Multilevel Panel Method Validation Using the New MEXICO Wind Tunnel Measurements

Erik Prieto S.
M.Sc. Thesis
November 2017
To my father, my mother and my brother with all my love.
I would like to thank first to my supervisors Arne van Garrel and Kees Venner, for the enlightenment through this work, their teaching and ideas fed my curiosity for the fluid dynamic field. Their sharp comments and optimistic advice were the fuel and the fire that helped me to get which I desire.

Special thanks to Arne van Garrel who devoted their time and patience unconditionally and punctually every Thursday morning through all the thesis, with talks about the future of the world without sustainability and renewable energies, who also teach me the multilevel panel method spiny details, for sharing his experience in the aerodynamic field, and finally for offering me his honest friendship.

My total gratefulness to Kees Venner for accepting me as part of his research group, for helping me in the darkest moments of my master, who provided me with all the necessary tools to develop the knowledge by my self, who take the time to care about his students and finally for offered me the opportunity of doing my internship at GE, where I had an incredible and rich experience.

My gratitude to the committee for this examination for taking the time to read my work and give their comments, my two supervisors, prof.dr.ir. C.H. Venner, and dr.ir. A. van Garrel. the prof. ir. Theo van der Meer and from ECN the Dr. ir, Koen Boorsma.

My thanks to Saskia Honhoff, who was my supervisor at GE and who make my stay during the short period of time a wonderful experience, not just as a critical supervisor but as a friend and a colleague.

I want to express my gratuity to the Science and Technology Council of Mexico (Consejo Nacional de Ciencia y Tecnología, CONACYT) that has granted me with the scholarship throughout my Master of Science Program, and all the people there that helped me with all my academic and financial issues.

None of this would be possible without my dear friend Leonel Reyes who encourage me to pursue my master degree, his advice, reprimanding, his share of experiences and faith on my work have helped me in the most difficult moments.

During my master, I have the privilege to get to know many wonderful persons. I am indebted to my friend Lorena for several discussions that helped me to be not just a robot but also a human being. Also, I want to thank all my SET friends for giving their opinions and comments on other
field and point of view.

It must also be acknowledged all the people in México who has maintained the time to care about my studies. My debt to the Dr Miguel de Icaza Herrera, who is not just the best physic professor I ever had, but also an endearing friend. Also to my dear Mar who has always believed in me. I thank all my friends around the world, who has made this experience a little bit less scary.

Last but not the least, I would like to thank my family: my parents Angel Prieto and Micaela Serratos, and my brother Ovidio for giving not just economical support but also supporting me spiritually throughout my life.
Summary

The human activities nowadays have a huge impact in the world. The best known consequence is “climate change” that is heavily related to our excessive consumption of natural resources. This provokes numerous problems like the production of green house gases that trap heat in the atmosphere and change the natural climate conditions on earth. As the dominant species on the planet, there are a lot of solutions that we can implement to fight climate change. Solutions like population reduction through sexual education, smart food production practices, better energy policies, recycling, circular economies, and renewable energy production are some examples.

Increasing the use of renewable energies such as solar, biomass and wind power is a relevant and important strategy nowadays for combating climate change due the rising energy consumption in the world. There is no single solution to climate change but rather a combination of several solutions is necessary. Particularly in the energy field interesting solutions can be used. Among all the energy solutions, wind energy is catalogued as the second most influential solution to reduce CO2 emissions just after “Refrigerant Management”. Wind energy keeps growing and new developments are carried out constantly in all areas. Several developments are related to the correct prediction of aerodynamic phenomena that are present in the extraction of energy from the wind.

Developments in the aerodynamic theory help to predict the forces acting on the wind turbine and improve the power production of the turbines. The blade element momentum (BEM) theory is one of the earliest aerodynamic theories for wind turbines and is still in use nowadays. The more advanced methods are based on Reynolds-Averaged Navier-Stokes (RANS) equations and make use of turbulent models or even resort to Large Eddy Simulation (LES) [1].

Some of these methods constitute what is known as Computation Fluid Dynamics (CFD), a field that has a wide range of methods to solve fluid flow problems. Generally these methods involve complicated mathematical and numerical approaches and while being accurate their application is computationally expensive and too slow for use in day to day design practice.

In this thesis the validation of a so-called ‘Multilevel Panel Method’ is performed. The panel method is an attractive flow analysis technique that, in contrast to other CFD methods, is computationally efficient and still has fundamental physics involved to work out the solutions. The validation is performed using the results from a European wind tunnel experiment called “New MEXICO”. We demonstrate the fidelity of the advanced multilevel panel method under unsteady conditions. We consider cases with rotor yaw at angles $-30^\circ$, $15^\circ$, $30^\circ$ and $45^\circ$, and two types of step changes:
steps in rotational velocity and steps in rotor blade pitch angle.

This thesis concludes by giving some recommendations for further validations and the use of the state-of-the-art multilevel panel method.
x
Contents

Acknowledgements ....................................... V
Summary ................................................... VII
Contents ................................................ XI
Nomenclature ............................................ XIX

1 Introduction ........................................... 1
   1.1 Context ........................................... 1
       1.1.1 Climate Change and Global Warming .............. 1
       1.1.2 Mitigating the Effects of Human Behaviour ........ 5
       1.1.3 Wind Energy and Wind Turbines ................... 6
       1.1.4 Aerodynamic Theories on Wind Energy ............ 8
   1.2 Problem Statement ................................ 10
   1.3 Objectives ....................................... 10
   1.4 Approach ......................................... 10
   1.5 Thesis outline .................................... 11

2 Multilevel Panel Method ............................... 13
   2.1 Introduction ..................................... 13
   2.2 Governing Equations ............................... 14
       2.2.1 Boundary Integral Equation ....................... 15
       2.2.2 Boundary Conditions ............................ 16
           2.2.2.1 Body Surface ............................. 16
       2.2.3 Wake Surface ................................ 18
       2.2.4 Pressure Coefficient Definition ................ 19
       2.2.5 Panel Method ................................ 20
   2.3 Multi Level Multi Integration Cluster Scheme ......... 22

3 New MEXICO Wind Tunnel Experiment ................. 25
   3.1 Introduction ..................................... 25
   3.2 Definition and Conventions ....................... 26
   3.3 Wind Turbine .................................... 28

XI
List of Figures

1.1 Hockey stick graph ............................................. 3
1.2 Hurricane Irma ................................................. 4
1.3 Three hurricanes ................................................. 5
1.4 Wind turbines ..................................................... 6
1.5 World wind capacity ............................................ 8
1.6 Big turbines ....................................................... 9

2.1 Flow domain ....................................................... 15
2.2 Panel and node ordering ......................................... 21
2.3 Panels over blade. ................................................. 21
2.4 Multi level concept .............................................. 22

3.1 Azimuth angle, yaw, and coordinate system definitions [63]. ...................... 27
3.2 Pitch angle definition. ............................................ 27
3.3 Airfoil sections .................................................... 29
3.4 Blade composition shape from [63]. ................................ 29
3.5 Schematic view of LLF tunnel .................................. 30

4.1 Azimuth averaging locations ................................... 38
4.2 Rotor geometry and wake ....................................... 40

XIII
4.3 Velocity triangle for non dimenzionalisation ........................................ 41
4.4 Effects of Non-dimensionalization in $C_p$ ........................................... 43
4.5 15° Yaw, azimuth angle 0° ............................................................. 46
4.6 15° Yaw, azimuth angle 60° ............................................................. 47
4.7 15° Yaw, azimuth angle 120° ........................................................... 48
4.8 15° Yaw, azimuth angle 180° ........................................................... 49
4.9 15° Yaw, azimuth angle 240° ........................................................... 50
4.10 15° Yaw, azimuth angle 300° .......................................................... 51
4.11 30° Yaw, azimuth angle 0° .............................................................. 53
4.12 30° Yaw, azimuth angle 60° .............................................................. 54
4.13 30° Yaw, azimuth angle 120° .......................................................... 55
4.14 30° Yaw, azimuth angle 180° .......................................................... 56
4.15 30° Yaw, azimuth angle 240° .......................................................... 57
4.16 30° Yaw, azimuth angle 300° .......................................................... 58
4.17 −30° Yaw, azimuth angle 0° ............................................................. 60
4.18 −30° Yaw, azimuth angle 60° ............................................................. 61
4.19 −30° Yaw, azimuth angle 120° ........................................................ 62
4.20 −30° Yaw, azimuth angle 180° ........................................................ 63
4.21 −30° Yaw, azimuth angle 240° ........................................................ 64
4.22 −30° Yaw, azimuth angle 300° ........................................................ 65
4.23 45° Yaw, azimuth angle 0° .............................................................. 67
4.24 45° Yaw, azimuth angle 60° .............................................................. 68
4.25 45° Yaw, azimuth angle 120° ........................................................... 69
4.26 45° Yaw, azimuth angle 180° ........................................................... 70
4.27 45° Yaw, azimuth angle 240° ........................................................... 71

XIV
4.28 45° Yaw, azimuth angle 300° ........................................ 72
4.29 Pitch step plot ......................................................... 73
4.30 Pitch $t_1 = 4.65s$ .................................................... 75
4.31 Pitch $t_2 = 5.20s$ .................................................... 76
4.32 Pitch $t_3 = 10.85s$ ................................................... 77
4.33 Pitch $t_4 = 11.25s$ ................................................... 78
4.34 RPM step plot ......................................................... 80
4.35 RPM integration ....................................................... 82
4.36 Velocity triangle changes for 424 rpm and 324 rpm. ............... 83
4.37 Pitch $t_1 = 3.63s$ .................................................... 84
4.38 Pitch $t_2 = 5.68s$ .................................................... 85
4.39 Pitch $t_3 = 9.67s$ .................................................... 86
4.40 Pitch $t_4 = 11.88s$ ................................................... 87
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Increase factor of human activities during the 20th century.</td>
<td>2</td>
</tr>
<tr>
<td>3.1</td>
<td>General turbine information, modified from [63].</td>
<td>28</td>
</tr>
<tr>
<td>3.2</td>
<td>Position of pressure sensors at blade and radial location</td>
<td>31</td>
</tr>
<tr>
<td>3.3</td>
<td>Estimated pressure coefficient due to uncertainty in the pressure sensors</td>
<td>32</td>
</tr>
<tr>
<td>3.4</td>
<td>Correlation of data for cases used for the validation.</td>
<td>34</td>
</tr>
<tr>
<td>B1</td>
<td>Model configuration legends for the New MEXICO</td>
<td>164</td>
</tr>
<tr>
<td>B2</td>
<td>Correlation of data for cases and files of the New MEXICO experiment</td>
<td>165</td>
</tr>
<tr>
<td>C1</td>
<td>Specifications of Kulite® XCQ-95-062-5A.</td>
<td>166</td>
</tr>
<tr>
<td>D1</td>
<td>Example of file generated with Matlab routines.</td>
<td>167</td>
</tr>
</tbody>
</table>
Nomenclature

Dimensions

[kg] kilogram mass
[m] meter length
[s] second time

[rad] radian angle [m·m⁻¹]

[−] dimensionless

Acronyms

AoA Angle of Attack
BC Boundary Condition
BEM Blade Element Momentum
CFD Computational Fluid Dynamics
DNW German-Dutch Wind Tunnel Organisation
ECN Energy research Centre of the Netherlands
LES Large Eddy Simulation
LLF Large Scale Low-Speed Facility
MG Multigrid
MEXICO Model Experiments in Controlled Conditions
MLMIC Multi Level Multi Integration Cluster
RaNS Reynolds-averaged Navier-Stokes
RPM Revolutions Per Minute
VII Viscous-Inviscid Interaction
### Roman symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_p$</td>
<td>[-]</td>
<td>pressure coefficient</td>
</tr>
<tr>
<td>$c$</td>
<td>[m]</td>
<td>chord length</td>
</tr>
<tr>
<td>$i, j$</td>
<td>[-]</td>
<td>index numbers</td>
</tr>
<tr>
<td>$L$</td>
<td>[m]</td>
<td>length</td>
</tr>
<tr>
<td>$Ma$</td>
<td>[-]</td>
<td>Mach number</td>
</tr>
<tr>
<td>$N$</td>
<td>[-]</td>
<td>problem size</td>
</tr>
<tr>
<td>$n$</td>
<td>[-]</td>
<td>number of elements</td>
</tr>
<tr>
<td>$P$</td>
<td>[W]</td>
<td>power</td>
</tr>
<tr>
<td>$p$</td>
<td>[N·m$^{-2}$]</td>
<td>pressure</td>
</tr>
<tr>
<td>$Re$</td>
<td>[-]</td>
<td>Reynolds number</td>
</tr>
<tr>
<td>$t$</td>
<td>[s]</td>
<td>time</td>
</tr>
<tr>
<td>$U$</td>
<td>[m·s$^{-1}$]</td>
<td>velocity</td>
</tr>
<tr>
<td>$v$</td>
<td>[m·s$^{-1}$]</td>
<td>velocity</td>
</tr>
<tr>
<td>$\partial S$</td>
<td>[m]</td>
<td>surface boundary</td>
</tr>
<tr>
<td>$\partial V$</td>
<td>[m$^2$]</td>
<td>volume boundary</td>
</tr>
<tr>
<td>$x, y, z$</td>
<td>[m]</td>
<td>Cartesian coordinates</td>
</tr>
</tbody>
</table>

### Greek symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>[rad]</td>
<td>angle of attack</td>
</tr>
<tr>
<td>$\delta, \Delta$</td>
<td>[m]</td>
<td>length scales</td>
</tr>
<tr>
<td>$\Delta$</td>
<td>[-]</td>
<td>difference</td>
</tr>
<tr>
<td>$\Phi$</td>
<td>[m$^2$·s$^{-1}$]</td>
<td>velocity potential</td>
</tr>
<tr>
<td>$\varphi$</td>
<td>[m$^2$·s$^{-1}$]</td>
<td>velocity perturbation potential</td>
</tr>
<tr>
<td>$\Gamma$</td>
<td>[m$^2$·s$^{-1}$]</td>
<td>vortex strength</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>[-]</td>
<td>local speed ratio, scaling factor, wing aspect ratio</td>
</tr>
<tr>
<td>$\mu$</td>
<td>[kg·m$^{-1}$·s$^{-1}$]</td>
<td>dynamic viscosity coefficient</td>
</tr>
<tr>
<td>$\mu_0$</td>
<td>[m$^2$·s$^{-1}$]</td>
<td>dipole strength</td>
</tr>
<tr>
<td>$\nu$</td>
<td>[m$^2$·s$^{-1}$]</td>
<td>kinematic viscosity</td>
</tr>
<tr>
<td>$\Omega$</td>
<td>[⋯]</td>
<td>space</td>
</tr>
<tr>
<td>$\Psi$</td>
<td>[⋯]</td>
<td>scalar field</td>
</tr>
<tr>
<td>$\Psi_\Phi$</td>
<td>[°]</td>
<td>azimuth angle location</td>
</tr>
<tr>
<td>$\psi$</td>
<td>[⋯]</td>
<td>scalar field</td>
</tr>
<tr>
<td>$\rho$</td>
<td>[kg·m$^3$]</td>
<td>mass density</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>[m·s$^{-1}$]</td>
<td>source strength</td>
</tr>
<tr>
<td>$\theta$</td>
<td>[rad]</td>
<td>rotation angle</td>
</tr>
<tr>
<td>$\chi$</td>
<td>[°]</td>
<td>amount of degrees travelled by a blade.</td>
</tr>
</tbody>
</table>
Vectors, matrices, and tensors

\( \vec{\Omega} \) [rad \cdot s^{-1}] \quad \text{angular velocity vector}

\( \vec{\omega}_v \) [s^{-1}] \quad \text{volume vorticity distribution}

\( \vec{u} \) [m\cdot s^{-1}] \quad \text{velocity vector}

\( \vec{x} \) [m] \quad \text{point location}

\( \vec{y} \) [m] \quad \text{point location}
Chapter 1

Introduction

You who read me, are you sure of understanding my language?

