Fx_up = Fx_1.sum() + Fx_2.sum() + Fx_3.sum() +  Fx_4.sum() +  Fx_5.sum() + Fx_6.sum()
Sum_Fy_up = Fy_1.sum() + Fy_2.sum() + Fy_3.sum() +  Fy_4.sum() +  Fy_5.sum() + Fy_6.sum()

Sum_Fx_down = Fx_7.sum() + Fx_8.sum() + Fx_9.sum() +  Fx_10.sum() +  Fx_11.sum() + Fx_12.sum()
Sum_Fy_down = Fy_7.sum() + Fy_8.sum() + Fy_9.sum() +  Fy_10.sum() +  Fy_11.sum() + Fy_12.sum()


#%% Plotting distributed load
scale = 1000

#Upstream box
L_x1 = X_S1 + Fx_1*scale
L_y1 = Y_S1 + Fy_1*scale

L_x2 = X_S2 + Fx_2*scale
L_y2 = Y_S2 + Fy_2*scale

L_x3 = X_S3 + Fx_3*scale
L_y3 = Y_S3 + Fy_3*scale

L_x4 = X_S4 + Fx_4*scale
L_y4 = Y_S4 + Fy_4*scale

L_x5 = X_S5 + Fx_5*scale
L_y5 = Y_S5 + Fy_5*scale

L_x6 = X_S6 + Fx_6*scale
L_y6 = Y_S6 + Fy_6*scale

#Downstream box
L_x7 = X_S7 + Fx_7*scale
L_y7 = Y_S7 + Fy_7*scale

L_x8 = X_S8 + Fx_8*scale
L_y8 = Y_S8 + Fy_8*scale

L_x9 = X_S9 + Fx_9*scale
L_y9 = Y_S9 + Fy_9*scale

L_x10 = X_S10 + Fx_10*scale
L_y10 = Y_S10 + Fy_10*scale

L_x11 = X_S11 + Fx_11*scale
L_y11 = Y_S11 + Fy_11*scale

L_x12 = X_S12 + Fx_12*scale
L_y12 = Y_S12 + Fy_12*scale

Load_up_x = np.concatenate((L_x1, L_x2, L_x3, L_x4, L_x5, L_x6))
Load_up_y = np.concatenate((L_y1, L_y2, L_y3, L_y4, L_y5, L_y6))

Load_down_x = np.concatenate((L_x7, L_x8, L_x9, L_x10, L_x11, L_x12))
Load_down_y = np.concatenate((L_y7, L_y8, L_y9, L_y10, L_y11, L_y12))


#%% Plotting

file = folder.split('C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/ScannerFiler/ScannerFiler/')
file = file[1]
savename1 = file + '_ILM_TotalLoad_Line1'


xy = np.zeros((len(X_coord_up)*2, 2))
xy_load = np.zeros((len(X_coord_up)*2, 2))

for i in range(len(X_coord_up)):
    xy[i,:] = [X_coord_up[i], Y_coord_up[i]]
    xy[i+len(X_coord_up),:] = [X_coord_down[i], Y_coord_down[i]]

for i in range(len(X_coord_up)):
```

```
    xy_load[i,:] = [Load_up_x[i], Load_up_y[i]]
    xy_load[i+len(X_coord_up),:] = [Load_down_x[i],Load_down_y[i]]

fig = plt.figure(figsize = (12, 4), dpi = 100)
plt.plot(x_coord_up, y_coord_up, color = 'black', linewidth = 1.5)
plt.plot(x_coord_down, y_coord_down,color = 'black', linewidth = 1.5)
plt.axis('equal')

ax = fig.add_subplot(111)
cmap = plt.get_cmap('jet')


dx = xy_load[:,0] - xy[:,0]
dy = xy_load[:,1] - xy[:,1]

#Normalizer
D = np.sqrt(dx**2 + dy**2)

norm = mpl.colors.Normalize(0, max(D))
scalarMap = cmx.ScalarMappable(norm=norm,cmap=cmap)

for i in range(len(X_coord_up)*2):
    colorVal = scalarMap.to_rgba(D[i])
    ax.annotate("", xy = (xy_load[i,0],xy_load[i,1]), xycoords='data',
                xytext=(xy[i,0], xy[i,1]), textcoords='data',
                arrowprops=dict(arrowstyle = "-", connectionstyle="arc3",
                color=colorVal, linewidth = 1.5))

#Creating ScalarMappable
sm = plt.cm.ScalarMappable(cmap = cmap, norm = norm)
sm.set_array([])
ticks = np.arange(0,max(D),10)

clb = plt.colorbar(sm, ticks = ticks, pad = 0.02)
clb.set_label('Force [N] $\cdot 10^{-3}$', labelpad = 10)
plt.title('Distributed Load', fontsize = 15)
plt.xlabel('x')
plt.ylabel('z')
plt.savefig(savename1 + 'DistributedLoad.png', bbox_inches='tight')
plt.show()

#%% Moment forces
center = [0,0]

#Moment arm, converting from mm to m
dist_up_x = abs(xy[0:691,0]*10**(-3))
dist_up_y = abs(xy[0:691,1]*10**(-3))
dist_down_x = abs(xy[691:,0]*10**(-3))
dist_down_y = abs(xy[691:,1]*10**(-3))

m1 = len(X_S1)
m2 = len(X_S2)
m3 = len(X_S3)
m4 = len(X_S4)
m5 = len(X_S5)
m6 = len(X_S6)

M = m1+m2+m3+m4+m5+m6

Moment_PS_up = np.zeros(len(X_coord_up))
Moment_PS_down = np.zeros(len(X_coord_up))

#Upstream box
for i in range(len(X_coord_up)):
    if X_coord_up[i] <= center[0] and Y_coord_up[i] >= center[1]: #Tube nr. 1-23
        if i <= m1+m2:
            Moment_PS_up[i] = Fx_up[i]*dist_up_y[i] + Fy_up[i]*dist_up_x[i]
        else: #Tube nr. 22-23
             Moment_PS_up[i] =  Fx_up[i]*dist_up_y[i] + Fy_up[i]*dist_up_x[i]
    else: #Tube nr. 24-48
        if i >= M-m6: #Tube nr. 38-48
            Moment_PS_up[i] = -Fx_up[i]*dist_up_y[i] + Fy_up[i]*dist_up_x[i]
        elif M-m6 <= i <= M-m6-m5: #Tube nr. 30-37
            Moment_PS_up[i] = Fy_up[i]*dist_up_x[i]
        elif i >= m1+m2+m3:#Tube nr. 25-29
            Moment_PS_up[i] = - Fx_up[i]*dist_up_y[i] + Fy_up[i]*dist_up_x[i]
        else: #Tube 24
            Moment_PS_up[i] = - Fx_up[i]*dist_up_y[i] + Fy_up[i]*dist_up_x[i]


#Downstream box
for i in range(len(X_coord_down)):
    if X_coord_down[i] >= center[0] and Y_coord_up[i] >= center[1]: #Tube nr. 1-23
        if i <= m1+m2: #Tube nr. 1-21
             Moment_PS_down[i] =  Fx_down[i]*dist_down_y[i] - Fy_down[i]*dist_down_x[i]
        else: #Tube nr. 22-23
             Moment_PS_down[i] =  Fx_down[i]*dist_down_y[i] - Fy_down[i]*dist_down_x[i]
```

```
    else: #Tube nr. 24-48
        if i >= M-m6: #Tube nr. 38-48
            Moment_PS_down[i] =  - Fx_down[i]*dist_down_y[i] - Fy_down[i]*dist_down_x[i]
        elif M-m6 <= i <= M-m6-m5: #Tube nr. 30-37
            Moment_PS_down[i] = - Fy_down[i]*dist_down_x[i]
        elif i >= m1+m2+m3:#Tube nr. 25-29
            Moment_PS_down[i] = - Fx_down[i]*dist_down_y[i] - Fy_down[i]*dist_down_x[i]
        else: #Tube 24
            Moment_PS_down[i] = - Fx_down[i]*dist_down_y[i] - Fy_down[i]*dist_down_x[i]


Sum_Moment_up = sum(Moment_PS_up)
Sum_Moment_down = sum(Moment_PS_down)

#Save forces and moments
savename1 = savename1 + '.npz'
np.savez_compressed(savename1, Fx_down = Sum_Fx_down, Fx_up = Sum_Fx_up, Fz_down = Sum_Fy_down,
                    Fz_up = Sum_Fy_up, Moment_up = Sum_Moment_up, Moment_down = Sum_Moment_down)
```

B.3 Functions for Load Estimation

Several functions have been used in the script for The Piece-wise Load Method and The Interpolated Load Method:

- **CoordinatesAndAreaFunc**: Imports coordinates to the cross-section and pressure tubes for different angles of attack. In addition, it imports the area and angles of the cross-section.
- **SortPressure**: Sorts the pressure from the pressure scanner to the right correlation line.
- **Area16Taps**: Finds surface area for the point pressures in correlation line 2-6.
- **CoordinatesEqualSpacing**: Used in the Interpolated Load Method. The function returns new coordinates with equal spacing, dt.

B.3.1 CoordinatesAndAreaFunc

```python
# -*- coding: utf-8 -*-
"""
Created on Thu Feb 17 10:58:33 2022

@author: marth
"""

import pandas as pd
import numpy as np

def CoordinatesAndArea(deg):

    if deg == 0:

        #Coordinates, including corners
        Coordinates_cross = pd.read_excel(r'C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/Python/Koordinater.xlsx',
                                          sheet_name ='Koordinater 48')

        Coordinates_cross = np.array(Coordinates_cross)

        #Coordiantes to pressure taps only
        Coordinates = pd.read_excel(r'C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/Python/Koordinater.xlsx',
                                    sheet_name='Taps 48')

        Coordinates = np.array(Coordinates)


        # 48 tubes
        #Upstream box
        x_taps_up = Coordinates[0:,2] #Only pressure tubes
        y_taps_up = Coordinates[0:,3]
        x_coord_up = Coordinates_cross[0:,2] #Coordinates for pressure tubes and corners
        y_coord_up = Coordinates_cross[0:,3]

        #Downstream
        x_taps_down = -x_taps_up
        y_taps_down = y_taps_up
        x_coord_down = -x_coord_up
        y_coord_down = y_coord_up


    else:

        #Coordinates
        Coordinates_cross = pd.read_excel(r'C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/Python/Koordinater.xlsx',
                                          sheet_name ='Upstream ' + str(deg))

        Coordinates_cross = np.array(Coordinates_cross)

        Coordinates_cross_d = pd.read_excel(r'C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/Python/Koordinater.xlsx',
                                            sheet_name='Downstream ' + str(deg))

        Coordinates_cross_d = np.array(Coordinates_cross_d)

        #Coordiantes to pressure taps only
        Coordinates = pd.read_excel(r'C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/Python/Koordinater.xlsx',
                                    sheet_name='Upstream taps ' + str(deg))

        Coordinates = np.array(Coordinates)

        Coordinates_d = pd.read_excel(r'C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/Python/Koordinater.xlsx',
                                      sheet_name='Downstream taps ' + str(deg))

        Coordinates_d = np.array(Coordinates_d)


        # 48 tubes
        #Upstream box
        x_taps_up = Coordinates[0:,2]
        y_taps_up = Coordinates[0:,3]
        x_coord_up = Coordinates_cross[0:,2]
        y_coord_up = Coordinates_cross[0:,3]

        #Downstream box
        x_taps_down = Coordinates_d[0:,2]
        y_taps_down = Coordinates_d[0:,3]
        x_coord_down = Coordinates_cross_d[0:,2]
        y_coord_down = Coordinates_cross_d[0:,3]


    #For zero degrees only (16 tubes)
    #Coordinates cross-section corrline 2-6
    Coordinates_16_cross = pd.read_excel(r'C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/Python/Koordinater.xlsx',
                                         sheet_name='Koordinater 16')

    Coordinates_16_cross = np.array(Coordinates_16_cross)
```

```
#Coordinates to pressure taps, only corrline 2-6
Coordinates_16= pd.read_excel(r'C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/Python/Koordinater.xlsx',
                              sheet_name='Taps 16')

Coordinates_16 = np.array(Coordinates_16)

#16 tubes
x_taps_up_16 = Coordinates_16[0:,2]
y_taps_up_16 = Coordinates_16[0:,3]
x_coord_up_16= Coordinates_16_cross[0:,2]
y_coord_up_16 = Coordinates_16_cross[0:,3]

x_taps_down_16 = - x_taps_up_16
y_taps_down_16 = y_taps_up_16
x_coord_down_16 = - x_coord_up_16
y_coord_down_16 = y_coord_up_16


# Angles, upstream box
a_S1_up = np.arctan2(abs(y_coord_up[4]-y_coord_up[0]),abs(x_coord_up[4]-x_coord_up[0]))*180/np.pi
a_S2_up = np.arctan2(abs(y_coord_up[24]-y_coord_up[5]),abs(x_coord_up[24]-x_coord_up[5]))*180/np.pi
a_S3_up = np.arctan2(abs(y_coord_up[29]-y_coord_up[25]),abs(x_coord_up[29]-x_coord_up[25]))*180/np.pi
a_S4_up = np.arctan2(abs(y_coord_up[36]-y_coord_up[30]),abs(x_coord_up[36]-x_coord_up[30]))*180/np.pi
a_S5_up = np.arctan2(abs(y_coord_up[46]-y_coord_up[37]),abs(x_coord_up[46]-x_coord_up[37]))*180/np.pi
a_S6_up = np.arctan2(abs(y_coord_up[59]-y_coord_up[47]),abs(x_coord_up[59]-x_coord_up[47]))*180/np.pi

angle_up =  [a_S1_up, a_S2_up, a_S3_up, a_S4_up, a_S5_up, a_S6_up]


# Angles, downstream box
a_S1_down = np.arctan2(abs(y_coord_down[4]-y_coord_down[0]),abs(x_coord_down[4]-x_coord_down[0]))*180/np.pi
a_S2_down = np.arctan2(abs(y_coord_down[24]-y_coord_down[5]),abs(x_coord_down[24]-x_coord_down[5]))*180/np.pi
a_S3_down = np.arctan2(abs(y_coord_down[29]-y_coord_down[25]),abs(x_coord_down[29]-x_coord_down[25]))*180/np.pi
a_S4_down = np.arctan2(abs(y_coord_down[36]-y_coord_down[30]),abs(x_coord_down[36]-x_coord_down[30]))*180/np.pi
a_S5_down = np.arctan2(abs(y_coord_down[46]-y_coord_down[37]),abs(x_coord_down[46]-x_coord_down[37]))*180/np.pi
a_S6_down = np.arctan2(abs(y_coord_down[59]-y_coord_down[47]),abs(x_coord_down[59]-x_coord_down[47]))*180/np.pi

angle_down =  [a_S1_down, a_S2_down, a_S3_down, a_S4_down, a_S5_down, a_S6_down]


#Length between taps - first number is edge to tap, last number is tap to edge

S1 = [12, 12, 11.62, 3.5] #3
S2 = [3.5, 12.07, 12.07, 12.07, 12.07, 12.07, 12.07, 12.07, 12.07, 17.93, 17.93, 17.93, 22.46, 20.06, 20.06, 20.06,
     20.06, 20.06, 3.5] #22
S3 = [3.5, 13.2, 13.2, 5] #26
S4 = [5, 17.11, 17.11, 17.11, 17.11, 3.8] #32
S5 = [3.5, 20.13, 20.13, 22.5, 12.56, 12.56, 12.56, 12.56, 3.5] #41
S6 = [3.5, 12.89, 12.88, 12.88, 12.88, 12.05, 12.05, 12.05, 12.05, 12.05, 12.05, 10] #53

Surface = S1 + S2 + S3 + S4 + S5 + S6

# Surface area for point loads

A = np.zeros(49)

#Side 1: 0-2
A[0] = Surface[0]+Surface[1]/2
A[1] = (Surface[1]+Surface[2])/2
A[2] = Surface[2]/2+Surface[3]

#Side 2: 3-20
A[3] = Surface[4]+Surface[5]/2

for i in range(4,20):
    A[i] = (Surface[i+1]+Surface[i+2])/2

A[20] = Surface[21]/2 + Surface[22]

#Side 3: 21-23
A[21] = Surface[23] + Surface[24]/2
A[22] = (Surface[24] + Surface[25])/2
A[23] = Surface[25]/2 + Surface[26]

#Side 4: 24-28
A[24] = Surface[27] + Surface[28]/2

for i in range(25,28):
    A[i] = (Surface[i+3]+Surface[i+4])/2

A[28] = Surface[31]/2 + Surface[32]

 #Side 5: 29-36
A[29] = Surface[33]+Surface[34]/2

for i in range(30,36):
    A[i] = (Surface[i+4]+Surface[i+5])/2
```

```
A[36] = Surface[40]/2 + Surface[41]

#Side 6: 37-48
A[37] = Surface[42] + Surface[43]/2

for i in range(38,48):
    A[i] = (Surface[i+5]+Surface[i+6])/2

A[48] = Surface[52]/2 + Surface[53]


return(x_taps_up, y_taps_up, x_taps_down, y_taps_down, x_coord_up, y_coord_up, x_coord_down,y_coord_down,
       x_taps_up_16, y_taps_up_16, x_taps_down_16, y_taps_down_16, x_coord_up_16, y_coord_up_16, x_coord_down_16,
       y_coord_down_16, Surface, A, angle_up, angle_down)
```

B.3.2 SortPressure

```python
# -*- coding: utf-8 -*-
"""
Created on Thu Mar 17 14:22:54 2022

@author: marth
"""

import numpy as np

def SortPressure(pres,scanner):
    """
    Sorting the pressure from the pressure scanners after correlation line

    Parameters
    ----------
    pres : pressure from pressure scanners
    scanner: number on scanner (Upstream = 1,2  Downstream = 3,4)

    Returns
    -------
    Pres_up1 = Upstream box, correlation line 1  etc.
    Pres_down1 = Downstream box, correlation line 1 etc.