Jorge Luis Borges, The Library of Babel

1.1 Context

1.1.1 Climate Change and Global Warming

Human activities have grown in the last three centuries at impressive rates, world population has grown by a factor of four [2], the industrial output grew by a factor of 40 [2]. This was accompanied by an increase in energy use by a factor of 16 just during the 20th century alone. As consequence nearly 66% of atmospheric and ocean rise temperature can be confidently attributed to human activities [3]. The radical example of the decreasing of the blue whale population by 99.75% is one of many examples of how human activities exert an increasing impact on the environment on all scales [2]. Some other impacts of human activities during the 20th century can be seen on the Table 1.1.

The most known consequence of the human activities is “climate change”. The causes are heavily related to the human activities [4-6], from unprecedented warming [7-10] to heavy snowfalls like the North American blizzard that occurred on the 5-6 of February 2010 [11]. Although some major climatic events happened before [6], it seems that nowadays the main drivers are us humans ourselves. The addiction to the consumption of natural resources like the burning of coal and oil produces so called “green house gases” (GHG) [12], gases that trap heat in the atmosphere, thus changing the natural climate conditions on earth.

Carbon dioxide $CO_2$ and methane $CH_4$ are two harmful greenhouse gases [13–15]. The greenhouse gas $CO_2$ expelled into the atmosphere through burning fossil fuels like coal, natural gas and oil.
CHAPTER 1. INTRODUCTION

Table 1.1: A partial record of the growths and impacts of human activities during the 20th century, from [2]

<table>
<thead>
<tr>
<th>Item</th>
<th>Increase Factor, 1890s-1990s</th>
</tr>
</thead>
<tbody>
<tr>
<td>World population</td>
<td>4</td>
</tr>
<tr>
<td>Total world urban population</td>
<td>13</td>
</tr>
<tr>
<td>World economy</td>
<td>14</td>
</tr>
<tr>
<td>Industrial output</td>
<td>40</td>
</tr>
<tr>
<td>Energy use</td>
<td>16</td>
</tr>
<tr>
<td>Coal production</td>
<td>7</td>
</tr>
<tr>
<td>Carbon dioxide emissions</td>
<td>17</td>
</tr>
<tr>
<td>Sulphur dioxide emissions</td>
<td>13</td>
</tr>
<tr>
<td>Lead emissions</td>
<td>≈ 8</td>
</tr>
<tr>
<td>Water use</td>
<td>9</td>
</tr>
<tr>
<td>Marine fish catch</td>
<td>35</td>
</tr>
<tr>
<td>Cattle population</td>
<td>4</td>
</tr>
<tr>
<td>Pig population</td>
<td>9</td>
</tr>
<tr>
<td>Irrigated area</td>
<td>5</td>
</tr>
<tr>
<td>Cropland</td>
<td>2</td>
</tr>
<tr>
<td>Forest area</td>
<td>20% decrease</td>
</tr>
<tr>
<td>Blue whale population (Southern Ocean)</td>
<td>99.75% decrease</td>
</tr>
<tr>
<td>Fin whale population</td>
<td>97% decrease</td>
</tr>
<tr>
<td>Bird and mammal species</td>
<td>1% decrease</td>
</tr>
</tbody>
</table>

Burning wood products, solid waste and trees, and even the result of certain type of chemical reactions, for example manufacturing cement, contribute to the amount of released CO$_2$. Over the last 200 years: the amount of CO$_2$ in the atmosphere has increased more than 30% and CH$_4$ by even more than 100%. As a result, an increase of 0.5 degrees Celsius in average global temperature in the past century has been observed [2]. Methane is emitted during the production and transport of oil, coal, and natural gas. Another source of the methane emissions is the result of livestock and other agricultural practices. Carbon dioxide accounts for the 82% of all greenhouse gas emissions and methane is the second most prevalent greenhouse emitted by human activities in the U.S.A. [13, 16], despite that methane only accounts for 9% of total greenhouse gas emissions methane is a more potent greenhouse gas than CO$_2$ [16, 17].

To assess a comparison between the impact of different gases, the Global Warming Potential (GWP) is used. The GWP specifically measures how much energy the emission of 1 ton of gas will absorb over a given period of time, using CO$_2$ as comparison over that time period [18]. The larger GWP, the more the earth warms compared to CO$_2$. One hundred years is the usual period of time chosen for this comparison. CO$_2$ has a GWP of 1 by definition while methane has a GWP of 28–36 over 100 years. Methane last about one decade on average but absorbs much more energy than CO$_2$ and similar, to carbon dioxide, methane has natural processes that remove it from the atmosphere. However, these natural processes can not remove these greenhouse gases at the same rate the amount of gases are produced nowadays.

Effects of global warming are vast and can have long-term consequences that could last for millennia.
1.1. CONTEXT

It is argued that near-term climate changes need to be viewed on a long-term perspective to emphasise the severity of this problem, and consciousness should be raised to the fact that climate change will extend longer than the entire history of the human civilisation [19].

The increase in the average temperature due to global warming\(^1\) [9, 10] could lead to a global risk of deadly heat waves [20], with associated mortality that can be attributed to this climate change [21]. The effect of global warming is shown dramatically in a figure with the famous hockey stick (see Figure 1.1), a graph describing the reconstruction of the temperature over the past 1000 to 2000 years using tree-rings, ice cores, coral and other records that act as proxies for temperature. This reconstruction found that global temperature gradually cooled over the past millennia with a sharp upturn in the 20th century. The main result from the paper [22] is that global temperatures over the last few decades are the warmest in the last 100 years. The effects of this warming are not just restricted to the human species but also to plants and non-human animals where local extinction is already happening [23]. It is really important to make people aware of the dangers that we are facing and realise that the human beings can not survive without a biodiverse environment.

![Figure 1.1: Hockey stick graph from [22].](image)

Effects in the ecosystem are visible now, like the calving event of the Larsen-C ice shelf, the largest remaining ice shelf on the Antarctic peninsula [24], or the weakening of the Atlantic Meridional Overturning Circulation, a large-scale circulation pattern in the Atlantic that plays an important role as a heat and freshwater transport, having a directly influence on climate change [25].

Currently, hurricanes grow in severity reaching 300 km/h wind speed, increasing the frequency of occurrence or the size with super hurricanes like Irma shown in the Figure 1.2. Formation of such hurricanes is heavily related to the increased water temperatures in the Atlantic and Pacific Ocean. The frequency of these type of events is also increasing: at the same time Irma hit the south of Florida, another hurricane was hitting the cost of Mexico, and a third one was approaching with probably the same outcome (see Figure 1.3).

---

\(^1\)Climate change, I put global warming as an effect of the climate change.
CHAPTER 1. INTRODUCTION

Figure 1.2: Geocolor Image of Hurricane Irma passing the eastern end of Cuba at about 8:00 a.m. EDT on Sept. 8, 2017, Courtesy of NASA.

The human extraction of natural resources for consumption purposes or for commodities has many key points, one of those is the production of meat at industrial scale farms since this represents the number one-driver of tropical deforestation in south America and worldwide with a nearly 60% of embodied deforestation [26, 27]. The second large driver of deforestation is soy production, accounting for 19% of global deforestation reported within 1990 and 2008 [28]. However, the majority (around 70 to 75% of the world’s soy) goes to feed cows, chickens, pigs, and farmed fish [27, 28], accounting for the meat production. Another example is that half of all accessible fresh water is used by mankind, and all these activities (the increasing of fossil fuel burning, agricultural activities, deforestation, and intensive cattle raising) have increased the number of several greenhouse gases (GHG) in the atmosphere. According to [29] the livestock sector generates more greenhouse gas emissions as measured in $CO_2$ equivalent than all cars, planes, trains, ship and trucks together [30].

This generates positive feedback loops, which can be seen as a vicious circle that accelerates the warming trend (a negative feedback loop will decelerate the warming trend). These positive feedback loops are well know, one example is the melting of ice due the warming temperature. The ice is light-coloured and reflective, acting as a mirror and reflecting a large portion of the sunlight that reaches the earth, thus reducing the amount of warming that this causes. The melting of the ice due the warming reduce white ice surface and increase the “darker-coloured” land and sea water below it. The result is the increase of energy absorbed which lead to more warming that turns out to more ice melted etcetera. These positive feedback loops affect the artic tundra and heating of the arctic coastal waters resulting in additional release of methane, a very potent greenhouse gas that contributes to the problem as well. It is of vital importance to have negative feedback loops related to the human activities.

\footnote{This is truly important since the tropical forest play an important role as natural carbon storage.}
1.1. CONTEXT

Figure 1.3: Three Hurricanes heading east coast of America cuba and Mexico. Courtesy of NASA.

1.1.2 Mitigating the Effects of Human Behaviour

Many efforts have been made to mitigate the impact of human behaviour for example the creation of the Intergovernmental Panel on Climate Change (IPCC) in 1988 as a scientific and intergovernmental body dedicated to providing the world with an objective scientific point of view of climate change and its political and economic impacts [31]. In particular for the fifth assessment report (AR5), the probability of global mean temperatures exceeding the $1.5^\circ$ C and $2^\circ$ C above 1850-1900 levels was estimated, leading to catastrophic effects for the future life on the earth. This report has been used to generate several scenarios to predict future outcomes depending on the measures taken today [32]. Another important effort is the Kyoto Protocol which took place the 11 of December of 1997, where the industrialised countries agreed to apply a set of mandatory targets on GHG emissions. These countries agreed to reduce the mean average of pollutant emissions between 2008 and 2012 by 5.2% taking as a reference the levels of 1990. The protocol entered into force on 16 February 2005 after the ratification by Russia the 18 of November of 2004. However, the USA signed the protocol but did not ratify, even when the USA is the second emitter of CO$_2$ in the world [33].

The most recent example of the efforts is the so called COP21 which was held in Paris in December 2015, the COP21 result in the agreement of 195 countries to keep global temperatures below $2.0^\circ$ C above pre-industrial times and “endeavour to limit” them even more, to $1.5^\circ$ C[34]. Although the agreement was initially seen like a major change towards the improvement of the climate change, in 2017 the USA president decided to withdraw from the agreement, this represent a draw back in the mutual objective of reduce the effects of the climate changes. Thus it is necessary to work

---

$^3$However, the baseline for the ‘pre-industrial’ was not defined, and depending on the baseline selected the scenarios presented by the COP21 can change.
harder and better to overcome all the obstacles and give more precise data and information about the effects of the climate change. Despite the bad news on powerful economies and actors against climate change mitigation there still a wider spectrum of attempts to reduce the emission related to human activities in several possible fields. These fields include; energy, food and water, women and girls, buildings and cities, land use, transport and materials, etc. There are many different ways to oppose the effects of global warming and climate change like the capture and storage of carbon, the use of bioenergy, the combination of these two is known as BECCS (Bioenergy Combined with Carbon Capture and Storage) [35]. Other measurements are the recovery and protection of land by means of protective policies [36], the reduction of flight time per year per person [37], improvement on the methods of construction using alternative cement [36], and the reduction of the use of fossil-fuels as main source of energy.

1.1.3 Wind Energy and Wind Turbines

Among all the solutions to reduce the release of greenhouse gases in the atmosphere, one of the most important aspects is the generation of clean energy, when renewable energy sources like solar, tidal, geothermal and others are implemented. The use of these type of technologies seeks to reduce the use of fossil fuels, even though the use of the fossil-fuels seems to be an addictive habit that is going to be with humans for a long period. The impact of the implementation of new renewable energy technologies will help to reduce pollution and the emission of greenhouse gases.

From all the types of renewable energy and measures to address global warming, wind energy (onshore) is catalogued as the second most influential solution to reduce CO2 emissions on a plausible scenario for 2050 just after “Refrigerant Management” [39]. The impressive number of wind turbines nowadays is astonishing, by the end of 2016, 4% of global electricity demand was covered by 341,320 wind turbines [38].

Figure 1.4: Wind turbines.
The use of the wind as a source of energy has being practised for centuries, for example the vertical axis windmills found at the Persian-Afghan border around 200 BC, the horizontal-axis windmills of the Netherlands and the Mediterranean, and of course all the sailing boats powered by wind energy mentioned since Odysseus, the greeks, vikings and many other cultures who use wind to move across the seas [39].

Windmills got further improvements that were developed in the USA during the 19th century. However, it was not until the notable work in aerodynamics by Betz in Germany and Lancaster in England that allowed subsequent efforts in Denmark, France, Germany, and the UK (during the period between 1935 and 1970), and showed that large-scale wind turbines could work [39, 40].

With the realisation of this idea, wind turbines were installed and its re-emergence as an important source of energy for the world became one of the most significant developments of the late 20th-century [41] in such way that at the end of 2016 the global cumulative installed wind power capacity was 486,790 MW, with China, USA, Germany, India and Spain among the top five countries with 34.7%, 16.9%, 10.3% 5.9% 4.7% of the share respectively [42]. During 2015 a record of 63,467 MW of wind power was installed around the world, and during 2016, the amount of wind power installed was of 54,642 MW around the world [42]. In Figure 1.5 the share of the cumulative capacity of wind energy installed is shown.

For the specific case of the Netherlands, the wind energy installation broke records in 2016, entering the global top 10 in terms of annual market for the first time in decades. Installing 887 MW in 2016, ending with a total of 4.32 GW, accounted for the 8.9% country’s electricity demand [42].

With the proliferation of turbines, the reduction of costs, the high performance of the turbines, the wind industry is growing. With future cost-reductions wind energy will be the least expensive source of electricity within a decade [43, 44]. Wind farms onshore usually have a small footprint, using around 1% of the land they sit on, so other activities can happen simultaneously with power generation, like farming, grazing and recreation. Moreover, the time required to build a wind farm is between 2 months and 1 year (depending on the capacity of the wind farm) [45] giving the advantage of quickly producing energy and return on investment.

Like other renewable energy technologies, energy production by means of wind turbines depends on weather or seasonal conditions and is not completely constant. Thus it is important to invest in 24-7 renewables such as geothermal and in energy storage and transmission infrastructure.

Some scenarios forecast a CO2 reduction of 84.6 gigatons by 2050 just increasing the world electricity use base on wind energy from 4% to 21.6%, with an estimated cost of €1.03 trillion within 30 years of operation. Under conservative estimates this could result in a €6.2 trillion net savings not including new technological developments and cost reductions which eventually will lead to an increase in capacity to generate more electricity at lower cost using wind power [36, 43].

Wind energy is an option to help reduce the effects of climate change, however, this is not the master solution. In fact, there is not a single solution to stop or reduce climate change, at most, the only viable solution is the combination of several solutions to reduce and fight climate change.

\*Although simpler wind devices date back thousands of years ago.*
1.1.4 Aerodynamic Theories on Wind Energy

Developments in many areas of technology were adapted for application to wind turbines. This helped to the re-emergence of this technology to make it more reliable, more cost effective, and quieter. Areas like power electronics, analytical design and analysis methods, computer science, materials science, testing and monitoring, and aerodynamics benefited from these developments. The topic of aerodynamics was originally a development for aerospace industry but has now been adopted by the wind industry. Developments in the aerodynamic theory help to predict the forces acting on the wind turbine and improve the power production of the turbines. However, the blade element momentum (BEM) theory, one of the earliest aerodynamic theories for wind turbines is still in use nowadays. It is a method based on dividing the flow in annular control volumes and then apply a momentum balance and energy conservation to each control volume. The vortex-lattice and vortex-particle methods assume incompressible, inviscid flow and attempt to describe it with either vorticity sheets or vortex particles. The most advanced methods, are based on Reynolds-Averaged Navier-Stokes (RANS) equations and make use of turbulent models or even resort to Large Eddy Simulation (LES) [1].

Some of these methods constitute what is known as Computation Fluid Dynamics (CFD), a field
that has a wide range of methods to solve fluid flow problems. However, CFD methods like RANS or LES solvers, although accurate, tend to be expensive (in time and money). If unsteady considerations are taken into account the computational time could extend to months for a fully described wind turbine solution. On top of that, the trend of the continuous growth of the wind turbines (Figure 1.6) make it even more difficult to perform accurate and fast simulations. With this in mind, is necessary to find some other method that can help to make accurate predictions, in which computation time is reduced, but an adequate level of physics in the description is maintained.

The panel method is an attractive flow analysis technique that, in contrast to other CFD methods, is not expensive and still has fundamental physics involved to work out the solutions. This is a tool of choice for wind turbine and propeller simulations, its major advantage. Its the ability to rapidly simulate complex configuration with large wakes in external flow domains. Another advantage of the panel method is that the discretisation is made in the surface elements only, meaning a much faster and simple meshing.

However, the panel method does not model viscosity, compressibility, and distributed volume vorticity, although some additional models allow a degree of correction in the compressible case. With boundary layer simulation the effect of viscosity can be artificially accounted for [46].

Some of the first uses of the panel method was for the aerospace industry, since the panel method performs better for fully-attached and high-Reynolds-number ($> 10^6$) and subsonic (Mach number $<< 1$) flows [47]. Panel method results are very effective at obtaining lift force and drag force on a
CHAPTER 1. INTRODUCTION

wing (and by extension on a wind turbine blade) as long as the viscous effects are negligible (away from stall conditions). The panel method requires a wake shed from the wing trailing edge in order to predict lift. As soon as the trailing edge is identified the wakes can be generated automatically. If there is no wake defined, the panel method operates in its pure non-lifting inviscid mode.

The panel method is a valuable tool for the analysis of fully attached subsonic flows with high Reynolds number. More about this method and the complete description can be found in Chapter 2.

1.2 Problem Statement

Despite the accuracy of CFD methods, those are usually expensive methods to solve and predict the turbine behaviour in unsteady problems, so it is necessary to come with a fast solution that still involves adequate physics for the solution. In particular for unsteady flow cases, an efficient option is the panel method, which solves the problem to engineering accuracy and is not that computationally expensive. This method has shown a good match with experiment for steady cases [48]. In this thesis we analyse and validate the accuracy of the method described in [48] for unsteady flow cases.

1.3 Objectives

This thesis work will focus on the validation of the multilevel panel method code developed in [48]. Specific points will be discussed in this thesis.

1. Validate the multilevel panel method using the database of results from the New MEXICO wind tunnel experiment.
   - Validation of the accuracy of the method under the unsteady conditions.
   - Discuss the possible flow cases, areas of application and disadvantages of the method for wind turbine aerodynamics simulations.

2. Generate a base work for future projects using the routines and codes for the extraction of the information of the New MEXICO experiment.

1.4 Approach

For the validation of the multilevel panel method for the prediction of the aerodynamics behaviour of wind turbines it was decided to use the New MEXICO database to obtain pressure distributions at 5 sections along the span of the blade. The pressure data can be obtained easily from the
1.5. **THESIS OUTLINE**

numerical simulations. It is then easy to compare the experimentally and numerically obtained values of the pressure coefficients as function of dimensionless chord wise position.

The selected radii for the comparison of pressure distribution followed the sensor locations of the MEXICO and New MEXICO experiments, being at 25%, 35%, 60%, 82% and 92% of the radius. The experimental data consists of several gigabytes of information. From the available pressure data the yaw cases were selected for the validation of the multilevel panel method (among with other cases to verify limitation of such method).

For a fixed set of azimuth angles \([0^\circ, 60^\circ, 120^\circ, 180^\circ, 240^\circ, 300^\circ]\) the pressure data was collected from the database. For each azimuth position an ensemble average was determined for all 5 airfoils sections. The five chord-wise pressure distributions for each specific azimuth angle will be used for the validation of the multilevel panel method as described in Chapter 4. Some time dependent compilations were made for the cases of the pitch ramp and for the step in rotor speed.

Finally the comparison was made among the simulated values with the extracted value of the experiment.

### 1.5 Thesis outline

The thesis is divided in 4 chapters. The second chapter gives an introduction and a briefly description of the multilevel panel method itself and discusses the equations solved during the simulations as well as the boundary conditions selected and the limitations of this panel method are discussed. The third chapter contains the relevant information to understand the New MEXICO wind tunnel experiment, and the specific conditions used in this thesis. The fourth chapter contains the comparison of the numerically simulated and the experimental data along with a discussion of each relevant case. Chapter five includes the summary and conclusions of the works as well as some possible ideas for future works. Finally the appendixes contain the information related to the pre and post processing of the information from the New MEXICO experiment for future works and students, including the codes used for processing the experimental data.
Chapter 2

Multilevel Panel Method

“If the honor and wisdom and joy of such a reading are not to be my own, then let them be for others. Let heaven exist, though my own place be in hell.”

Jorge Luis Borges, The Library of Babel

2.1 Introduction

In 1962 Hess and Smith [49] published a paper called “Calculation of Non-lifting Potential Flow About Arbitrary Three-Dimensional Bodies” which as the title indicates, explores a general method for computing the incompressible potential flow about arbitrary, non-lifting, three-dimensional bodies. This is considered a pioneer work in the 3D panel methods field, which in fact could also be considered as a pioneer work in the entire CFD field too.

From the theoretical methods available the panel method is one of the most versatile methods from the user’s point of view. The method is commonly known as “panel method”, but also sometimes known as “boundary integral” or “surface singularity” method, even sometimes is called “boundary element method”. Nowadays the boundary element method is known in the wind energy field under the classic name of “panel method”, since it could be confused with the blade element momentum method, also related to wind turbine aerodynamics, that shares the same acronym (BEM). Therefore the name “panel method” will be used throughout this thesis in order to avoid confusions.

Since the work of Hess and Smith, the panel method has evolved and there exist many types of panel methods. However, the main principle of the method remains unchanged, that is “covering the surface with singularity elements”, which is the main advantage of the panel method since the problem is reduced to an integral equation on the surface of the geometry only. Therefore, no volume discretisation is required [46]. As a result the method can handle easily numerical
simulations of relatively complicated geometries rather easily.

The shortcomings of the panel method are connected with the assumption of inviscid potential flow. Standard panel methods do not model viscosity, strong non-linear compressibility and volume distributed vorticity. This inability to simulate high-speed transonic flows, allow methods based on the unsteady Navier-Stokes equations\textsuperscript{1} like RANS or LES to be used in the field of CFD. However, the panel method is still being considered an engineering tool of choice for wind turbine and propeller simulations. The major advantage for these flow problems lies in the ability to rapidly simulate configurations with large and complex wakes in external flow domains\textsuperscript{46}.

In this thesis we give a concise description of the panel method developed in\textsuperscript{48}. For a more thorough discussion the reader is referred to the original work.

### 2.2 Governing Equations

For the description of the panel method it is necessary to have certain physical considerations. First, the normal operation of the wind turbine will be in which the local flow velocities occurring at the rotor blades are at most 30\% the speed of the sound\textsuperscript{2}, meanwhile far from the wind turbine the velocities are even slower than this. Hence, the flow can be assumed incompressible. Moreover, the effects related to heat will be not taken in account, and the mass density can be considered constant. Secondly, the effects of viscosity are ignored since the high operational values of the Reynolds numbers limits the regions of vorticity to thin layers near the boundary. These considerations allow us to reduce the set of equations.

For unsteady incompressible flow, the equation for the conservation of mass reduces to

$$\nabla \cdot \vec{u} = 0.$$  

This equation does not have an explicit time derivative term, however, the unsteady boundary conditions will add the time dependence to the solution. The equation for the conservation of momentum for unsteady, incompressible, and inviscid flows is

$$\rho_\infty \frac{\partial \vec{u}}{\partial t} + \rho_\infty (\vec{u} \cdot \nabla)\vec{u} + \nabla p = \vec{0}.$$  

If rotational flow is confined to infinitesimal thin boundary layer and wake regions is assume the complexity can be reduce notably. We assume that the flow is irrotational everywhere else, i.e. $\nabla \times \vec{u} = \vec{0}$. This makes it possible to write the velocity vector field $\vec{u}(\vec{x}, t)$ as the gradient of a scalar velocity potential function $\Phi(\vec{x}, t)$:

\textsuperscript{1}Or Navier-Stokes equations
\textsuperscript{2}That means that the Mach number will be $Ma \leq 0.3$
2.2. GOVERNING EQUATIONS

\[ \vec{u} = \nabla \Phi. \] (2.3)

Substituting equation (2.3) in the continuity equation (2.1) gives the Laplace equation for the velocity potential in volume \( V \):

\[ \nabla \cdot \nabla \Phi = 0. \] (2.4)

When equation (2.3) is substituted in the equation for conservation of momentum (2.2), the Bernoulli equation for unsteady potential flow is obtained,

\[ \frac{\partial \Phi}{\partial t} + \frac{1}{2} \nabla \Phi \cdot \nabla \Phi + \frac{p}{\rho_\infty} = C(t), \] (2.5)

which is valid everywhere in the flow domain.

2.2.1 Boundary Integral Equation

Let us assume that a volume \( V \) can be decomposed into a set of non-overlapping volumes \( V_m \), the boundaries of this set are \( \partial V_m \) like in Figure 2.1. We will use the internal Dirichlet formulation popularised by Maskew [50]. This formulation assumes that we are only interested in the flow field on one side of the surface. The internal flow field on the other side of the surface will be set explicitly in advance.

Figure 2.1: Flow domain \( V \in \mathbb{R}^3 \) is the union of non-overlapping volumes \( V \) and inner boundaries \( S_{m,k} \) that separate volume \( V_m \) from volume \( V_k \). Unit normal vector \( \vec{n}_m \) is defined to point into volume \( V_m \).
For Laplace equation (2.4), it is possible to express the potential for point $\vec{x}$ in volume $V$ in terms of velocity potential contributions, that is “background” potential ($\Phi_{\infty}$), plus contribution from dipole and source distributions.

$$\Phi = \Phi_{\infty} + \varphi_\mu + \varphi_\sigma,$$

(2.6)

It is assumed that $\Phi_{\infty}$ and its gradient $\vec{u}_{\infty}$ are chosen beforehand. For the dipole distribution $\mu(\vec{y}, t)$ and the source distribution $\sigma(\vec{y}, t)$ with $\vec{y} \in S$ it is possible to define the perturbation velocity potentials $\varphi_\mu$ and $\varphi_\sigma$ as

$$\varphi_\mu(\vec{x}, t) = \frac{-1}{4\pi} \int_S \mu \frac{\vec{n}_m \cdot \vec{r}}{r^3} dS,$$

(2.7)

$$\varphi_\sigma(\vec{x}, t) = \frac{-1}{4\pi} \int_S \sigma \frac{1}{r} dS,$$

(2.8)

where $\vec{n}_m(\vec{y}, t)$ is the unit normal vector pointing into volume $V_m$ of interest.