    """

    Pres_up1 = np.zeros((len(pres),48))
    Pres_up2 = np.zeros((len(pres),16))
    Pres_up3 = np.zeros((len(pres),16))
    Pres_up4 = np.zeros((len(pres),16))
    Pres_up5 = np.zeros((len(pres),16))
    Pres_up6 = np.zeros((len(pres),16))

    Pres_down1 = np.zeros((len(pres),48))
    Pres_down2 = np.zeros((len(pres),16))
    Pres_down3 = np.zeros((len(pres),16))
    Pres_down4 = np.zeros((len(pres),16))
    Pres_down5 = np.zeros((len(pres),16))
    Pres_down6 = np.zeros((len(pres),16))

    #Upstream box, scanner 1
    if scanner == 1:
        Pres_up1[:,0:12] = pres[:,0:12]
        Pres_up1[:,36:48] = pres[:,12:24]

        Pres_up2[:,0:4] = pres[:,24:28]
        Pres_up2[:,12:16] = pres[:,28:32]

        Pres_up3[:,0:4] = pres[:,32:36]
        Pres_up3[:,12:16] = pres[:,36:40]

        Pres_up4[:,0:4] = pres[:,40:44]
        Pres_up4[:,12:16] = pres[:,44:48]

        Pres_up5[:,0:4] = pres[:,48:52]
        Pres_up5[:,12:16] = pres[:,52:56]

        Pres_up6[:,0:4] = pres[:,56:60]
        Pres_up6[:,12:16] = pres[:,60:64]

        return Pres_up1, Pres_up2, Pres_up3, Pres_up4, Pres_up5, Pres_up6
```

```
#Upstream box, scanner 2
elif scanner == 2:
    Pres_up1[:,12:36] = pres[:,0:24]

    Pres_up2[:,4:12] = pres[:,24:32]

    Pres_up3[:,4:12] = pres[:,32:40]

    Pres_up4[:,4:12] = pres[:,40:48]

    Pres_up5[:,4:12] = pres[:,48:56]

    Pres_up6[:,4:12] = pres[:,56:64]

    return Pres_up1, Pres_up2, Pres_up3, Pres_up4, Pres_up5, Pres_up6



#Downstream box, scanner 3
elif scanner == 3:
    Pres_down1[:,12:36] = pres[:,0:24]

    Pres_down2[:,4:12] = pres[:,24:32]

    Pres_down3[:,4:12] = pres[:,32:40]

    Pres_down4[:,4:12] = pres[:,40:48]

    Pres_down5[:,4:12] = pres[:,48:56]

    Pres_down6[:,4:12] = pres[:,56:64]

    return Pres_down1, Pres_down2, Pres_down3, Pres_down4, Pres_down5, Pres_down6


#Downstream box, scanner 4
elif scanner == 4:
    Pres_down1[:,0:12] = pres[:,0:12]
    Pres_down1[:,36:48] = pres[:,12:24]

    Pres_down2[:,0:4] = pres[:,24:28]
    Pres_down2[:,12:16] = pres[:,28:32]

    Pres_down3[:,0:4] = pres[:,32:36]
    Pres_down3[:,12:16] = pres[:,36:40]

    Pres_down4[:,0:4] = pres[:,40:44]
    Pres_down4[:,12:16] = pres[:,44:48]

    Pres_down5[:,0:4] = pres[:,48:52]
    Pres_down5[:,12:16] = pres[:,52:56]

    Pres_down6[:,0:4] = pres[:,56:60]
    Pres_down6[:,12:16] = pres[:,60:64]

    return Pres_down1, Pres_down2, Pres_down3, Pres_down4, Pres_down5, Pres_down6
```

B.3.3 Area16Taps

```python
# -*- coding: utf-8 -*-
"""
Created on Mon Feb 28 15:46:15 2022

@author: marth
"""

import numpy as np

def Area16Taps():

    #Finding surface area for point loads for correlation line 2-6.

    S1 = [23.62, 15.12] #1
    S2 = [3.5, 48.28, 48.28,76.25, 60.19, 43.62] #7
    S3 = [16.8, 18.2] #9
    S4 = [5, 34.23, 38.02] #12
    S5 = [3.5, 40.26, 35.06, 37.68, 3.5] #17
    S6 = [42.15, 49.02, 36.14, 10] #21

    Surface = S1 + S2 + S3 + S4 + S5 + S6

    #Surface area for point loads
    A = np.zeros(16)

    #Side 1
    A[0] = Surface[0] + Surface[1]

    #Side 2
    A[1] = Surface[2]+Surface[3]/2

    for i in range(2,5):
        A[i] =(Surface[i+1]+Surface[i+2])/2

    A[5] = Surface[6]+Surface[7]/2

    #Side 3
    A[6] = Surface[8]+Surface[9]

    #Side 4
    A[7] = Surface[10] + Surface[11]/2
    A[8] = (Surface[11] + Surface[12])/2

    #Side 5
    A[9] = Surface[13] + Surface[14]/2

    for i in range(10,12):
        A[i] = (Surface[i+4]+Surface[i+5])/2

    A[12] = Surface[16]/2 + Surface[17]

    #Side 6
    A[13] = Surface[18]+Surface[19]/2
    A[14]= (Surface[19] + Surface[20])/2
    A[15] = Surface[20]/2 + Surface[21]

    A = A.transpose()

    return A
```

B.3.4 CoordinatesEqualSpacing

```python
# -*- coding: utf-8 -*-
"""
Created on Tue Mar  1 10:05:56 2022

@author: marth
"""

import numpy as np

def CoordinatesEqualSpacing(x_coord, y_coord, dt, angle, N):

    """
    #Makes new coordinate with equal spacing, dt.

    Parameters
    -----------
    x_coord: coordiantes in x-direction
    y_coord: coordinates im y-direction
    dt: spacing
    angle: angle between sides
    N:Number of coordinates

    Returns
    ----------
    X: coordinates in x-direction with spacing dt
    Y: coordinates in y-direction with spacing dt

    """

    X = np.zeros((N))
    Y = np.zeros((N))

    X[0] = x_coord
    Y[0] = y_coord

    for i in range(1,N):
        X[i] = -dt*np.cos((np.pi/180)*angle)+X[i-1]
        Y[i] = -dt*np.sin((np.pi/180)*angle)+Y[i-1]

    return X, Y
```

Appendix C

Python Script for Estimation of Aerodynamic Admittance Functions

This appendix contains the Python scripts for estimation of the aerodynamic admittance functions using the general, the auto-spectral and the cross-spectral method. In addition, a function for importing the processed data from Matlab and a function for calculating the static load coefficients is attached.

C.1 Aerodynamic Admittance Functions

Script for estimating the aerodynamic admittance functions. The script also contains codes for plotting the buffeting force spectra, turbulence spectra and coherence.

```python
# -*- coding: utf-8 -*-
"""
Created on Wed Mar 16 10:51:45 2022

@author: marth
"""

import numpy as np
import math
from matplotlib import pyplot as plt
from scipy import signal
from ImportDataFromMatlab import ImportMeasuredData
import numpy.polynomial.polynomial as poly

#%% Importing processed data from Matlab

MatFile = 'AGTD21_S2-G1_Pressure_CPD105_03_12_003.mat'
(Pitotprobe, Displacements, Forces_global, Forces_cr, Temperature, Frequency,
AirDensity, Cobra, t) = ImportMeasuredData(MatFile)

#%%Time series of forces and turbulence

#Measured mean wind velocity
u = Cobra[0,:]
v = Cobra[1,:]
w = Cobra[2,:]

#Remove zero values at the end of the matrix
u = u[:61000]
v = v[:61000]
w = w[:61000]

#Sampling frequency
F = Frequency

#Time vector
t = np.arange(0,len(u)/F,1/F)
dt = t[1]- t[0]

#Air density
rho = AirDensity

#Mean wind flow
U = np.mean(u)
V = U

D = 0.05 #Height
B = 0.74 #Width
L = 2.64 #Length

#Dynamic pressure
Pd = 0.5*rho*U**2


#%% Import loads
test = '05_03_2022_12_48_38_PM'
file = test + '_PointLoad_Line1.npz'
PointLoad = np.load(file)

Load_up_x = PointLoad['Load_up_x']
```

```
Load_up_y = PointLoad['Load_up_y']
M_up = PointLoad['Moment_up']

Load_down_x = PointLoad['Load_down_x']
Load_down_y = PointLoad['Load_down_y']
M_down = PointLoad['Moment_down']

#Local to global forces
deg = 0
beta = deg*np.pi/180

#Drag
Fx_up = Load_up_x*np.cos(beta) + Load_up_y*np.sin(beta)
Fx_down = Load_down_x*np.cos(beta) + Load_down_y*np.sin(beta)
Fx_tot = Fx_up + Fx_down

#Lift
Fz_up = -Load_up_x*np.sin(beta) + Load_up_y*np.cos(beta)
Fz_down = -Load_down_x*np.sin(beta) + Load_down_y*np.cos(beta)
Fz_tot = Fz_up + Fz_down

#Moment
M_tot = M_up + M_down


Drag_up = np.zeros(len(Fz_up))
Drag_down = np.zeros(len(Fz_up))
Drag_tot = np.zeros(len(Fz_up))

Lift_up = np.zeros(len(Fz_up))
Lift_down = np.zeros(len(Fz_up))
Lift_tot = np.zeros(len(Fz_up))

Moment_up = np.zeros(len(Fz_up))
Moment_down = np.zeros(len(Fz_up))
Moment_tot = np.zeros(len(Fz_up))


for i in range(len(Fz_up)):
    Drag_up[i] = np.sum(Fx_up[i,:])
    Drag_down[i] = np.sum(Fx_down[i,:])
    Drag_tot[i] = np.sum(Fx_tot[i,:])

    Lift_up[i] = np.sum(Fz_up[i,:])
    Lift_down[i] = np.sum(Fz_down[i,:])
    Lift_tot[i] = np.sum(Fz_tot[i,:])