Using the velocity potential functions $\Phi_m(\vec{y}, t)$ and $\Phi_k(\vec{y}, t)$ on both sides of the surface, it is possible to express the dipole strength and the source strength, $\mu(\vec{y}, t)$ and $\sigma(\vec{y}, t)$ as

$$\mu(\vec{y}, t) = -(\Phi_m - \Phi_k),$$

(2.9)

$$\sigma(\vec{y}, t) = \nabla(\Phi_m - \Phi_k) \cdot \vec{n}_m$$

(2.10)

We can set the potential $\Phi_k$ inside the body to a convenient value for us, while $\Phi_m$ it is the solution we are interested to find in the physical flow domain. The fictitious flow field is set to be equal to the onset flow field, that is to say, $\Phi_k = \Phi_{\infty}$ and $\vec{u}_k = \vec{u}_{\infty}$.

### 2.2.2 Boundary Conditions

To solve the Laplace equation (2.4) for the potential field $\Phi(\vec{x}, t)$ it is necessary to have the appropriate boundary conditions, in reference to the field $\Phi_{\infty}(\vec{x}, t)$ plus source and dipole perturbations potential fields $\varphi_\sigma(\vec{x}, t)$ and $\varphi_\mu(\vec{x}, t)$.

#### 2.2.2.1 Body Surface

The boundary condition for the physical solution specifies that flow velocity at the surface with a normal direction has to be equal to the normal component of surface velocity $\vec{v}_s$, plus a specified outflow velocity relative to the moving boundary, this last one also in normal direction $v_n(\vec{x}, t)$:
2.2. GOVERNING EQUATIONS

\[ \nabla \Phi_m \cdot \vec{n}_m = \vec{u}_s \cdot \vec{n}_m + v_n \]  

(2.11)

In our case the velocity potential in the fictitious flow domains \( V_k \) is known in advance, and set to

\[ \Phi_k(\vec{x}, t) = \Phi_\infty(\vec{x}, t), \quad \vec{x} \in V_k, \]  

(2.12)

and

\[ \vec{u}_k(\vec{x}, t) = \vec{u}_\infty \quad \vec{x} \in V_k \]  

(2.13)

The boundary integral equation (2.6) at point \( \vec{x} \in V_k \) gives

\[ \varphi_\mu(\vec{x}, t) + \varphi_\sigma(\vec{x}, t) = 0 \quad \vec{x} \in V_k. \]  

(2.14)

The equation (2.6) can be expressed in terms of Principal Value and Finite Part taking the limit of point \( \vec{x} \) approaching the surface \( S_k \) at point \( \vec{y} \), giving:

\[ \frac{1}{2} \mu + \varphi_\mu(\vec{x}, t) + \varphi_\sigma(\vec{x}, t) = 0 \quad \vec{x} \to \vec{y} \in S_k. \]  

(2.15)

Boundary condition (2.11) and the specified velocity potential \( \Phi_k \) (2.12) can be substituted in the definition of the source strength (2.10). This gives an expression for the source strength in terms of known quantities:

\[ \sigma(\vec{y}, t) = (\vec{u}_s - \vec{u}_\infty) \cdot \vec{n}_m + v_n, \quad \vec{y} \in S_{m,k}. \]  

(2.16)

Equation (2.15) evaluated at \( \vec{x} \to S_k \) gives thus an expression that contains the dipole strength \( \mu(\vec{y}, t) \) as a function of known quantities. The surface gradient of the dipole strength (2.9) at the surface side of interest \( (S_{m,}) \) gives the tangential component of the velocity

\[ \nabla_s \Phi_m(\vec{x}, t) = \nabla_s \Phi_\infty - \nabla_s \mu, \quad \vec{x} \to S_m. \]  

(2.17)

Combining the expression of the normal velocity from boundary condition (2.11) and the tangential velocity (2.17), an expression for the velocity at the surface in the inertial reference system can be found:

\[ \nabla \Phi_m(\vec{x}, t) = (\vec{u}_s \cdot \vec{n}_m + v_n)\vec{n}_m + \nabla_s \Phi_\infty - \nabla_s \mu, \quad \vec{x} \to S_m. \]  

(2.18)
Chapter 2. Multilevel Panel Method

which can also be written as

\[
\vec{u}_m(\vec{x}, t) = \vec{u}_\infty + \sigma \vec{n}_m - \nabla_s \mu \quad \vec{x} \to S_m. \tag{2.19}
\]

Equation 2.19 shows the components of the velocity at the surface side of interest. These components are a known base flow field \( \vec{u}_\infty \), a perturbation flow field related to the source \( \sigma \vec{n}_m \), and a perturbation due to the flow field dipole singularity distribution \( \nabla_s \mu \) tangential to the surface.

For an observer at point \( \vec{x} \in S_m \), moving with the local surface velocity \( \vec{u}_s \) the relative velocity that is experienced will be

\[
\vec{u}_{rel}(\vec{x}, t) = \vec{u}_m - \vec{u}_s. \tag{2.20}
\]

Combined with (2.19) this expression becomes

\[
\vec{u}_{rel} = \vec{u}_\infty - \vec{u}_s + \sigma \vec{n}_m - \nabla_s \mu. \tag{2.21}
\]

In conclusion, the resulting set of equations is

\[
\begin{align*}
\sigma &= (\vec{u}_s - \vec{u}_\infty) \cdot \vec{n} + v_n, \\
\frac{1}{2} \mu + \varphi^p &= -\varphi^p, \quad \vec{x} \to S \\
\vec{u}_m &= \vec{u}_\infty + \sigma \vec{n} - \nabla_s \mu. \quad \vec{x} \to S
\end{align*} \tag{2.22}
\]

2.2.3 Wake Surface

The wake is explicitly added in the potential flow model. This is done at the point where the flow leaves the surface, the trailing edge of the lifting body. At this point a Kutta condition must be imposed in order to obtain a smooth flow with finite velocity. The Kutta conditions equate the dipole strength at the first point of the wake to the jump in dipole strengths across the trailing edge.

\[
\mu_{we} = [[\varphi]]_{te} = [[\mu]]_{te}. \tag{2.23}
\]

For the evolution of the wake the theorems of Helmholtz and Kelvin for vorticity dynamics will be used.

In [48] it is concluded that in incompressible inviscid flows a tube of vorticity preserves its identity (in general deforming and stretching) when moving with the velocity field. The equations
corresponding in terms of the evolution of the wake element position \( \vec{x}_w \) and wake element dipole strength \( \mu_w \) are

\[
\frac{d\vec{x}_w}{dt} = \vec{u}, \quad \vec{x}_w(t_0) = \vec{x}_{te}(t_0),
\]

(2.24)

and the material derivative of the wake dipole strength

\[
\frac{D\mu_w}{Dt} = 0, \quad \mu_w(t_0) = \mu_{w_{te}}(t_0),
\]

(2.25)

where \( t_0 \) is the time of wake element creation.

After differentiating the potential field contributions (2.6) with respect to \( \vec{x} \) the velocity field gives:

\[
\vec{u}(\vec{x},t) = \vec{u}_\infty + \vec{u}_\mu + \vec{u}_\sigma,
\]

(2.26)

where the perturbation velocities induced at point \( \vec{x} \) by the dipole and source distributions on surface \( S_{m,k} \) can be shown to be

\[
\vec{u}_\mu(\vec{x},t) = -\frac{1}{4\pi} \int_S (\vec{n}_m \times \nabla \mu) \times \frac{\vec{r}}{r^3} dS + \frac{1}{4\pi} \int_{\delta S} \mu \frac{\vec{r}}{r^3} \times d\vec{l},
\]

(2.27)

\[
\vec{u}_\sigma(\vec{x},t) = \frac{1}{4\pi} \int_S \vec{\sigma} \frac{\vec{r}}{r^3} dS.
\]

(2.28)

The velocity field associated with a dipole distribution is equivalent to the induced velocity by a surface vorticity distribution \( \vec{\gamma} \) of strength \( \vec{\gamma} = -\vec{n} \times \nabla \mu \) plus the velocity induced by a discrete vortex filament \( \Gamma \) of strength \( \Gamma = \mu \) along the edge of \( S \). Though the line integral in (2.27) is along the contour of the surface, in general such a contribution appears whenever there is a jump in the dipole distribution. The advection of the wake is performed by integrating (2.24) over a time interval. In this integration the local velocity \( \vec{u} \) is required at wake element positions \( \vec{x}_w \). For each point an evaluation of the integrals in Equations (2.27) and (2.28) over the surface and along its edge is needed.

### 2.2.4 Pressure Coefficient Definition

Now we will use the unsteady Bernoulli equation (2.5) to get an expression for the pressure \( p \) by relating upstream flow quantities with perturbed local quantities. Let \( \Phi_\infty(\vec{x},t) \) be the upstream total velocity potential and \( p_\infty \) the undisturbed upstream pressure, in this way the unperturbed
onset velocity is \( \vec{u}_{\infty} = \nabla \Phi_{\infty} \). In the same way, the perturbed local quantities are the local flow velocity \( \vec{u}_{m} = \nabla \Phi_{\infty} + \nabla \varphi_{m} \), total velocity potential \( \Phi = \Phi_{\infty} + \varphi_{m} \) and pressure \( p \). Substituted in the Bernoulli equation (2.5) we obtain

\[
\frac{p - p_{\infty}}{\frac{1}{2} \rho_{\infty}} = \vec{u}_{\infty} \cdot \vec{u}_{\infty} - \vec{u}_{m} \cdot \vec{u}_{m} - 2 \frac{\partial \varphi_{m}}{\partial t}, \tag{2.29}
\]

Getting a definition of the pressure distribution requires the material derivative of the perturbation potential \( \varphi_{m} \) expressed at a point on the surface moving with velocity \( \vec{u}_{s} \):

\[
\frac{D_{s} \varphi_{m}}{Dt} = \frac{\partial \varphi_{m}}{\partial t} + \vec{u}_{s} \cdot \nabla \varphi_{m} = \frac{\partial \varphi_{m}}{\partial t} + \vec{u}_{s} \cdot (\vec{u}_{m} - \vec{u}_{\infty}) \tag{2.30}
\]

We can define the reference velocity \( \vec{v}_{ref} \) as

\[
\vec{v}_{ref} \overset{\text{def}}{=} \vec{u}_{\infty} - \vec{u}_{s} \tag{2.31}
\]

We substitute equations (2.20), (2.30), and (2.31) in equation (2.29) and obtain a definition for the pressure coefficient:

\[
C_{p} \overset{\text{def}}{=} \frac{\frac{1}{2} \rho_{\infty} v_{ref}^{2}}{p - p_{\infty}} = 1 - \frac{u_{rel}^{2}}{v_{ref}^{2}} - 2 \frac{D_{s} \varphi_{m}}{Dt}, \tag{2.32}
\]

where \( u_{rel}^{2} = \vec{u}_{rel} \cdot \vec{u}_{rel} \) and \( v_{ref}^{2} = \vec{v}_{ref} \cdot \vec{v}_{ref} \).

### 2.2.5 Panel Method

Figure 2.2 shows the ordering of the nodes and panels used in the current implementation, as well as the definition of the normal unit vector, (which is used as reference). In Figure 2.3 the panels over the surface of one blade and how these can be grouped into surface patches is shown.
2.2. GOVERNING EQUATIONS

(a) Illustration of the ordering of panels and nodes in the grid with the use of indices \(i\) and \(j\) with corresponding directions \(\bar{i}\) and \(\bar{j}\).

(b) The unit normal vector \(\bar{n}\) is assumed to point into the flow domain and is deduced from the right-hand rule vector cross product of the \(i\) and \(j\) index related directions: \(\bar{n} \equiv \bar{i} \times \bar{j}\).

Figure 2.2: Panel and node Ordering

Figure 2.3: Panels over blade.
2.3 Multi Level Multi Integration Cluster Scheme

The panel method is the main tool to solve and predict numerically the behaviour of the wind turbine. However, this method is improved with the help of a multilevel scheme (therefore the name “multilevel panel method”). We will describe the basic components and ideas of the multilevel scheme used in this panel method, for more details the reader is referred to [48].

The motivation behind the use of the multilevel scheme is the reduction in the number of operations performed during the evaluation of the integrals over the entire surface of the configuration. This scheme enable to reduce the work done from $O(N^2)$ operations to a problem of $O(N)$ computations.

Before the general idea of the multilevel scheme is described, it is necessary to explain some basic concepts. The term source refers to any kernel function, which for the panel methods is usually reserved for the velocity potential integral involving kernel and its gradient (2.8) and the term dipole is linked to the integral of the dipole distribution involving kernel and its gradient (2.7).

The scheme if going to carry information along different grids, these grids will be known as source and receiver grids. Receivers denote the locations where the evaluation of an integral transform is needed. The term source will reflect the locations that will provide the information to the receivers. Transformations from sources to receivers (or vice versa) will involve interpolation or anterpolation. Lowercase and upper case letters are used for fine and coarse grid variables respectively.

The key element in this scheme is the approximate representation of the kernel function $K(x, y)$. In the multilevel panel method the kernel is approximated by a Lagrange interpolation. This briefly explanation is made using as an example smooth non singular kernel function $K(x, y)$ on a simple one-dimension. Let’s start with the discretised integral transform (using the panel method) and evaluated in receiver points $x_i^h$.

Figure 2.4: Concept of the multilevel scheme for a smooth kernel integral in !D.
\[ \phi^h(x_i^h) = \phi_i^h = \int K(x_i^h, y) \bar{\sigma}^h(y) dy = \sum_j K_{i,j}^{hh} \sigma_j^h \] (2.33)

where the approximation of the kernel can be written as

\[ \bar{K}_{I,J}^{hh} = \sum_l \sum_j K_{I,j}^{HH} L^I(x_i^h) L^J(y_j^h) \] (2.34)

equation (2.34) is substituted in (2.33) which gives

\[ \phi_i^h \approx \sum_j \sum_l \sum_j K_{I,j}^{HH} L^I(x_i^h) L^J(y_j^h) \sigma_j^h \] (2.35)

equation (2.36) can be reordered without any assumptions to obtain

\[ \phi_i^h \approx \sum_l L^I(x_i^h) \left( \sum_j K_{I,j}^{HH} \left( \sum_j L^J(y_j^h) \sigma_j^h \right) \right) \] (2.36)

Equation (2.36) summarises the general idea of the multilevel (multi integration) scheme for a smooth kernel with three elements. Using Figure 2.4 these steps are described going inward-out of the parentheses.