    Moment_up[i] = np.sum(M_up[i,:])
    Moment_down[i] = np.sum(M_down[i,:])
    Moment_tot[i] = np.sum(M_tot[i,:])


Drag_up = Drag_up - np.mean(Drag_up)
Drag_down = Drag_down - np.mean(Drag_down)
Drag_tot = Drag_tot - np.mean(Drag_tot)

Lift_up = Lift_up - np.mean(Lift_up)
Lift_down = Lift_down - np.mean(Lift_down)
Lift_tot = Lift_tot - np.mean(Lift_tot)
```

```
Moment_up = Moment_up - np.mean(Moment_up)
Moment_down = Moment_down - np.mean(Moment_down)
Moment_tot = Moment_tot - np.mean(Moment_tot)

#%% Equal sampling length

l = min(len(Fx_up),len(u),len(t))

u = u[:l]
v = v[:l]
w = w[:l]

t = t[:l]

#%% Buffeting force spectrum

Nwelch = 20 #Number of divisions
Nwindow = np.round(len(t)/Nwelch) #Length of window
nfft = 2**math.log2(Nwindow) #Number of FFT points
window = signal.windows.hamming(int(Nwindow))

f, SD_up =  signal.welch(Drag_up, F, window = window, nfft = nfft)
f, SD_down = signal.welch(Drag_down, F, window = window, nfft = nfft)
f, SD_tot = signal.welch(Drag_tot, F, window = window, nfft = nfft)

f, SL_up =  signal.welch(Lift_up, F, window = window, nfft = nfft)
f, SL_down = signal.welch(Lift_down, F, window = window , nfft = nfft)
f, SL_tot = signal.welch(Lift_tot, F, window = window, nfft = nfft)

f, SM_up =  signal.welch(Moment_up, F, window = window, nfft = nfft)
f, SM_down = signal.welch(Moment_down, F, window = window, nfft = nfft)
f, SM_tot = signal.welch(Moment_tot, F, window = window, nfft = nfft)

#%% Plot drag, lift and moment spectrum

plt.figure()
plt.plot(f, SD_up,label = 'Upstream', linewidth = 0.8)
plt.plot(f, SD_down,  label =  'Downstream', linewidth = 0.8)
plt.plot(f, SD_tot, label = 'Total', linewidth = 0.8)
plt.title('Drag Spectrum', fontsize = 15)
plt.yscale('log')
plt.xscale('log')
plt.xlabel('$f$ [Hz]')
plt.ylabel('$S_M(f)$')
plt.legend(loc = 3)
plt.xlim(000.1,100)
plt.ylim(10**(-6),1)

savename = file.split('.npz')
savename = savename[0]
plt.savefig(savename + 'Drag_9_7Hz.png', bbox_inches='tight')
plt.show()

plt.figure()
plt.plot(f, SL_up,label = 'Upstream', linewidth = 0.8)
plt.plot(f, SL_down,  label =  'Downstream', linewidth = 0.8)
plt.plot(f, SL_tot, label = 'Total', linewidth = 0.8)
plt.title('Lift Spectrum', fontsize = 15)
plt.yscale('log')
plt.xscale('log')
```

```
plt.xlabel('$f$ [Hz]')
plt.ylabel('$S_M(f)$')
plt.legend(loc = 3)
plt.xlim(00.1,100)
plt.ylim(10**(-8),10)

plt.savefig(savename + 'Lift_9_7Hz.png', bbox_inches='tight')
plt.show()


plt.figure()
plt.plot(f, SM_up,label = 'Upstream', linewidth = 0.8)
plt.plot(f, SM_down,  label =  'Downstream', linewidth = 0.8)
plt.plot(f, SM_tot, label = 'Total', linewidth = 0.8)
plt.title('Moment Spectrum', fontsize = 15)
plt.yscale('log')
plt.xscale('log')
plt.xlabel('$f$ [Hz]')
plt.ylabel('$S_M(f)$')
plt.legend(loc = 3)
plt.xlim(00.1,100)
plt.ylim(10**(-8),1)

plt.savefig(savename + 'Moment_9_7Hz.png', bbox_inches='tight')
plt.show()

#%% Correlation line 2-6

#Import loads
file2 = test + '_PointLoad_Line2-6.npz'
PointLoad2 = np.load(file2)

Loadx_up_cl = PointLoad2['Load_up_x']
Loadz_up_cl = PointLoad2['Load_up_y']
M_up_cl = PointLoad2['Moment_up']

Loadx_down_cl = PointLoad2['Load_down_x']
Loadz_down_cl = PointLoad2['Load_down_y']
M_down_cl = PointLoad2['Moment_down']

#Drag
Fx_up_cl = Loadx_up_cl*np.cos(beta) + Loadz_up_cl*np.sin(beta)
Fx_down_cl = Loadx_down_cl*np.cos(beta) + Loadz_down_cl*np.sin(beta)
Fx_tot_cl = Fx_up_cl + Fx_down_cl

#Lift
Fz_up_cl = -Loadx_up_cl*np.sin(beta) + Loadz_up_cl*np.cos(beta)
Fz_down_cl = -Loadx_down_cl*np.sin(beta) + Loadz_down_cl*np.cos(beta)
Fz_tot_cl = Fz_up_cl + Fz_down_cl

#Moment
M_tot_cl = M_up_cl + M_down_cl

Drag_up_cl = np.zeros((len(Fz_up_cl), 5))
Drag_down_cl = np.zeros((len(Fz_up_cl),5))
Drag_tot_cl = np.zeros((len(Fz_up_cl),5))

Lift_up_cl = np.zeros((len(Fz_up_cl),5))
Lift_down_cl = np.zeros((len(Fz_up_cl),5))
Lift_tot_cl = np.zeros((len(Fz_up_cl),5))
```

```
Moment_up_cl = np.zeros((len(Fz_up_cl),5))
Moment_down_cl = np.zeros((len(Fz_up_cl),5))
Moment_tot_cl = np.zeros((len(Fz_up_cl),5))


Drag_up_cl = np.sum(Fx_up_cl, axis = 1)
Drag_down_cl = np.sum(Fx_down_cl, axis = 1)
Drag_tot_cl= np.sum(Fx_tot_cl, axis = 1)

Lift_up_cl= np.sum(Fz_up_cl, axis = 1)
Lift_down_cl= np.sum(Fz_down_cl, axis = 1)
Lift_tot_cl= np.sum(Fz_tot_cl, axis = 1)

Moment_up_cl= np.sum(M_up_cl, axis = 1)
Moment_down_cl= np.sum(M_down_cl, axis = 1)
Moment_tot_cl = np.sum(M_tot_cl, axis = 1)


for i in range(5):
    Drag_up_cl[:,i] = Drag_up_cl[:,i] - np.mean(Drag_up_cl[:,i])
    Drag_down_cl[:,i] = Drag_down_cl[:,i] - np.mean(Drag_down_cl[:,i])
    Drag_tot_cl[:,i] = Drag_tot_cl[:,i] - np.mean(Drag_tot_cl[:,i])

    Lift_up_cl[:,i] = Lift_up_cl[:,i]- np.mean(Lift_up_cl[:,i])
    Lift_down_cl[:,i] = Lift_down_cl[:,i] - np.mean(Lift_down_cl[:,i])
    Lift_tot_cl[:,i] = Lift_tot_cl[:,i] - np.mean(Lift_tot_cl[:,i])

    Moment_up_cl[:,i] = Moment_up_cl[:,i] - np.mean(Moment_up_cl[:,i])
    Moment_down_cl[:,i] = Moment_down_cl[:,i] - np.mean(Moment_down_cl[:,i])
    Moment_tot_cl[:,i] = Moment_tot_cl[:,i] - np.mean(Moment_tot_cl[:,i])


#%%Buffeting force spectrum - Line 2-6

Nwelch = 20
Nwindow = np.round(len(t)/Nwelch)
nfft = 2**math.log2(Nwindow)
window = signal.windows.hamming(int(Nwindow))

SD_up_cl = np.zeros((1401,5))
SD_down_cl = np.zeros((1401,5))
SD_tot_cl = np.zeros((1401,5))

SL_up_cl = np.zeros((1401,5))
SL_down_cl = np.zeros((1401,5))
SL_tot_cl = np.zeros((1401,5))

SM_up_cl = np.zeros((1401,5))
SM_down_cl = np.zeros((1401,5))
SM_tot_cl = np.zeros((1401,5))

for i in range(5):

    f, SD_up_cl[:,i] =  signal.welch(Drag_up_cl[:,i], F, window = window, nfft = nfft)
    f, SD_down_cl[:,i]= signal.welch(Drag_down_cl[:,i], F, window = window, nfft = nfft)
    f, SD_tot_cl[:,i]= signal.welch(Drag_tot_cl[:,i], F, window = window, nfft = nfft)

    f, SL_up_cl[:,i] =  signal.welch(Lift_up_cl[:,i], F, window = window, nfft = nfft)
    f, SL_down_cl[:,i] = signal.welch(Lift_down_cl[:,i], F,window = window, nfft = nfft)
```

```
    f, SL_tot_cl[:,i] = signal.welch(Lift_tot_cl[:,i], F, window = window, nfft = nfft)

    f, SM_up_cl[:,i] =  signal.welch(Moment_up_cl[:,i], F, window = window, nfft = nfft)
    f, SM_down_cl[:,i] = signal.welch(Moment_down_cl[:,i], F, window = window, nfft = nfft)
    f, SM_tot_cl[:,i] = signal.welch(Moment_tot_cl[:,i], F, window = window, nfft = nfft)