1. **Anterpolation**
   Refers to the effective transfer of the source strenghts \( \sigma_j^h \) in points \( y_j^h \) to the “pseudo-sources” \( \sigma_{I,j}^H \) located in the encompassing box grid nodes \( Y_{I,j}^H \) by
   \[ \sigma_{I,j}^H = \sum_j L^J(y_j^h) \sigma_j^h \] (2.37)

2. **Coarse grid summation**
   A matrix-vector multiplication is performed using the kernel values \( K_{I,j}^{HH} \) representing the influence of source box nodes on receiver box grid nodes and the vector of “pseudo-sources” \( \sigma_{I,j}^H \) in source nodes \( Y_{I,j}^H \) to give the receiver values \( \phi_{I,j}^H \) in the receiver box grid nodes \( X_I^H \) by means of
   \[ \phi_{I,j}^H = \sum_j K_{I,j}^{HH} \sigma_{I,j}^H \] (2.38)

3. **Interpolation**
The receiver values $\phi_i^h$ in points $x_i^h$ are obtained by interpolation of the receiver values $\phi_j^h$ in the surrounding nodes $X_j^H$ of the receiver box grid through

$$
\phi_i^h = \sum_j L^I(x_i^h) \phi_j^H
$$

(2.39)

Since the course grid summation in equation (2.38) has the form of the original discrete problem (2.33), the procedure can be used recursively introducing a hierarchy of unceasingly larger (coarser) grid (boxes). The interpolation is always done within the coarse receiver box analysed. The interpolation assigns information only to the nodes of the encompassing “parent” source box. The multilevel scheme reduces significantly the number of sourcing points, allowing a fast computation without losing information.

The simplification made in this thesis for the multilevel scheme serves merely to give the general idea of this scheme. The reader is strongly encouraged to seek a more detailed explanation of the multilevel multi integration scheme cluster used in this panel method developed by Van Garrel [48] where singular kernels and higher dimensions are analysed. For a general concept of the multilevel method the reader is referred to the work by Venner and Lubrecht [51].
Chapter 3

New MEXICO Wind Tunnel Experiment

“He who is to perform a horrendous act should imagine to himself that it is already done, should impose upon himself a future as irrevocable as the past.”

Jorge Luis Borges, The Garden of Forking Paths

3.1 Introduction

Experiments on wind turbines have been performed over the last 30 years, tests in wind tunnels and test in field. Wind turbine experiments are essential not only for the understanding of the aerodynamic mechanism but also for code validation.

Field experiments have been performed extensively and with detailed reporting, for example, the IEA Wind Annex XIV [52] and the Annex XVIII [53] and more recently the Annex XX [54]. These type of test can provide information about wind turbines working in natural conditions. However, in these type of experiments the conditions are not fully controlled and known. Frequently they have long measurement campaigns to get stochastically significant data. Moreover, they require an extensive reduction of the data that could turn into a really complicated task. In wind tunnel tests the operation conditions are know and can be controlled. The results from this wind tunnel test are commonly used nowadays to compare CFD simulations with wind tunnel data [55, 56].

Experiments have been done in wind tunnels, like the NREL Unsteady Aerodynamics Experiment (UAE) Phase VI accomplished in 2000 with a test model of a two-bladed stall-regulated wind turbine in the 24m x 36m NASA-Ames wind tunnel. The (UAE) Phase VI program has been running for nearly 10 years and a large amount of useful information is available for scientific research [57]. Another experiment was carried out at the Norwegian University of Science and
CHAPTER 3. NEW MEXICO WIND TUNNEL EXPERIMENT

Technology (NTNU) on a small scale turbine placed in the subsonic NTNU wind tunnel [58], these experiments constitute a data base for the validation of simulations.

The follow-up of IEA Wind Task 20, was IEA Task 29, a systematic wind turbine test where wind tunnel measurements from the EU project Model Experiments in Controlled Conditions (MEXICO) were used. The first campaign was held December 2006, which allowed to obtain numerous and detailed information about the aerodynamic and loads on a the wind turbine [59, 60].

The New MEXICO experiment was then carried out in June/July 2014\(^1\) with several improvements with respect to the previous one, correcting errors and solving the problems from the original experiment [61, 62]. Both experiments have several cases with different conditions, and in order to preserve the validity of the first MEXICO database and validate the settings of the new measurements, some of the previous cases were reproduced. The second experiment is chosen as the source of test data for the validation of the multilevel panel method.

The (New) MEXICO project is a sophisticated aerodynamic experiment carried on the Large Scale Low-Speed Facility (LLF) of the German-Dutch Wind Tunnel Organisation (DNW) with the main objective of “create a data base of detailed aerodynamic and load measurements on a wind turbine model in a large and high-quality wind tunnel to be used for model validation and improvement” [60].

The experiment successfully generated a 100 GB database that is available for all the consortium members of nine research partners in five countries, including Denmark, Sweden, Greece and Israel. ECN (Energy Research Centre of the Netherlands) is the project coordinator [59].

3.2 Definition and Conventions

For the New MEXICO experiment, a three blade rotor model was tested. A discussion of the definitions used in of the experiment is necessary to facilitate the interpretation of the data and detail the experimental instrumentation. The following definitions are illustrated in Figure 3.1.

- **Blade Numbering:** The order in which the blades pass the tower is 1,2,3.

- **Rotor Azimuth Angle:** The zero rotor azimuth angle is defined to be the 12 o’clock position for blade 1 and coincides with the 360\(^\circ\) azimuth angle. Hence, the blade azimuth angles for the three blades will be 0\(^\circ\), 240\(^\circ\) and 120\(^\circ\) for blade 1, 2 and 3 respectively.

- **Yaw Angle and Coordinate System:** The coordinate system is defined with the origin in the rotor centre and fixed to the model. This coordinate system is rotating with the model in case of yawed flow, but does not rotate with the azimuth variations. The other coordinate system is the fixed tunnel system, which originates in the centre of the rotor as well but does not change with the yaw angle. The yaw angle is defined as the angle between the flow velocity \(V_\infty\) and the \(X_{\text{model}}\) axis.

\(^1\)Almost 8 years after the first tunnel slot!
3.2. DEFINITION AND CONVENTIONS

• **Pitch angle:** The pitch angle is measured using the tip section of the blade as a reference. Figure 3.2a and 3.2b show two different pitch positions compared with the $0^\circ$ pitch at the tip of the blade.

Figure 3.1: Azimuth angle, yaw, and coordinate system definitions [63].

![Diagram of blade and coordinate system](image)

(a) **Front view**

(b) **Top view**

Figure 3.2: Pitch angle definition. In (a) is shown the blade profile at the tip of the blade with $0^\circ$ pitch angle (dash point) and the position of the blade with a pitch of $-2.3^\circ$ (green line). In (b) the blade profile $0^\circ$ pitch angle (dash point) and the position of the blade with a pitch of $5^\circ$ (blue line).
### 3.3 Wind Turbine

The wind turbine model subject to the experiment includes a three blades rotor of 4.5 m diameter, with a speed controller and a pitch actuator included. This model was designed specifically for the MEXICO experiment. In the Table 3.1 a summary of the general information of the wind turbine is given.

Table 3.1: General turbine information, modified from [63].

<table>
<thead>
<tr>
<th>Rotor</th>
<th>Number of blades</th>
<th>Clockwise (facing the upwind part of the rotor)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power regulation</td>
<td>-</td>
<td>Not present, speed control by motor/generator</td>
</tr>
<tr>
<td>Rotor speed $[RPM]$</td>
<td>$[0, 324.5, 424.5]$</td>
<td>For some measures a rotor speed ramp was used.</td>
</tr>
<tr>
<td>Yaw angle $[^\circ]$</td>
<td>$[-30, 0, 15, 30, 45]$</td>
<td></td>
</tr>
<tr>
<td>Swept area $[m^2]$</td>
<td>15.9</td>
<td></td>
</tr>
<tr>
<td>Rotor diameter $[m]$</td>
<td>4.5</td>
<td></td>
</tr>
<tr>
<td>Hub height $[m]$</td>
<td>5.49</td>
<td></td>
</tr>
<tr>
<td>Tilt angle $[^\circ]$</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Blades</th>
<th>Blade length $[m]$</th>
<th>2.04</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cone angle $[^\circ]$</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Prebend</td>
<td>-</td>
<td>No prebend</td>
</tr>
<tr>
<td>Roughness</td>
<td>-</td>
<td>Zig-zag tape at 5% chord (0.25mm thick, pressure and suction side)</td>
</tr>
<tr>
<td>Material</td>
<td>-</td>
<td>Aluminum</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tower</th>
<th>Type</th>
<th>Tubular</th>
</tr>
</thead>
<tbody>
<tr>
<td>Height including base $[m]$</td>
<td>5.120</td>
<td></td>
</tr>
<tr>
<td>Diameter $[m]$</td>
<td>0.508</td>
<td></td>
</tr>
<tr>
<td>Wall thickness $[m]$</td>
<td>0.011</td>
<td></td>
</tr>
<tr>
<td>Roughness</td>
<td>-</td>
<td>Spiral flange to provoke transition</td>
</tr>
<tr>
<td>Material</td>
<td>-</td>
<td>Steel</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pitch System</th>
<th>Type range</th>
<th>Linear actuator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range $[^\circ]$</td>
<td>$[-5.3, 90]$</td>
<td>For some measures a pitch step was applied.</td>
</tr>
</tbody>
</table>
3.3. WIND TURBINE

3.3.1 Blades

The blades used in the New MEXICO rotor is constituted by three different airfoils: DU91-W2-250 closer to the root of the blade from 20% to 45.56% radius, at midspan the RISØA2-21\(^2\) covering a region from 54.44% to 65.56% and in the outer rotor section the blade with the NACA64-418 profile from the 74.44% to the 100%. Each part of their blade were constituted by a constant airfoil cross-section shown in the Figure 3.3. In between each of this airfoil profiles a transfer zone is lofted with tangency (as illustrated on the Figure 3.4). The tip of the blade is remodelled to a shifted ellipse.

![Figure 3.3: Airfoil sections used in the New MEXICO rotor.](image)

![Figure 3.4: Blade composition shape from [63].](image)

\(^2\)A persistent error in all the documentation is the use of the incorrect airfoil name of RISOA1-21.
3.4 Wind Tunnel

The LLF wind tunnel used is a closed circuit, atmospheric, continuous low-speed wind tunnel. Figure 3.5 sketches the layout of the wind tunnel configuration. The wind tunnel is operated in open jet configuration throughout the MEXICO experiment. The configuration features an open test section of $9.5 \times 9.5 \, m^2$. In this section the wind flows from a nozzle towards the collector (closed loop between nozzle and collector). The range of the upstream velocity of this tunnel is between 5.5 m/s and 30 m/s.

The blockage of the rotor model counts for 18% of the total area of the nozzle which should result in a negligible solid blockage. An open jet is used instead of a solid wall wind tunnel.

In addition to the tunnel balance data sensors were installed to measure: velocity, temperature, barometric pressure (to obtain density). In the collector entrance 8 pressures sensors were installed additionally.

![Side view](image1)

(a) Side view

![Top view](image2)

(b) Top view

Figure 3.5: Schematic view of LLF tunnel, Nozzle and Collector geometry courtesy from DNW.
3.5 Measurements and Instrumentation

Several types of measurements were obtained during the experiment, like pressure, acceleration, temperature, moments in three directions at the root of the tower (with a 6-component balance), edgewise and flatwise bending moments at the root of the blades, strain gauges, upstream velocity, yaw angle, rotor speed, generator torque, pitch angle, and an optical 1P sensor that generated the trigger signal for the recording of the information.

The model rotor was instrumented with 148 fast Kulite® XCQ-95-062-5A pressure sensors to measure the pressure distribution over 5 sections of the blades. More specific information about the pressure sensors can be found on the Table C1 in the Appendix C. For the two directions of the bending moments, strain gauge bridges were applied at the root of the three blades. Lastly, a large number of Particle Image Velocimetry (PIV) studies were programmed to indicated tip vortices and determine near field inflow and wake velocities as well as the flow field around the rotor. From all these measurements the pressure distribution is of special interest for this work.

The pressure distributions will be used for validation of the multilevel panel method. We focus on the time dependent pressure distributions from the New MEXICO experiment to validate the multilevel panel method from [48] as these are the primary aerodynamic data. Validation of this panel method with derived quantities like lift, bending moment or torque would obfuscate the analysis as a result of the possibility of cancelling errors.