#%% Plot drag, lift and moment spectrum for all correlation lines

cm = 1/2.54
fig, axs = plt.subplots(1,3,figsize=(30*cm, 12*cm))

axs[0].plot(f, SD_tot, label = 'Line 1', linewidth = 0.8)
axs[0].plot(f, SD_tot_cl[:,0], label = 'Line 2', linewidth = 0.8)
axs[0].plot(f, SD_tot_cl[:,1], label = 'Line 3', linewidth = 0.8)
axs[0].plot(f, SD_tot_cl[:,2], label = 'Line 4', linewidth = 0.8)
axs[0].plot(f, SD_tot_cl[:,3], label = 'Line 5', linewidth = 0.8)
axs[0].plot(f, SD_tot_cl[:,4], label = 'Line 6', linewidth = 0.8, color = 'c')
axs[0].set_title('Drag Force Spectrum', fontsize = 15)
axs[0].set_yscale('log')
axs[0].set_xscale('log')
axs[0].set_xlabel('$f$ [Hz]')
axs[0].set_ylabel('$S_D(f)$')
axs[0].legend(loc = 3)
axs[0].set_xlim(00.1,100)
axs[0].set_ylim(0.00001,1)

axs[1].plot(f, SL_tot, label = 'Line 1', linewidth = 0.8)
axs[1].plot(f, SL_tot_cl[:,0], label = 'Line 2', linewidth = 0.8)
axs[1].plot(f, SL_tot_cl[:,1], label = 'Line 3', linewidth = 0.8)
axs[1].plot(f, SL_tot_cl[:,2], label = 'Line 4', linewidth = 0.8)
axs[1].plot(f, SL_tot_cl[:,3], label = 'Line 5', linewidth = 0.8)
axs[1].plot(f, SL_tot_cl[:,4], label = 'Line 6', linewidth = 0.8, color = 'c')
axs[1].set_title('Lift Force Spectrum', fontsize = 15)
axs[1].set_yscale('log')
axs[1].set_xscale('log')
axs[1].set_xlabel('$f$ [Hz]')
axs[1].set_ylabel('$S_L(f)$')
axs[1].set_xlim(00.1,100)
axs[1].set_ylim(0.00001,10)

axs[2].plot(f, SM_tot, label = 'Line 1', linewidth = 0.8)
axs[2].set_title('Moment Spectrum', fontsize = 15)
axs[2].plot(f, SM_tot_cl[:,0], label = 'Line 2', linewidth = 0.8)
axs[2].plot(f, SM_tot_cl[:,1], label = 'Line 3', linewidth = 0.8)
axs[2].plot(f, SM_tot_cl[:,2], label = 'Line 4', linewidth = 0.8)
axs[2].plot(f, SM_tot_cl[:,3], label = 'Line 5', linewidth = 0.8)
axs[2].plot(f, SM_tot_cl[:,4], label = 'Line 6', linewidth = 0.8, color = 'c')
axs[2].set_yscale('log')
axs[2].set_xscale('log')
axs[2].set_xlabel('$f$ [Hz]')
axs[2].set_ylabel('$S_M(f)$')
axs[2].set_xlim(00.1,100)
axs[2].set_ylim(0.000001,1)

plt.tight_layout()
plt.savefig('Drag_lift_moment_all_lines_9_Opengrid.png', bbox_inches='tight')

#%% Plot Buffeting spectra - Correlation line 2-6

cl = 2  #Correlation line
```

```
index = cl-2

plt.figure()
plt.plot(f, SD_up_cl[:,index],label = 'Upstream', linewidth = 1)
plt.plot(f, SD_down_cl[:,index],  label =  'Downstream', linewidth = 1)
plt.plot(f, SD_tot_cl[:,index], label = 'Total',linewidth = 1)
plt.title('Drag Force Spectra - Correlation line ' + str(cl), fontsize = 12)
plt.yscale('log')
plt.xscale('log')
plt.xlabel('$f$ [Hz]')
plt.ylabel('$S_L(f)$')
plt.legend(loc = 3)
plt.ylim(10**(-6),10)
plt.xlim(10**(-1), 10**(2))

plt.savefig(file + 'DragSpectra_Correlationline.png', bbox_inches='tight')
plt.show()

plt.figure()
plt.plot(f, SL_up_cl[:,index],label = 'Upstream', linewidth = 1)
plt.plot(f, SL_down_cl[:,index],  label =  'Downstream', linewidth = 1)
plt.plot(f, SL_tot_cl[:,index], label = 'Total',linewidth = 1)
plt.title('Lift Force Spectra - Correlation line ' + str(cl), fontsize = 12)

plt.yscale('log')
plt.xscale('log')
plt.xlabel('$f$ [Hz]')
plt.ylabel('$S_L(f)$')
plt.legend(loc = 3)
plt.ylim(0.0001,10)
plt.xlim(10**(-1), 10**(2))

plt.savefig(file + 'LiftSpectra_Correlationline.png', bbox_inches='tight')
plt.show()

plt.figure()
plt.plot(f, SM_up_cl[:,index],label = 'Upstream', linewidth = 1)
plt.plot(f, SM_down_cl[:,index],  label =  'Downstream', linewidth = 1)
plt.plot(f, SM_tot_cl[:,index], label = 'Total',linewidth = 1)
plt.title('Moment Spectra - Correlation line ' + str(cl), fontsize = 12)

plt.yscale('log')
plt.xscale('log')
plt.xlabel('$f$ [Hz]')
plt.ylabel('$S_L(f)$')
plt.legend(loc = 3)
plt.ylim(10**(-6),10)
plt.xlim(10**(-1), 10**(2))

plt.savefig(file + 'MomentSpectra_Correlationline.png', bbox_inches='tight')
plt.show()

#%% Admittance Functions


Nwelch = 20 #Number of divisions
Nwindow = np.round(len(t)/Nwelch) #Length of window
nfft = 2**math.log2(Nwindow) #Number of FFT points
window = signal.windows.hamming(int(Nwindow))
```

```
#Turbulence
u = u - np.mean(u)
w = w - np.mean(w)

#Turbulence spectrum
f, Su = signal.welch(u, F, window = window, nfft = nfft)
f, Sw = signal.welch(w, F, window = window, nfft = nfft)
f, Swu = signal.csd(w, u, F,window = window, nfft = nfft)
f, Suw = signal.csd(u, w, F, window = window, nfft = nfft)

plt.figure()
plt.plot(f, Su, label = 'Su', linewidth = 0.8)
plt.plot(f, Sw, label = 'Sw', linewidth = 0.8)
plt.title('Turbulence spectra')
plt.legend(loc = 'best')
plt.yscale('log')
plt.xscale('log')
plt.xlabel('$f$ [Hz]')
plt.ylabel('$S_L(f)$')
plt.savefig(file + 'TurbulenceSpectra.png', bbox_inches='tight')
plt.show()

#%%

#Cross spectrum - Total
f, SDu = signal.csd(Drag_tot, u, F, window = window, nfft = nfft)
f, SLu = signal.csd(Lift_tot, u, F, window = window, nfft = nfft)
f, SMu = signal.csd(Moment_tot, u, F, window = window, nfft = nfft)

f, SDw = signal.csd(Drag_tot, w, F, window = window, nfft = nfft)
f, SLw = signal.csd(Lift_tot, w, F, window = window, nfft = nfft)
f, SMw = signal.csd(Moment_tot, w, F, window = window, nfft = nfft)

#Cross spectrum - Upstream
f, SDu_up = signal.csd(Drag_up, u, F, window = window, nfft = nfft)
f, SLu_up = signal.csd(Lift_up, u, F, window = window, nfft = nfft)
f, SMu_up = signal.csd(Moment_up, u, F, window = window, nfft = nfft)

f, SDw_up = signal.csd(Drag_up, w, F, window = window, nfft = nfft)
f, SLw_up = signal.csd(Lift_up, w, F, window = window, nfft = nfft)
f, SMw_up = signal.csd(Moment_up, w, F, window = window, nfft = nfft)

#Cross spectrum - Downstream
f, SDu_down = signal.csd(Drag_down, u, F, window = window, nfft = nfft)
f, SLu_down  = signal.csd(Lift_down, u, F, window = window, nfft = nfft)
f, SMu_down  = signal.csd(Moment_down, u, F, window = window, nfft = nfft)

f, SDw_down = signal.csd(Drag_down, w, F, window = window, nfft = nfft)
f, SLw_down = signal.csd(Lift_down, w, F, window = window, nfft = nfft)
f, SMw_down  = signal.csd(Moment_down, w, F, window = window, nfft = nfft)


#%% General admittance functions

Au_drag = SD_tot/Su
Aw_lift = SL_tot/Sw
Aw_moment = SM_tot/Sw

cm = 1/2.54
fig, axs = plt.subplots(1,3,figsize=(34*cm,12*cm))
```

```
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace=0.3, hspace=0.2)

axs[0].plot(f, Au_drag, label = '$\chi u - Drag$')
axs[0].set_xlabel('f [Hz]')
axs[0].set_ylabel('Admittance $[N^2/(m/s)^2]$')
axs[0].set_yscale('log')
axs[0].set_xscale('log')
axs[0].legend(loc = 2)
axs[0].set_xlim(1, 10**2)
axs[0].set_ylim(10**(-2), 10**5)

axs[1].plot(f, Aw_lift, label = '$\chi w - Lift$ ')
axs[1].set_xlabel('f [Hz]')
axs[1].set_ylabel('Admittance $[N^2/(m/s)^2]$')
axs[1].set_yscale('log')
axs[1].set_xscale('log')
axs[1].legend(loc = 2)
axs[1].set_xlim(1, 10**2)
axs[1].set_ylim(10**(-2), 10**5)
ttl = axs[1].set_title("General Admittance Functions", fontsize = 14)
ttl.set_position([0.5, 1.1])


axs[2].plot(f, Aw_moment, label = '$\chi w - Moment$')
axs[2].set_xlabel('f [Hz]')
axs[2].set_ylabel('Admittance $[N^2/(m/s)^2]$')
axs[2].set_yscale('log')
axs[2].set_xscale('log')
axs[2].legend(loc = 2)
axs[2].set_xlim(1, 10**2)
axs[2].set_ylim(10**(-2), 10**5)

plt.tight_layout()
plt.savefig(file + 'GeneralAdmittance_7_Opengrid.png', bbox_inches='tight')

#%% Admittance functions - Total

#Static coefficients
Cd = 0.667
Cl = -0.1558
Cm = -0.0214
dCd = 2.002
dCl = -0.4673
dCm = -0.0642

#af
aD = 2*Cd
aL = 2*Cl
aM = 2*Cm

#bf
bD = (dCd-Cl)
bL = (dCl+Cd)
bM = dCm

# Sears function
fred = (f*B)/V
sears = np.zeros(len(f))

#Sears approximated function
```

```
for i in range(len(f)):
    sears[i] = 1/(1+2*np.pi**2*fred[i])

#AL
AL = SL_tot/((0.5*rho*B*V)**2*(aL**2*Su + bL**2*Sw))
ALw = (Su*SLw-Suw*SLu)/((0.5*rho*B*V)*(bL*(Su*Sw-Swu*Suw)))
ALu = (Sw*SLu-Suw*SLw)/((0.5*rho*B*V)*(aL*(Su*Sw-Swu*Suw)))

#AD
AD = SD_tot/((0.5*rho*B*V)**2*(aD**2*Su+bD**2*Sw))
ADw = (Su*SDw-Suw*SDu)/((0.5*rho*B*V)*(bD*(Su*Sw-Swu*Suw)))
ADu = (Sw*SDu-Suw*SDw)/((0.5*rho*B*V)*(aD*(Su*Sw-Swu*Suw)))

#AM
AM = SM_tot/((0.5*rho*B**2*V)**2*(aM**2*Su+bM**2*Sw))
AMw = (Su*SMw-Suw*SMu)/((0.5*rho*B**2*V)*(bM*(Su*Sw-Swu*Suw)))
AMu = (Sw*SMu-Suw*SMw)/((0.5*rho*B**2*V)*(aM*(Su*Sw-Swu*Suw)))

#Line 2-6
AL_cl = SL_tot_cl[:,0]/((0.5*rho*B*V)**2*(aL**2*Su+bL**2*Sw))
AD_cl = SD_tot_cl[:,0]/((0.5*rho*B*V)**2*(aD**2*Su+bD**2*Sw))
AM_cl = SM_tot_cl[:,0]/((0.5*rho*B**2*V)**2*(aM**2*Su+bM**2*Sw))

#Plot Admittance functions

#Drag
plt.figure()
plt.plot(fred, sears, label = 'Sears', linewidth = 0.6)
plt.plot(fred, AD, label = '$\chi_D$',linewidth = 0.6 )
plt.plot(fred, abs(ADw)**2, label = '$\chi_{Dw}$', linewidth = 0.6)
plt.plot(fred, abs(ADu)**2, label = '$\chi_{Du}$', linewidth = 0.6)
plt.xlabel('$f* = \dfrac{fB}{V}$')
plt.ylabel('$|\chi_i|^2$')
plt.xlim(00.1, 10)
plt.ylim(10**(-8), 10**(5))
plt.yscale('log')
plt.xscale('log')
plt.legend(loc = 2, labelspacing = 0.2)
plt.title('Admittance functions - Drag')
plt.tight_layout()
plt.savefig(test + 'Admittance_Drag_7_Opengrid.png', bbox_inches='tight')

#Lift
plt.figure()
plt.plot(fred, sears, label = 'Sears', linewidth = 0.6)
plt.plot(fred, AL, label = '$\chi_L$', linewidth = 0.6)
plt.plot(fred, abs(ALw)**2, label = '$\chi_{Lw}$', linewidth = 0.6)
plt.plot(fred, abs(ALu)**2, label = '$\chi_{Lu}$', linewidth = 0.6)
plt.xlabel('$f* = \dfrac{fB}{V}$')
plt.ylabel('$|\chi_i|^2$')
plt.legend(loc = 2, labelspacing = 0.2)
plt.xlim(0.1, 10)
plt.ylim(10**(-4), 10**(5))
plt.yscale('log')
plt.xscale('log')
plt.title('Admittance functions - Lift')
plt.tight_layout()
plt.savefig(test + 'Admittance_Lift_7_Opengrid.png', bbox_inches='tight')

#Moment
```

```
plt.figure()
plt.plot(fred, sears, label = 'Sears', linewidth = 0.6)
plt.plot(fred, AM, label = '$\chi_M$', linewidth = 0.6)
plt.plot(fred, abs(AMw)**2, label = '$\chi_{Mw}$', linewidth = 0.6)
plt.plot(fred, abs(AMu)**2, label = '$\chi_{Mu}$', linewidth = 0.6)
plt.xlabel('$f* = \dfrac{fB}{V}$')
plt.ylabel('$|\chi_i|^2$')
plt.legend(loc = 2, labelspacing = 0.2)
plt.xlim(0.1, 10)
plt.ylim(10**(-4),10**(5))
plt.xscale('log')
plt.yscale('log')
plt.title('Admittance functions - Moment')
plt.tight_layout()
plt.savefig(test + 'Admittance_Moment_7_Opengrid.png', bbox_inches='tight')


#%% Equivalent admittance functions - Upstream box

Cd_up = 0.2784
Cl_up = -0.1359
Cm_up = -0.03622
dCd_up = 0.8351
dCl_up = -0.4078
dCm_up = -0.1087

#af
aD_up = 2*Cd_up
aL_up = 2*Cl_up
aM_up = 2*Cm_up

#bf
bD_up = (dCd_up-Cl_up)
bL_up = (dCl_up+Cd_up)
bM_up = dCm_up

# Sears function
fred_up = (f*B)/V
Sears = 1/(1+2*np.pi**2*fred_up)

#AL
AL_up = SL_up/((0.5*rho*B*V)**2*(aL_up**2*Su + bL_up**2*Sw))
ALw_up = (Su*SLw_up-Suw*SLu_up)/((0.5*rho*B*V)*(bL_up*(Su*Sw-Swu*Suw)))
ALu_up = (Sw*SLu_up-Suw*SLw_up)/((0.5*rho*B*V)*(aL_up*(Su*Sw-Swu*Suw)))

#AD
AD_up = SD_up/((0.5*rho*B**2*V)**2*(aD_up**2*Su+bD_up**2*Sw))
ADw_up = (Su*SDw_up-Suw*SDu_up)/((0.5*rho*B*V)*(bD_up*(Su*Sw-Swu*Suw)))
ADu_up = (Sw*SDu_up-Suw*SDw_up)/((0.5*rho*B*V)*(aD_up*(Su*Sw-Swu*Suw)))

#AM
AM_up = SM_up/((0.5*rho*B**2*V)**2*(aM**2*Su+bM**2*Sw))
AMw_up = (Su*SMw_up-Suw*SMu_up)/((0.5*rho*B**2*V)*(bM_up*(Su*Sw-Swu*Suw)))
AMu_up = (Sw*SMu_up-Suw*SMw_up)/((0.5*rho*B**2*V)*(aM_up*(Su*Sw-Swu*Suw)))

#Plot Admittance functions

#Drag
plt.figure()
plt.plot(fred, Sears, label = 'Sears', linewidth = 1)
```

```
plt.plot(fred, AD_up, label = '$\chi_D$', linewidth = 1)
plt.plot(fred, abs(ADw)**2, label = '$\chi_{Dw}$', linewidth = 1)
plt.plot(fred, abs(ADu)**2, label = '$\chi_{Du}$', linewidth = 1)
plt.xlabel('$f* = \dfrac{fB}{V}$')
plt.ylabel('$|\chi_i|^2$')
plt.yscale('log')
plt.xscale('log')
plt.xlim(0.1, 10)
plt.ylim(10**(-8),10**(5))
plt.legend(loc = 2, labelspacing = 0.2)
plt.title('Admittance functions - Drag upstream')
plt.savefig(test + 'Admittance_Drag_upstream_7_Opengrid.png', bbox_inches='tight')
plt.show()

#Lift
plt.figure()
plt.plot(fred, Sears, label = 'Sears', linewidth = 1)
plt.plot(fred, AL_up, label = '$\chi_L$', linewidth = 1)
plt.plot(fred, abs(ALw_up)**2, label = '$\chi_{Lw}$', linewidth = 1)
plt.plot(fred, abs(ALu_up)**2, label = '$\chi_{Lu}$', linewidth = 1)
plt.xlabel('$f* = \dfrac{fB}{V}$')
plt.ylabel('$|\chi_i|^2$')
plt.legend(loc = 2, labelspacing = 0.2)
plt.yscale('log')
plt.xscale('log')
plt.xlim(0.1, 10)
plt.ylim(10**(-6),10**(5))
plt.title('Admittance functions - Lift upstream')
plt.savefig(test + 'Admittance_Lift_upstream_7_Opengrid.png', bbox_inches='tight')
plt.show()