Table 3.2: Position of pressure sensors at blade and radial location

<table>
<thead>
<tr>
<th>Radial location [%]</th>
<th>Blade #</th>
<th>Number of sensors [-]</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>1</td>
<td>28</td>
<td>Measurement</td>
</tr>
<tr>
<td>35</td>
<td>1</td>
<td>27</td>
<td>Measurement</td>
</tr>
<tr>
<td>60</td>
<td>2</td>
<td>25</td>
<td>Measurement</td>
</tr>
<tr>
<td>60</td>
<td>1</td>
<td>8</td>
<td>Reproducibility</td>
</tr>
<tr>
<td>60</td>
<td>3</td>
<td>5</td>
<td>Reproducibility</td>
</tr>
<tr>
<td>82</td>
<td>3</td>
<td>25</td>
<td>Measurement</td>
</tr>
<tr>
<td>82</td>
<td>2</td>
<td>5</td>
<td>Reproducibility</td>
</tr>
<tr>
<td>92</td>
<td>3</td>
<td>25</td>
<td>Measurement</td>
</tr>
</tbody>
</table>

3.5.1 Data Acquisition

The pressure sensors were mounted at five span wise sections, 25%, 35%, 60%, 82% and 92% radial position respectively. The number of pressure transducers per station varied between 25 and 28. Through PBCs (Printed Board Circuits) mounted in the blade the data were led (digitally) over a slip-ring to the data acquisition units mounted on the non-rotation part of the model. The pressure sensors were distributed over the three blades since it was not possible to mount all of them in
one blade. Along these measurement sensors, a limited number of extra sensors were mounted in similar locations at the other blades to check the reproducibility of the pressure measurements on different blades. The distribution of the pressure sensors according to its function is shown in the Table 3.2

### 3.5.2 Uncertainty of Pressure Sensors

Careful attention must be given to the pressure sensors measurements since the sensors registered absolute pressure. The pressure range of this sensors is 35 kPa. It is mentioned in [64] that the sensors a have an uncertainty of 1% of the Full Scale Output (FSO), resulting in an uncertainty of 350 Pa. The dynamic pressure is defined equal to

\[
q = \frac{1}{2} \rho \left[ U_\infty^2 + (\omega r)^2 \right]
\]

This allow us to see that the difference in the pressure coefficient \( \Delta C_p \) by cause of the uncertainty in the pressure sensors is

\[
\Delta C_p = \frac{350}{q} = \frac{350}{\frac{1}{2} \rho \left[ U_\infty^2 + (\omega r)^2 \right]}
\]

The crucial importance of the uncertainty for the measurements and for the future comparison of the pressure distribution lies on the influence of the radial location. Is possible to see in the Table 3.3 the estimated difference in pressure coefficient related to the increase in the radial location. The influence of the uncertainty in the pressure sensors will be lower in the outer sections of the rotor, this is because the relative velocity and the dynamic pressure increase in this direction.

Table 3.3: Estimated pressure coefficient due to uncertainty in the pressure sensors at radial locations for \( \rho = 1.20421 \text{ kg/m}^3 \), \( U_\infty = 14.7 \text{ m/s} \) and 425.1 rpm.

<table>
<thead>
<tr>
<th>Radial location [%]</th>
<th>q [Pa]</th>
<th>( \Delta C_p ) [-]</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>508</td>
<td>0.69</td>
</tr>
<tr>
<td>35</td>
<td>870</td>
<td>0.40</td>
</tr>
<tr>
<td>60</td>
<td>2305</td>
<td>0.15</td>
</tr>
<tr>
<td>82</td>
<td>4192</td>
<td>0.08</td>
</tr>
<tr>
<td>92</td>
<td>5243</td>
<td>0.07</td>
</tr>
</tbody>
</table>
3.5.3 Measurement Data Points

During the experiment every combination of model and tunnel configuration measured corresponds to a unique data point. Nearly 1400 points of such data were recorded. The data points are organised in runs (between tunnel on and tunnel off) and polar(s). This gives a unique name combination for each datapoint, the usual notation was R for run, P for polar and D for data point. Each measurement data point contains information recorded under the specific conditions of that data point. Every data point represents a recording of approximately 5 seconds\(^3\) and it contains the information collected by the sensors, like pressure, RPM, pitch, wind velocity, and other parameters. It is mentioned that for the first MEXICO campaign one of the most important features of the measurements is the extensive flow field mapping by stereo PIV measurements [60]. However, this comment is not made for the New MEXICO experiment, which repeats many of the first cases for the validations of those.

A measurement data point corresponds to a recording during which the conditions remained constant in terms of tunnel conditions and model configuration. The exception is on the dynamic inflow measurements (for example for data points with the pitch step or rotor speed step). It is mentioned before that every data point represents a recording of approximately 5 seconds. The sampling frequency is 5524 Hz, therefore each measurement data point file is going to contain around 27500 samplings. For the dynamic cases the time interval raised to 15 s per file giving nearly 82700 samplings. For most of the cases the rotor speed was around 7.06 Hz (\(\approx 424\) rpm). Hence, each measurement data point contains information of around 35 revolutions\(^4\). Finally, at 7 rev/s and with the sampling frequency is possible to see that the samplings were taken approximately every 0.5° angular interval (see 4.1).

3.6 Data Selection for Validation

It is noted by now that the measurements covered a large combination of parameters resulting in a large number of repeated cases [60, 63, 65]. A complete table of cases and aerodynamic conditions can be found in the Table B2 in Appendix B, and for more details the reader is referred to [66]. The list of the cases used for the current work is summarised as follows:

1. **Yawed rotor:** The measurements have been done at rotor yaw angles \(-30°, 15°, 30°\) and \(45°\). The rotor speed is kept constant at its design value of \(424.1\) rpm and tunnel wind speed is kept at \(15\) m/s. The pitch angle is kept constant at \(-2.3°\).

2. **Dynamic inflow step in blade pitch:** A blade pitch step varied from \(-2.3°\) to \(5°\) (up and down), while maintaining design conditions; rotor speed is kept constant at \(424.1\) rpm and tunnel speed of \(15\) m/s and axi-symmetrically.

\(^3\)In dynamic cases the files contain information of up to 15 seconds of recording

\(^4\)7\(\frac{\text{rev}}{\text{s}}\) \(\times 5\frac{\text{s}}{\text{rev}} = 35\frac{\text{rev}}{\text{s}}\)
3. Dynamic inflow step in rotor speed: A rotor speed step from 424.5 rpm to 324.5 rpm (down and up), while again maintaining design conditions for the other parameters for this case, a blade pitch constant of $-2.3^\circ$ and a tunnel speed of 15 m/s. Flow conditions are axi-symmetric for the other parameters.

This selection of these cases reduced the amount of information from the data base that could be used for the validation, however, the number of datapoint still was a vast amount, therefore it was necessary to narrow the number of datapoint. Then the design conditions came as handy filter for the reduction of datapoint to use. The design conditions are: wind speed flow of 15 m/s, blade pitch at $-2.3^\circ$ and rotor speed of 425.1 rpm, yaw angle $0^\circ$. Table 3.4 shows reduced number of the cases and the corresponding conditions and the selected data points.

Table 3.4: Correlation of data for cases used for the validation

<table>
<thead>
<tr>
<th>Type of measurement</th>
<th>Operational Conditions</th>
<th>RPD range†</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Rotor Speed [RPM]</td>
<td>Tunnel Speed [m/s]</td>
</tr>
<tr>
<td>Yawed flow</td>
<td>425.1</td>
<td>15</td>
</tr>
<tr>
<td>Dynamic inflow: blade pitch step</td>
<td>425.1</td>
<td>15.07</td>
</tr>
<tr>
<td>Dynamic inflow: rotor speed step</td>
<td>Step 424 to 324 and back</td>
<td>15.1</td>
</tr>
</tbody>
</table>

† Those specific datapoint are chosen as the datapoint with less malfunctioning sensors.

As mentioned before the measured pressure distributions will be used to perform the validation of the panel method. The information acquired during the experiment was in a raw format and in order to obtain the pressure distribution it was necessary to perform post processing of those RAW files. This is an extensive process that is discussed in the Appendix A, some remarks are explained below.

Having such immense amount of information it was necessary to perform an extraction of relevant and meaningful information from the data for the correct comparison between experiment and simulation. Thus, following ways of data choice and analysis from the Work Package 12 and Work Package 13 of the MEXICO and New MEXICO projects [59, 61, 67], it was decided to select and averaged sampling values for the pressure distribution at selected azimuth angles of $0^\circ$, $60^\circ$, $120^\circ$, $180^\circ$, $240^\circ$, and $300^\circ$.

The extraction and averaging of these values was done using Matlab routines that generated one ASCII file for each radial location. This is done for each data point file selected. Those ASCII files contain the 6 azimuth positions and its corresponding value of each pressure sensor along the chord, including the standard deviation of the averaging of the measurements, giving a matrix like
3.6. DATA SELECTION FOR VALIDATION

the one shown in the Appendix D in the Table D1 along with other useful information, like name of the file, rpm, frequency, wind speed, yaw angle, pitch angle, temperature of tunnel and blade, and air density. This file is ready to be used in the comparison. The Matlab routines and a briefly description of each one can be found in the Appendix A.

The information obtained from the numerical simulation is extracted with Tecplot where the slices of the blade are taken at the same radial locations as in the experiment. This information is then exported to ASCII files where the information again ready to be compared with the experiment results. The comparison and the validation of the panel method will be the topic discussed in the following chapter.
Chapter 4

Panel Method Validation

“The purpose of models is not to fit the data but to sharpen the questions.”

Samuel Karlin

4.1 Introduction

In the previous chapter the cases selected for the comparison were mentioned, as well the values for wind velocity, pitch angle and rotational velocity, which corresponding to the design values. In this section each case will contain the specific information related to its conditions. We start with the yawed rotor cases, then we will continue with the dynamic cases, pitch step and step in rotational velocity.

The validation of the multilevel panel method will be made by making a comparison between the numerically obtained pressure results and the experimental pressures. From the cases of the New MEXICO experiment it is possible to obtain the conditions for each simulation. The numerical simulations are done to reproduce the conditions of the experiment as accurate as possible.

The extraction of the information from the New MEXICO database is performed with specialised routines for the correct postprocessing of the information. The routines are written in MATLAB, the code and a general explanation of how the routines extract the information can be found in the Appendix A.
The non-dimensional quantity used to do the comparison is the pressure coefficient $C_p$, which can be defined as

$$
C_p = \frac{p - p_{ref}}{\frac{1}{2} \rho U_\infty^2 + (\omega r)^2},
$$

(4.1)

where $p$ is the absolute pressure over the airfoil surface, $p_{ref}$ the atmospheric pressure, $U_\infty$ is the free stream velocity and $\omega r$ is the angular velocity multiplied by the radial distance of the airfoil section.

The MATLAB routines used for post process the information transform the values obtained in the panel method through the definitions of the equations (2.31) and (2.32) to the same pressure coefficient defined with equation (4.1).

A basic distinction has to be made within the cases. For the yawed rotor cases, the plots for the pressure coefficient are established every 60 degrees of azimuth angle starting at 0°. The pressure values were obtained from averaging the sampling values over all the revolutions at the same azimuth value within a predefined $\Delta \Psi = \pm 1°$. Thus, the average for any station is made inside an arc of 2°. For example for the azimuth location 60° the average will be done between the $\Psi = 59°$ and the $\Psi = 61°$ as shown in Figure 4.1.

Figure 4.1: The locations for the azimuth averaging are 0°, 60°, 12°, 180°, 240°, and 30°, in blue the $\Delta \Psi$ arc for the average, downstream view.
The number of averaging of the samplings inside the blue area will be in function of the value of $\Delta \Psi$, the angular velocity and the sampling frequency (5514 Hz). Assuming that the design rotor speed is used we can compute the distance in degrees between a measurement and the next one.

\[
\left[ \frac{424 \text{ rev}}{\text{min}} \right] \times \left[ \frac{1 \text{ min}}{60 \text{s}} \right] \times \left[ \frac{360^\circ}{1 \text{ rev}} \right] \times \left[ \frac{1}{5514 \text{ Hz}} \right] \approx 0.4613^\circ \quad (4.2)
\]

Therefore, every 0.4613° we will have a value, since the $\Delta \Psi = \pm 1$ we have around 4 values per revolution plus the azimuth angle specified, however, since the azimuth is never aligned exactly with this specification, the number of sampling points can change between 5 and 4 per revolution, changing the number of azimuth sampling values for the averaging. If we multiplied the number of values within this arc times the number of revolutions we obtain the number of points used for the averaging. This value for a case with 36 revolutions will be between 144 and 180 points, with 4 and 5 points per revolution respectively.

For the step in rotational velocity and step in pitch angle the measurements were chosen at specific time instances according to the specific conditions of each case. These time instances will be describe with more detail at its respective section case.

### 4.1.1 Simulation Panels

The multilevel panel method used for the New MEXICO experiment simulated three blades without a nacelle. The blades feature zero trailing edge thickness. Two of the three blades used a coarser grid consisting of 46 panels in radial direction and 30 panels at the cross-section of the blade. The remain blade used a fine grid formed of 92 panels in radial direction and 120 panels in cross-section circumferential direction. The wake follow the corresponding number of panels for the radial direction of their respective blade, thus, wakes corresponding to the coarser grid, have 46 panels in radial direction, while the wake attached to the blade with fine grid has 92 panels in radial direction.

The length of the wake was defined as 10 times the length of the rotor, hence, the wake has a downstream length of 45 m. Wake is a geometrical entity that can go in any direction, however, there is not really physical flow will leave for example in a 90° angle from the trailing edge. In this case we are not doing any wake deformation as can be seen in the Figure 4.2 that the wake follows a helicoid shape.
4.1.2 Effect of Reference Velocity on Pressure Coefficients

The definition of the pressure coefficient for the multilevel panel method in equation (2.32) is different from the definition used in this thesis (equation (4.1)).

The equation in section 2.2.4 gives the pressure coefficient obtained by the multilevel panel method with 

\[ \vec{v}_{ref} = \vec{u}_\infty - \vec{u}_s \],

where \( \vec{u}_s \) is the local velocity of the (blade) surface and includes the contributions originated from the rotational and translational motion, as well as (blade) deformations. In a case of pure rotation 

\[ \vec{u}_s = \vec{\omega} \times \vec{r}, \]

where \( \vec{\omega} \) correspond to the rotational speed around an axis with direction \( \vec{\omega}/|\vec{\omega}| \) and magnitude \( |\vec{\omega}| \) and \( \vec{r} \) is defined by an arbitrary point at the axis and the point on the surface under consideration.

On the other hand, equation (4.1) gives the pressure coefficient used in this thesis and in the New MEXICO experiment with 

\[ \vec{v}_{ref} = \sqrt{U_\infty^2 + (\omega r)^2}, \]

where a single value of \((\omega r)^2\) is chosen for each section. Nonetheless, this is valid only for a curved blade section, so a small deviation could be expected.

The definitions of the reference velocity for the numerical method and for the experiment show a small difference for axi-symmetric cases i.e. a rotor yaw angle of \(0^\circ\) where the \( U_\infty \) and the velocity due to rotation \( \vec{\omega} \times \vec{r} \) are perpendicular to each other (see Figure 4.3a). In this case the deviation of the airfoil section from a constant radius section causes a small difference in the reference velocity and thus also in pressure coefficient.