#Moment
plt.figure()
plt.plot(fred, sears, label = 'Sears', linewidth = 1)
plt.plot(fred, AM_up, label = '$\chi_M$', linewidth = 1)
plt.plot(fred, abs(AMw_up)**2, label = '$\chi_{Mw}$', linewidth = 1)
plt.plot(fred, abs(AMu_up)**2, label = '$\chi_{Mu}$', linewidth = 1)
plt.xlabel('$f* = \dfrac{fB}{V}$')
plt.ylabel('$|\chi_i|^2$')
plt.legend(loc = 2, labelspacing = 0.2)
plt.xscale('log')
plt.yscale('log')
plt.xlim(0.1, 10)
plt.ylim(10**(-6),10**(5))
plt.title('Admittance functions - Moment upstream')
plt.savefig(test + 'Admittance_Moment_upstream_7_Opengrid.png', bbox_inches='tight')

#%% Equivalent admittance functions - Downstream box

Cd_d = 0.4049
Cl_d = -0.0269
Cm_d = 0.0133
dCd_d = 1.2148
dCl_d = -0.0806
dCm_d = 0.04

#af
aD_d = 2*Cd_d
aL_d = 2*Cl_d
aM_d = 2*Cm_d
```

```
#bf
bD_d = (dCd_d-Cl_d)
bL_d = (dCl_d+Cd_d)
bM_d = dCm_d

# Sears function
fred_d = (f*B)/V
Sears= 1/(1+2*np.pi**2*fred_d)

#AL
AL_d = SL_down/((0.5*rho*B*V)**2*(aL_d**2*Su + bL_d**2*Sw))
ALw_d = (Su*SLw_down-Suw*SLu_down)/((0.5*rho*B*V)*(bL_d*(Su*Sw-Swu*Suw)))
ALu_d = (Sw*SLu_down-Suw*SLw_down)/((0.5*rho*B*V)*(aL_d*(Su*Sw-Swu*Suw)))

#AD
AD_d = SD_down/((0.5*rho*B**2*V)**2*(aD_d**2*Su+bD_d**2*Sw))
ADw_d = (Su*SDw_down-Suw*SDu_down)/((0.5*rho*B*V)*(bD_d*(Su*Sw-Swu*Suw)))
ADu_d = (Sw*SDu_down-Suw*SDw_down)/((0.5*rho*B*V)*(aD_d*(Su*Sw-Swu*Suw)))

#AM
AM_d = SM_down/((0.5*rho*B*V)**2*(aM**2*Su+bM**2*Sw))
AMw_d = (Su*SMw_down-Suw*SMu_down)/((0.5*rho*B**2*V)*(bM_d*(Su*Sw-Swu*Suw)))
AMu_d = (Sw*SMu_down-Suw*SMw_down)/((0.5*rho*B**2*V)*(aM_d*(Su*Sw-Swu*Suw)))

#Plot Admittance functions

#Drag
plt.figure()
plt.plot(fred, sears, label = 'Sears', linewidth = 1)
plt.plot(fred, AD_d, label = '$\chi_D$', linewidth = 1)
plt.plot(fred, abs(ADw_d)**2, label = '$\chi_{Dw}$', linewidth = 1)
plt.plot(fred, abs(ADu_d)**2, label = '$\chi_{Du}$', linewidth = 1)
plt.xlabel('$f* = \dfrac{fB}{V}$')
plt.ylabel('$|\chi_i|^2$')
plt.yscale('log')
plt.xscale('log')
plt.xlim(0.1, 10)
plt.ylim(10**(-8),10**(5))
plt.legend(loc = 2, labelspacing = 0.2)
plt.title('Admittance functions - Drag downstream')
plt.savefig(test + 'Admittance_Drag_downstream_7_Opengrid.png', bbox_inches='tight')
plt.show()

#Lift
plt.figure()
plt.plot(fred, sears, label = 'Sears', linewidth = 1)
plt.plot(fred, AL_d, label = '$\chi_L$', linewidth = 1)
plt.plot(fred, abs(ALw_d)**2, label = '$\chi_{Lw}$', linewidth = 1)
plt.plot(fred, abs(ALu_d)**2, label = '$\chi_{Lu}$', linewidth = 1)
plt.xlabel('$f* = \dfrac{fB}{V}$')
plt.ylabel('$|\chi_i|^2$')
plt.legend(loc = 2, labelspacing = 0.2)
plt.yscale('log')
plt.xscale('log')
plt.xlim(0.1, 10)
plt.ylim(10**(-6),10**(5))
plt.title('Admittance functions - Lift downstream')
plt.savefig(test + 'Admittance_Lift_downstream_7_Opengrid.png', bbox_inches='tight')
plt.show()
```

```
#Moment
plt.figure()
plt.plot(fred, sears, label = 'Sears', linewidth = 1)
plt.plot(fred, AM_d, label = '$\chi_M$', linewidth = 1)
plt.plot(fred, abs(AMw_d)**2, label = '$\chi_{Mw}$', linewidth = 1)
plt.plot(fred, abs(AMu_d)**2, label = '$\chi_{Mu}$', linewidth = 1)
plt.xlabel('$f* = \dfrac{fB}{V}$')
plt.ylabel('$|\chi_i|^2$')
plt.legend(loc = 2, labelspacing = 0.2)
plt.yscale('log')
plt.xscale('log')
plt.xlim(0.1, 10)
plt.ylim(10**(-6),10**(5))
plt.title('Admittance functions - Moment downstream')
plt.savefig(test + 'Admittance_Moment_downstream_7_Opengrid.png', bbox_inches='tight')
plt.show()


#%% Coherence
cl = 6 #Correlation line
index = cl-2

#Drag
#Total
f, S_D_tot =  signal.csd(Drag_tot, Drag_tot_cl[:,index], F, window = window, nfft = nfft)
Coh_D_tot = abs(S_D_tot)**2/(SD_tot*SD_tot_cl[:,index])
#Upstream
f, S_D_up =  signal.csd(Drag_up, Drag_up_cl[:,index], F, window = window, nfft = nfft)
Coh_D_up = abs(S_D_up)**2/(SD_up*SD_up_cl[:,index])
#Downstream
f, S_D_down =  signal.csd(Drag_down, Drag_down_cl[:,index], F, window = window, nfft = nfft)
Coh_D_down = abs(S_D_down)**2/(SD_down*SD_down_cl[:,index])

#Lift
#Total
f, S_L_tot =  signal.csd(Lift_tot, Lift_tot_cl[:,index], F, window = window, nfft = nfft)
Coh_L_tot = abs(S_L_tot)**2/(SL_tot*SL_tot_cl[:,index])
#Upstream
f, S_L_up =  signal.csd(Lift_up, Lift_up_cl[:,index], F, window = window, nfft = nfft)
Coh_L_up = abs(S_L_up)**2/(SL_up*SL_up_cl[:,index])
#Downstream
f, S_L_down =  signal.csd(Lift_down, Lift_down_cl[:,index], F, window = window, nfft = nfft)
Coh_L_down = abs(S_L_down)**2/(SL_down*SL_down_cl[:,index])

#Moment
#Total
f, S_M_tot =  signal.csd(Moment_tot, Moment_tot_cl[:,index], F, window = window, nfft = nfft)
Coh_M_tot = abs(S_M_tot)**2/(SM_tot*SM_tot_cl[:,index])
#Upstream
f, S_M_up =  signal.csd(Moment_up, Moment_up_cl[:,index], F, window = window, nfft = nfft)
Coh_M_up = abs(S_M_up)**2/(SM_up*SM_up_cl[:,index])
#Downstream
f, S_M_down =  signal.csd(Moment_down, Moment_down_cl[:,index], F, window = window, nfft = nfft)
Coh_M_down = abs(S_M_down)**2/(SM_down*SM_down_cl[:,index])


#Polyfit -  total coherence
fit_d = poly.polyfit(f, Coh_D_tot, deg = 5)
fit_coh_d = poly.Polynomial(fit_d)(f)
```

```
fit_l = poly.polyfit(f, Coh_L_tot, deg = 5)
fit_coh_l= poly.Polynomial(fit_l)(f)

fit_m = poly.polyfit(f, Coh_M_tot, deg = 5)
fit_coh_m = poly.Polynomial(fit_m)(f)

y = [10,25,55,105,220] #distances between correlation lines, [mm]