The difference in the reference velocity for the simulation and experiment is more noticeable in cases of rotor yaw. With rotor yaw, the velocity \( U_\infty \) and the velocity due to the rotation \( \vec{\omega} \times \vec{r} \) are not perpendicular to each other anymore (see Figures 4.3b and 4.3c), thus, the effective wind speed has a different magnitude and direction for most of the rotation.
In the Figure 4.4 is possible to see the effects of this non-dimensionalization on the pressure distribution. The red dashed line indicates the value of the pressure distribution obtained directly with the multilevel panel method (equation (2.32)) and the blue solid line represent the pressure distribution computed according to equation (4.1).

The experimental pressure coefficients are also present with a light blue line and bars indicating standard deviation $\pm \sigma$. The standard deviation is obtained from the averaging of pressure values mentioned in section 4.1. It is possible to see the small error bars in the experimental points. These error bars are more noticeable at 25% and 35% radial location. The most notorious is at 60° azimuth angle on the suction side at 50% chord location. However, is also noticeable that the standard deviation is quite small compared to the pressure distribution. This standard deviation shows an increase when the sampling range of azimuth angles $\Delta \Psi$ increases, although our intention was to maintain the azimuth angle averaging as close as possible to the real value. Therefore, for the following pressure distribution plots the standard deviation is computed and plotted, but the values are quite small and “appear” not to be there. This case is the one with a rotor yaw of 45° angle and at 35% rotor radius which corresponds to the airfoil DU-W2-250.
With this plots we can say that there is no correlation anymore between the plot and the previous experience in 2D. The information generated during the New MEXICO experiment loses this information since it used an approximation related to the Figure 4.3a instead of taking into account the angle due to rotor yaw (represented in Figures 4.3b and 4.3c). Thus the plots need to be adapted to the definition used for the experiment (dark blue lines), this correction will be used along the following plots to do the comparison with the experimental data. This is also understandable since the data experiment has a different definition for the computation of the pressure distribution.

Note that the definition used normally for the numerical method results in a stagnation pressure coefficient of $C_p = 1$. The pressure distribution will be consistent with the local angle of attack, i.e. a higher AoA will always correspond higher suction levels.
Figure 4.4: The effects of the two pressure coefficient definition are demonstrated by the red dashed line and the continuous blue line for the 35\% rotor radius with airfoil DU-W2-250 and yaw rotor of 45° angle.
4.2 Yawed Rotor Cases

4.2.1 Yaw Angle 15°

The first unsteady case selected is the one with the smallest value of rotor yaw and as mentioned before we will follow the design conditions for all the parameters. Thus, this case has a wind velocity of 14.9 m/s, a rotor speed of 425.1 rpm and a pitch angle of \(-2.3°\). The data measurement points that satisfied the conditions were the R52P79D934 and R52P80D936, and from those two points we selected the measurement point R52P80D936 since this one has less malfunctioning sensors, (understand as “malfunctioning” sensors as values with constant value of either 0 or \(\infty\)). The same selection process was applied for the other cases, where the measurement points selected to do the comparison were the ones with the less amount of malfunctioning sensors. Due the small yaw angle this case has a strong axi-symmetric behaviour with some minor changes due the yaw angle.

The following figures shown the pressure distribution for the 6 azimuth locations and their corresponding values along the radius. In general it is possible to see that the numerical predictions better match the cases for the outer locations (82% and 92% radius). The airfoil pressure side in each plot shows a surface pressure distribution that matches the experimental data quite well.

At outboard locations (82% and 92% with the profile NACA64-418) the numerical results agree quite well to the experimental, with a small over prediction of the suction side, however, with a minimal difference that allow us to say that the results are accurately predicted.

At the 60% radial location tested with the Risø A2-21 profile, the pressure distribution at the suction side has a favourable pressure gradient as well as a strong adverse pressure gradient starting at 40% chordwise position. The difference between the numerical result and the experimental data is noticeable, showing an over-prediction in the numerical data for the suction side and a quite good approximation at the pressure side.

At the 25% and 35% (DU-W2-250 profile) radial locations it is possible to spot stronger adverse pressure gradients, specially at the 35% chord wise location. This situation is accentuated with the increase of the blade azimuth angle. The bigger gradient at 180° azimuth angle where the blade speed has a component opposite to the direction of the wind. This is contrary to the top case (0° azimuth) where the blade has a component of velocity in the same direction as the wind.

The inboard location at 25% radius present a less accurate prediction. At this location the numerical simulation and the experiment hardly match, however, the general shape of the pressure distribution is maintained. Is important to mention that the sensors for this radial location have been reported as the ones with greater uncertainty this due the small dynamic pressure of this location [64].

As mentioned in the section 3.5.2, the uncertainty in the pressure sensors and thus in the pressure coefficient is bigger for the smaller radial locations, specifically, 25% and 35%. This uncertainty can explain the difference among the numerical simulation and the experimental information are bigger at these radial locations. On the other hand, for the outer locations (82% and 92%) the uncertainty is lower due the bigger dynamic pressure, showing a smaller difference between the
4.2. YAWED ROTOR CASES

experimental and the numerical data.
Figure 4.5: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 0° and a yaw angle of 15°.
\subsection{Yawed Rotor Cases}

Figure 4.6: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 60° and a yaw angle of 15°.
Figure 4.7: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 120° and a yaw angle of 15°.
4.2. YAWED ROTOR CASES

Figure 4.8: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 180° and a yaw angle of 15°.
Figure 4.9: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 240° and a yaw angle of 15°.
Figure 4.10: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of $300^\circ$ and a yaw angle of $15^\circ$. 
4.2.2 Yaw Angle 30°

The next case is the case for the rotor under 30° yaw. This case has a wind velocity of 14.9 m/s, a rotor speed of 425.1 rpm and a pitch angle of −2.3°. The data point used for this comparison is R44P66D703.

The following figures show the pressure distribution for the 6 azimuth locations, each one with their respective radial locations. This case shows a stronger effect due to the yaw angle when compared to the 15° yaw case.

At the outboard locations (82% and 92%) where the NACA64-418 profile was used, the numerical simulation and the experiment show a good match at the pressure side. The numerical results at the suction side show a bit of over-prediction a situation that changes along the blade trajectory. Despite this difference we can say that the results present a close behaviour to the experimental data.

At 60% rotor radius an interesting effect is shown in the pressure side between 20% and 50% chord. The pressure is lower than ambient pressure. This situation is easier to identify in the azimuth angle 180 (see Figure 4.14c). Where the pressure is lower than 0. Giving a area that after integration will exert force opposite to the lift. The plots with the less effect of this area can be found at the 300 and 0 azimuth angles (Figure 4.11c and 4.14c). Again the simulation maintain an over-prediction on the suction side. This reinforces the idea that no matter at what azimuth location the prediction is done quire accurate.

The two remaining radial locations at 25% and 35% (DU-W2-250 profile) present a better prediction at azimuth locations closer to 180° (Figure 4.14a and 4.14b). Worth to remember is that at this azimuth angles the blade is moving with a component of the opposite to the direction of the wind. It is important to recall the discussion of the subsection 3.5.2 to mention that the sensors for this radial location have been reported as the ones with greater uncertainty, due to the small dynamic pressure of this location [64].
4.2. YAWED ROTOR CASES

Figure 4.11: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 0° and a yaw angle of 30°.
Figure 4.12: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 60° and a yaw angle of 30°.
4.2. YAWED ROTOR CASES

Figure 4.13: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 120° and a yaw angle of 30°.
Figure 4.14: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 180° and a yaw angle of 30°.
4.2. YAWED ROTOR CASES

Figure 4.15: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 240° and a yaw angle of 30°.
Figure 4.16: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 300° and a yaw angle of 30°.
4.2. YAWED ROTOR CASES

4.2.3 Yaw Angle $-30^\circ$

For this case the rotor yaw angle was established at $-30^\circ$. This case has a wind velocity of 14.9 m/s, a rotor speed of 425.1 rpm and a pitch angle of $-2.3^\circ$. The data point used for this comparison is R44P66D703.

This case shows similarities to the $30^\circ$ yaw. However, all the descriptions present a shift of 180 in azimuth angle degrees since now the direction of rotation at the lower part of the rotor (120, 180 and 240 azimuth), instead of moving with a component opposite to the wind direction, moves with a component in the same direction of the wind.

Again, it is possible to see a situation where the outboard locations 82% and 92% (NACA64-418 profile) show a better match with the New MEXICO experiment. However, opposite to the $30^\circ$ case, the best agreement between experiment and numerical results was not a $180^\circ$ but at $0^\circ$ azimuth angle, showing a small over-prediction at the suction side.

At 60% radial location, the pressure distribution we have an overspeed at lower surface between 20% and 50%, however, opposite to the positive $30^\circ$ yaw angle case, the 180 azimuth angle plot (4.20c) is the case with lower effect. Meanwhile the biggest effect is shown in the case of 0 azimuth angle (Figure 4.17c).

Finally, at radial locations 25% and 35% the prediction for the pressure side follows correctly the experimental information. However, the suction side presents the same over-prediction as the case before.
Figure 4.17: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of $0^\circ$ and a yaw angle of $-30^\circ$. 
4.2. YAWED ROTOR CASES

Figure 4.18: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 60° and a yaw angle of −30°.
Figure 4.19: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 120° and a yaw angle of −30°.
4.2. YAWED ROTOR CASES

Figure 4.20: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 180° and a yaw angle of −30°.
Figure 4.21: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 240° and a yaw angle of −30°.
4.2. YAWED ROTOR CASES

Figure 4.22: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of $300^\circ$ and a yaw angle of $-30^\circ$. 
4.2.4 Yaw Angle 45°

The last yaw case is the one with the largest value of yaw, 45°. The design conditions are: wind velocity of approximately 15 m/s, rotor speed of 424.5 rpm and a pitch angle of −2.3°. The data point with these specific conditions correspond to R52P83D949. This resulted to be the most interesting of the yaw cases since the conditions shown in the previous cases are accentuated more.

This case also allows us to see the effects of the yaw in the pressure distribution with more detail since it is the more extreme yaw case where the effective wind velocity direction is not following the design conditions at several azimuth locations. The predictions done in this case will some idea on how well the panel method used can be for the prediction under these unsteady flow conditions.

Again, the information of the 6 azimuth locations can be found if the following figures. The locations show a good match between the numerical solution and the experimental data. The numerical simulation follows this match with the experimental data at any azimuth angle. The outer radial sections (82% and 92%) show a closest prediction than the inner radial locations, (see subsection 3.5.2).

In general the matching between the numerical results and the experiment values is good for most of the azimuth angles and radial locations. With some exceptions like in the case of 25% where the simulation does not match the values of the experiment, the panel method does follow the general behaviour of the pressure distribution with changing azimuth angle. The best example of how the multilevel panel method follows the experimental behaviour is in the Figure 4.26a, where the simulation and the experiment show a good match independently of the extreme conditions presented and the uncertainty at this radial station. This location coincides with the idea of having the blade moving with a component completely opposing the wind velocity.
4.2. YAWED ROTOR CASES

Figure 4.23: 

The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 0° and a yaw angle of 45°.
Figure 4.24: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of $60^\circ$ and a yaw angle of $45^\circ$. 

(a) 25% Rotor radius, DU-W2-250.

(b) 35% Rotor radius, DU-W2-250.

(c) 60% Rotor radius, RisoA2-21.

(d) 82% Rotor radius, NACA64-418.

(e) 92% Rotor radius, NACA64-418.
Figure 4.25: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 120° and a yaw angle of 45°.
Figure 4.26: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 180° and a yaw angle of 45°.
4.2. YAWED ROTOR CASES

Figure 4.27: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 240° and a yaw angle of 45°.
Figure 4.28: The New MEXICO rotor experimental pressure data compared with the numeric results from the multilevel panel method. These plots represent the pressure coefficient at the five stations with an azimuth angle of 300° and a yaw angle of 45°.
4.3 Dynamic Inflow Cases

The following two cases present situations where a parameter of the experiment changes over time. The two cases chosen for this are pitch angle and rotor speed. The simulation starts from the axi-symmetrical case with design conditions that has already reached steady state, then the free parameter either pitch angle or rotor speed is modified rapidly given enough time to reach steady state, and then returns back to the original state.

4.3.1 Step in Blade Pitch Angle

The first dynamic case is the pitch ramp, in which the RPM speed is maintained constant while the pitch angle changes from $-2.3^\circ$ to $5^\circ$ and then back to $-2.3^\circ$. Figure 4.29 shows the mentioned step in pitch angle performed during the datapoint R54P88D978. The blue line is the filtered and scaled signal as obtained during this datapoint. The dynamic cases were recorded for a period of nearly 15 s, contrary to the 5 s for the yawed rotor cases. We wanted to see the effect of this dynamic change on the pressure distribution, and in order to achieve this four time sampling points were chosen. These four points are indicated in the figure below by the dark blue lines.

![Figure 4.29: Pitch step plot for point R54P88D978 (blue line), plot approximation (red line), time instances selected $t_1$, $t_2$, $t_3$ and $t_4$ (vertical blue lines).](image-url)
CHAPTER 4. PANEL METHOD VALIDATION

The four time points are divided in two groups. The first group consists of $t_1$ and $t_3$ and represent the aerodynamic equilibrium (or steady state) of the wind turbine, since the wind turbine has already has enough time to build the wake behind it.

The second group of the time points consists of $t_2$ and $t_4$ that represent the moment of a dramatic change in the conditions, in this case the change in pitch angle. The points $t_2$ and $t_4$ are located before the pitch step finish allowing us to see what is happening while the blades are still rotating over their radial axis. Even when the blade is already in a steady position the wake has not yet reached the equilibrium point at this moment. Therefore, these points will be good example of a dynamic flow.

Since the simulation has to represent the experiment the most accurate posible, it was necessary to approximate the pitch angle behaviour during this case. The red line in the Figure 4.29 represents the approximation made for this case. The multilevel panel method performs a linear interpolation of pitch angle values. In order to simulate the pitch case it is just necessary to indicate the time and pitch value for each point.