#Plot coherence
cm = 1/2.54
fig, axs = plt.subplots(1,3,figsize=(38*cm,8*cm))
plt.subplots_adjust(left=0.125, bottom=0.1, right=0.9, top=0.9, wspace=0.3, hspace=0.2)

axs[0].plot(f,Coh_D_up, label = 'Upstream', linewidth = 0.8)
axs[0].plot(f, Coh_D_down, label = 'Downstream', linewidth = 0.8)
axs[0].plot(f,Coh_D_tot, label = 'Total', linewidth = 0.8)
axs[0].plot(f, fit_coh_d, label = 'Total - Fitted', color = 'darkgreen' )
axs[0].set_ylim(0,1)
axs[0].set_xlim(0,100)
axs[0].set_ylabel('$Coh_D$')
axs[0].set_xlabel('f [Hz]')
axs[0].set_title('Drag-coherence, $\Delta y =  $'+ str(y[index]) + 'mm')

axs[1].plot(f,Coh_L_up, label = 'Upstream', linewidth = 0.8)
axs[1].plot(f, Coh_L_down, label = 'Downstream', linewidth = 0.8)
axs[1].plot(f,Coh_L_tot, label = 'Total', linewidth = 0.8)
axs[1].plot(f, fit_coh_l, label = 'Total - Fitted curve', color = 'darkgreen' )
axs[1].set_ylim(0,1)
axs[1].set_xlim(0,100)
axs[1].set_ylabel('$Coh_L$')
axs[1].set_xlabel('f [Hz]')
axs[1].set_title('Lift-coherence, $\Delta y =  $'+ str(y[index]) +'mm')

axs[2].plot(f,Coh_M_up, label = 'Upstream', linewidth = 0.8)
axs[2].plot(f, Coh_M_down, label = 'Downstream', linewidth = 0.8)
axs[2].plot(f,Coh_M_tot, label = 'Total', linewidth = 0.8)
axs[2].plot(f, fit_coh_m, label = 'Fitted curve', color = 'darkgreen' )
axs[2].set_ylim(0,1)
axs[2].set_xlim(0,100)
axs[2].set_ylabel('$Coh_M$')
axs[2].set_xlabel('f [Hz]')
axs[2].set_title('Moment-coherence,  $\Delta y =  $'+ str(y[index])+'mm')
axs[2].legend(loc = 1)


plt.savefig('New_Coherence_9_7Hz_line'+ str(cl), bbox_inches='tight')
```

C.1.1 Functions for Importing Processed Matlab Data

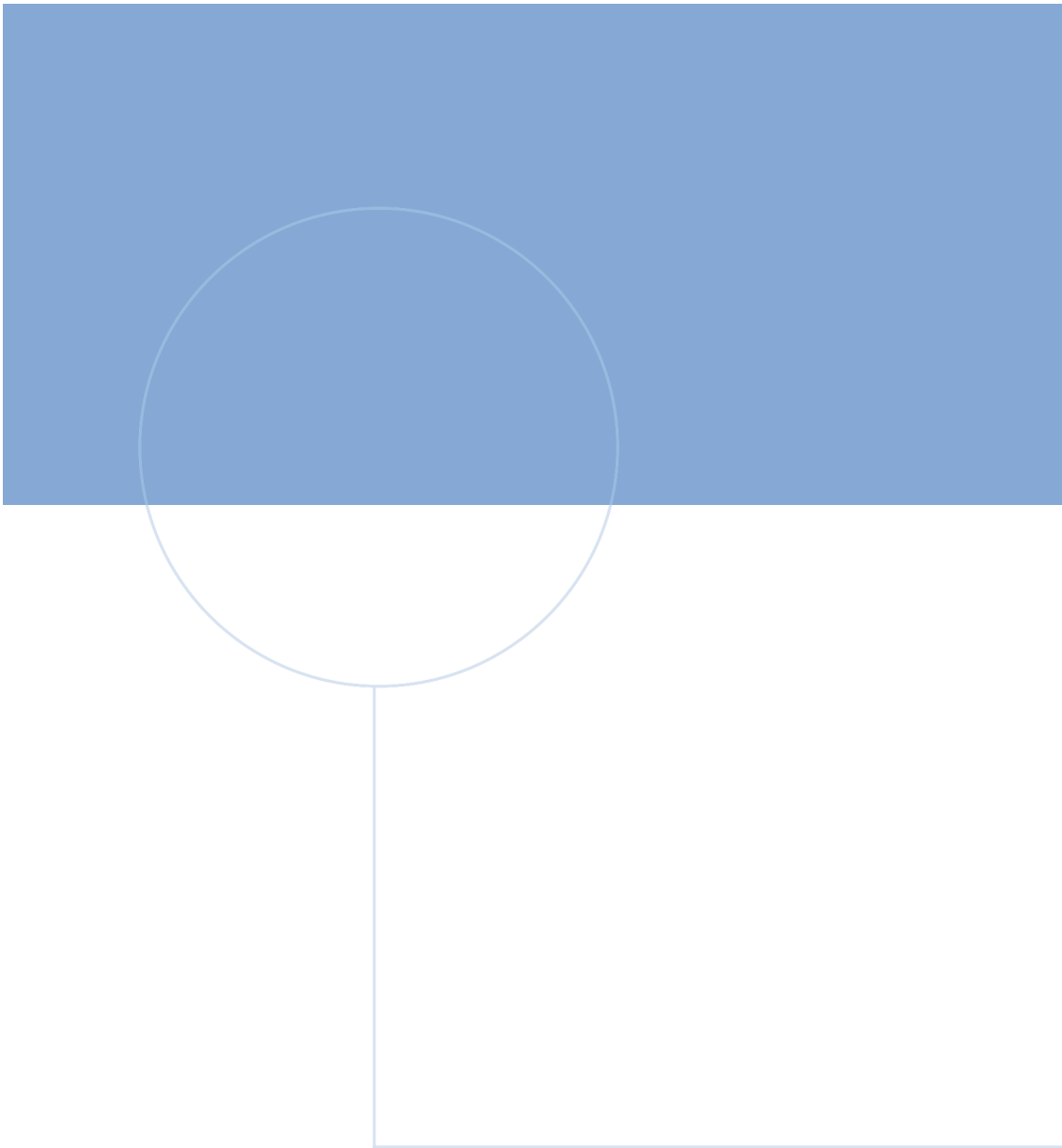
Function for importing the processed data from Matlab to Python.

```python
# -*- coding: utf-8 -*-
"""
Created on Sun May 15 09:23:11 2022

@author: marth
"""

import scipy.io
import numpy as np

def ImportMeasuredData(filename):
    """
    Parameters
    -----------
    filename : Matlab filename with processed data

    Returns
    -----------
    Pitotprobe : Velcity measured by Pitot Probes [m/s]
    Displacements : Measured ux[m], uz[m] and u_theta[rad]
    Forces_global : Forces in global coordinates in all four load celles [N, N, N, Nm, Nm, Nm] x4
    Forces_cr : Forces in global coordinates for all load cells with center of rotation as reference position [N, N, N, Nm, Nm, Nm] x4
    Cobra : Cobra probe wind velocity in three directions u[m/s], v[m/s], w[m/s] and pressure[Pa]
    Temperature : Temperature in celsius
    Frequency : Logging frequency in Hz
    AirDensity : Air density in kg/m^3

    """
    mat = scipy.io.loadmat('C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/Python/ProcessedMatFiles/' + filename)
    NTNUWT = mat.get('NTNUWT')

    Data = NTNUWT[0][0][4]
    ProcessedData = Data[0][0][2]
    Root = Data[0][0][0]

    Pitotprobe = ProcessedData[0][0][1]
    Displacements = ProcessedData[0][1][1]
    Forces_global = ProcessedData[0][2][1]
    Forces_cr = ProcessedData[0][3][1]

    Temperature = Root[0][0][1][0][11][1][0][0][0]
    Temperature = Temperature.split('Â')
    Temperature = float(Temperature[0])

    Frequency = Root[0][0][1][0][4][1][0][0][0]
    Frequency = Frequency.split()
    Frequency = Frequency[0]
    Frequency = int(Frequency.replace(',','.'))

    AirDensity = Root[0][0][1][0][9][1][0][0][0]
    AirDensity = AirDensity.split()
    AirDensity = AirDensity[0]
    AirDensity = float(AirDensity.replace(',','.'))

    #Cobra Probe
    f = filename.split('AGTD21_S2-G1_Pressure_CPD105_')
    f = 'Cobra_AGTD21_S2-G1_Pressure_' + f[1]
    file = scipy.io.loadmat('C:/Users/marth/OneDrive - NTNU/Masteroppgave våren 2022/Python/Cobra_AGTD21_S2-G1_all/' + f)
    cobra = file.get('cobra')

    u = cobra[0][0][5]
    v = cobra[0][0][6]
    w = cobra[0][0][7]
    t = cobra[0][0][9]

    u = u.mean(axis = 1)
    v = v.mean(axis = 1)
    w = w.mean(axis = 1)

    Cobra = np.array([u[:63800],v[:63800],w[:63800]])

    return Pitotprobe, Displacements, Forces_global, Forces_cr, Temperature, Frequency, AirDensity, Cobra, t
```

C.1.2 Static load coefficients

Function for calculation of the static coefficients.

```python
# -*- coding: utf-8 -*-
"""
Created on Thu May 19 15:02:18 2022

@author: marth
"""

def StaticCoeffs(Fd, Fl, Fm, rho, V):
    """

    Parameters
    ----------
    Fd : Drag force
    Fl : Lift force
    Fm : Moment force
    rho : Air Density
    V : Wind velocity

    Returns
    -------
    CD : Static drag coefficient
    CL : Static lift coefficient
    CM : Static moment coefficient

    """

    D = 0.05 #Height [m]
    B = 0.74 #Width [m]
    L = 2.64 #Length [m]

    CD = (Fd/D)/(0.5*rho*V**2*L)
    CL = (Fl/B)/(0.5*rho*V**2*L)
    CM = (Fm/B**2)/(0.5*rho*V**2*L)

    return CD, CL, CM
```



NTNU

Norwegian University of
Science and Technology