In the figures below is possible to see the pressure distributions obtained for the four time instances. For each time instance the surface pressure distribution at 5 radial locations are plotted. The first thing to notice is a reduction in the pressure distribution at all radial locations between $t_1$ and $t_2$, this is more evident at 82% and 92% radial locations, see Figures 4.30d and 4.30e for $t_1$ and Figures 4.31d and 4.31e for $t_2$ at this case the lift shows a drastically reduction. An adverse pressure gradient seems to be decrease at the 60% radius between 20% and 40% of the chord, (Figures 4.30c and 4.31c).

At 25 and 35 radial locations the simulation shows a small over-prediction, however the prediction adjust well to the numerical experiment Figure 4.30a and 4.30b for $t_1$ and Figure 4.31a and 4.31b for $t_2$, even for this radial locations with aa lot of uncertainty associated.
4.3. DYNAMIC INFLOW CASES

Figure 4.30: Pressure coefficient comparison between New MEXICO experiment and multilevel panel method, for dynamic pitch step at $t_1 = 4.65s$
Figure 4.31: Pressure coefficient comparison between New MEXICO experiment and multilevel panel method, for dynamic pitch step at $t_2 = 5.20s$
4.3. DYNAMIC INFLOW CASES

Figure 4.32: Pressure coefficient comparison between New MEXICO experiment and multilevel panel method, for dynamic pitch step at $t_3 = 10.85s$
Figure 4.33: Pressure coefficient comparison between New MEXICO experiment and multilevel panel method, for dynamic pitch step at $t_4 = 11.25s$
4.3. DYNAMIC INFLOW CASES

4.3.1.1 Sensors With Time Delay

During the analysis of the data points, a small animation showing the evolution in time of the pressure distribution shows that some points have a time delay possibly associated with the sensors during the data acquisition. The points showing this problem are the following.

- At the 25% radial location the point in the suction side at 0.940% of x/c.
- At the 82% radial location the point in the pressures side at 12.120% of x/c.
- At the 92% radial location the point in the pressures side at 18.99% of x/c.

The point at 7.130% chord for the 82% radial location was found to have a zero value during all the experiment.
4.3.2 Step in Rotor Angular Velocity

The second dynamic case is the step in rotational velocity. The blade pitch angle is kept constant while the rotor speed reduces from 424 rpm to 324 rpm and then changes quickly to 424 rpm. Figure 4.34 shows the rotor speed performed during the datapoint R54P88D981. The scaled signal obtained during the experiment is shown in light blue. Like in the pitch ramp, we are interested in see the behaviour of the pressure coefficients under a dynamic range and compare it with the steady state ones. Therefore time sampling points \( t_1, t_2, t_3 \) and \( t_4 \) were selected as indicated in the figure by the dark blue lines.

The moments \( t_1 \) and \( t_3 \) represent the steady conditions since at these points the wind turbine has already enough time to build the wake behind. The moments \( t_2 \) and \( t_4 \) represent the moment of change in the rotor speed step. These sampling points are located before the rotor step finish, allowing us to see what is happening while the rotor velocity is changing. This also implies that while the blades are already in a new condition the wake has not yet react and adapt to this new moment. Therefore this sampling points will be an excellent representation of a dynamic flow that we are looking for.

Figure 4.34: Step in rotor speed step for data point R54P88D981 (blue line), and for the numerical simulation (red line). Selected ime instances are \( t_1, t_2, t_3 \) and \( t_4 \) (dark blue lines).
4.3. *DYNAMIC INFLOW CASES*

4.3.2.1 *Rotor Speed Conversion*

The multilevel panel method requires the specification of the angular position of the rotor blades as a function of the time. The experimental data however provides us only with the rotational speed as function of time.

As an example, if we desire to know the number of degrees traveled by a blade when the rotor has a speed of 424 rpm and it rotates for 3 s we can do the following calculations

\[
424 \text{ rpm during } 3 \text{ s} \rightarrow 424 \frac{\text{rev}}{\text{min}} \times \left[ \frac{1 \text{ min}}{60 \text{ s}} \right] \times [3 \text{ s}] \times \left[ \frac{360^\circ}{1 \text{ rev}} \right] = 7632^\circ \text{ after 3 s} \quad (4.3)
\]

The points to express this rotation will be specified on the input files as

<table>
<thead>
<tr>
<th>Time</th>
<th>Degrees of rotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>7632</td>
</tr>
</tbody>
</table>

However, this example assumed that the value of the rotational velocity remained constant during the period of time. In Figure 4.34 it is possible to see that the rotational velocity is a function of the time, thus the degrees traveled at a certain rotor speed will be also dependent of time. We will therefore use a more sophisticated method that can guarantee the same result no matter which instance of time we select. To do so a Matlab program was written that integrates the rotational speed over time and gives the angular position as result base on the following analysis.

The number of degrees traveled at any period of time will be defined as

\[
\chi = \int_{t_1}^{t_2} f(t) \, dt \times \left[ \frac{1 \text{ min}}{60 \text{ s}} \right] \times \left[ \frac{360^\circ}{1 \text{ rev}} \right] \quad (4.4)
\]

where: \( t_1 \) is the initial time of the period

\( t_2 \) is the final time of the period

\( f(t) \) is the rotor speed between \( t_1 \) and \( t_2 \) in \( \frac{\text{rev}}{\text{min}} \)

\( \chi \) is the amount of degrees travelled in the period of time.

In order to simplify the computation of these values, the points in the approximation curve are chosen such a way that \( f(t) \) can be approximated by a linear function of the form \( y = mx + b \) and since we do know the points for \( x \) (the time values) and \( y \) (the rotor speeds) is possible to define \( f(t) \) for any two points in the curve, therefore we can calculate the area under the curve for those two points as show in the Figure 4.35.
The values obtained from these computation are now used as input for the multilevel panel method and allow us to simulate the rotational velocity step as close as possible. This approximation could of course be done taking less points to reduce the computation, however the procedure remains unchanged.

In the figures below the pressure distributions are shown for all the selected time instances. The first noticeable thing is the increase in the pressure distribution due the reduction of the RPM, this can be seen comparing the Figure 4.37 with the Figure 4.38. However in the numerical solution at $t_2$ the values for the pressure side seem to be higher than the experiment, and the pressure for the suction side are quite bigger than the experiment, however, at the pressure side the numerical values seems to agree with the experimental values.

At $t_4$ when the rotational velocity is closer to the design conditions, at 424 rpm the pressure distribution goes back to the original values, and the simulation and the experiment agree quite well at the pressure side of the blade, since the pressure side does not suffer from the viscous effects.

For the time instance $t_2$ and $t_3$, the experiment show separated flow as indicated by the near-constant pressure coefficient at the at part of the suction side (see Figures 4.38c, 4.38e, 4.39c, 4.39e). For $t_2$ and $t_3$ instances is noticeable that the multilevel panel method does not model viscous flow effects therefore for this locations is predicts a higher lift that the on seen in the experiment.
In the Figure 4.36 the changes in the angle of incidence due to the change in the rotational velocity are shown. Figure 4.36a corresponds to the instances $t_1$ and $t_4$ and shows the velocity triangle for 424 rpm and Figure 4.36b corresponds to the time instances $t_2$ and $t_3$ and it shows the velocity triangle at 324 rpm. Is is possible to see the changes in the incidence velocity are provoked by the change in the rotational velocity. Therefore the lift obtained by the experimental and the numerical simulation increase for the lower rotational speed. However, the lift predicted by the numerical experimental is bigger since it does not take viscous effects in account.

Figure 4.36: Velocity triangle shows the change at the incidence wind due to the change from 424 rpm to 324 rpm.
Figure 4.37: Pressure coefficient comparison between New MEXICO experiment and multilevel panel method, for dynamic pitch step at $t_1 = 3.63s$
Figure 4.38: Pressure coefficient comparison between New MEXICO experiment and multilevel panel method, for dynamic pitch step at $t_2 = 5.68s$. 

(a) 25% Rotor radius, DU-W2-250. 
(b) 35% Rotor radius, DU-W2-250. 
(c) 60% Rotor radius, RisøA2-21. 
(d) 82% Rotor radius, NACA64-418. 
(e) 92% Rotor radius, NACA64-418.
Figure 4.39: Pressure coefficient comparison between New MEXICO experiment and multilevel panel method, for dynamic pitch step at $t_3 = 9.67$ s

(a) 25% Rotor radius, DU-W2-250.
(b) 35% Rotor radius, DU-W2-250.
(c) 60% Rotor radius, RisøA2-21.
(d) 82% Rotor radius, NACA64-418.
(e) 92% Rotor radius, NACA64-418.
4.3. DYNAMIC INFLOW CASES

Figure 4.40: Pressure coefficient comparison between New MEXICO experiment and multilevel panel method, for dynamic pitch step at $t_4 = 11.88s$
Chapter 5

Conclusions and Recommendations

“I didn’t have time to write a short letter, so I wrote a long one instead.”

Mark Twain

5.1 Conclusions

In this thesis the multilevel panel method developed in [48] was validated using wind tunnel experiments on a scaled wind turbine model. These wind tunnel experiments were performed in the Large Scale Low-Speed Facility (LLF) of the German-Dutch Wind Tunnel Organisation (DNW) [59, 62, 66]. For the validation of the panel method several dynamic test cases were selected; rotor yaw angles $-30^\circ$, $15^\circ$, $30^\circ$, and $45^\circ$ and dynamic inflow cases where step in pitch and step in rotor speed were performed. All the cases where selected to be performed under design conditions for the remaining parameters, that is, rotor speed of 424 rpm, wind speed of 15 m/s and pitch angle of $-2.3^\circ$.

The experiment was provided with pressure sensors at 5 radial locations at 25%, 35%, 60%, 82% and 92% radius. The surface pressure distribution at these 5 locations were compared with the results obtained from the numerical simulation at the same conditions.

The comparison between the numerical simulation and the experimental data has to be done with the same pressure coefficient for both. Therefore it was necessary to adjust the definition of the pressure coefficient from the multilevel panel method. This allow us to see the difference between both definitions. The initial computation of the pressure coefficients for the numerical experiment does not have correlation with the New MEXICO since the values for the New MEXICO experiment are a 2D representation at the section, while the pressure coefficient from the multilevel panel method takes in account the changes related to the rotor Yaw. It has been show how the non-dimensionalization of the pressure coefficient could affect the interpretation of the pressure
coefficient. Finally an adjust from the numerical simulation was done to have the same definition for all the comparisons.

The simulations show a good match between the experimental data and the numerical results at the most outboard stations (60%, 82% and 92%) for attached flows. The inner most sections (25% and 35%) show a less satisfactory match between experimental data and numerical results. However, these section suffer from less reliable experimental results as a results of the accuracy of the absolute pressure measurements.

The cases with the rotor yaw show good agreement between panel method results and experimental values. At the suction side of the airfoil sections the results differ somewhat due to the inviscid nature of the panel method and the lack of the effects of the boundary layer on the pressure distribution.

The 45° experimental yaw case show the most strong effects of rotor yaw on the surface pressure distribution. These effects are quite well reproduced by the panel method, showing that at low angle of attack the simulation is quite close to the experiment.

The 30° and −30° rotor yaw cases demonstrates a symmetrical behaviour. The blade azimuth locations in the top (12 o’clock position) and in the bottom (6 o’clock position) show mirrored results.

The dynamic pitch step cases show that the numerical simulations adapt with exceptional speed to the experimental results, showing that at any time instance the simulation is close to the experiment. Also in these cases the suction side presents a lower pressure than the experiment.

The rotor speed test cases show a logical increase in the angle of attack when the rotational velocity is reduced. This behaviour is strongly accentuated at the inner sections of the rotor (25% and 35%) where at the pressure suction peak in the experiment disappears due to flow separation. The suction peaks reappear when the design conditions are met again.

Finally we can conclude that the multilevel panel method can predict the behaviour of the pressure distributions, and therefore also the loading quite well, even without taking in account the viscous effects as long as the rotor exhibits no flow separation.

5.2 Recommendations

In this thesis all numerical simulations were performed with a prescribed wake convection speed. No wake deformations due to self-influence were taken into account. It is recommended to repeat the validation of the panel method with the same set of experimental data and allowance for the wake to deform. Such simulations might reduce, or even remove, the small pressure differences at the trailing edge upper and lower sides due to a better handling of the Kutta condition.

The simulations were performed using three blades floating in the air, therefore, no influence of
5.2. **RECOMMENDATIONS**

other turbine components are present. It is recommended to include other components of the MEXICO wind turbine. For example the hub, nacelle or tower. Those simulations might allow to see the effects of a full modelled turbine.

Both, the MEXICO and New MEXICO experiments generated a considerable amount of cases with diverse conditions and situations, therefore, the selection of other cases to be simulated is recommended, for example, in the New MEXICO database a standstill run was performed, that could be an interesting case for the multilevel panel method to simulate and see the effects on the twisted blade.

The multilevel panel method tool has proven to be an excellent tool for the simulation of aerodynamic cases. A recommendation for the code itself from the users point of view is related to the way in which the angular rotation is implemented. Since it require the transformation form one quantity to another (from rpm to degrees), therefore it will be useful to have the opportunity to define the rotation, either in angular velocity or degrees travelled for time instance.
References


REFERENCES


REFERENCES


[43] Energy Institute at The University of Texas at Austin. Executive Summary : The Full Cost of Electricity.


REFERENCES


Appendix
A Matlab Routines

A.1 File Processing file

```matlab
% ************************************************************************
% NewMEXICO
% File Processing.m
%
% Routine to extract information from binary (*.hdf5) file and generate a
text file (*.txt) with the azimuth values or the time-series values of
the information extracted is basically the pressure coefficients and
at their respective locations.
%
% Is possible to specify the azimuth locations where the extraction is
desire, as well as a value ∆ which defined the spectrum values
taken to perform the averaging.
%
% This routine allows selecting one or several files. At the end of the
routine the number of files successfully processed and the ones that
failed is shown, and a report with the list of such files is generated.
%
% This routine makes used of the following functions to perform all the
computations.
%
% aux_Azimuth_average.m
% aux_time_series_columns.m
% aux_time_series_rows.m
%
% The selection of the processing auxiliary function depend on the
options selected for type of analysis, either azimuth or time, and for
this last one, either to show results in columns or row.
%
% University of Twente
% February / December 2017
%
% ************************************************************************
% Programmer: E Prieto
%
%clc; clear all; close all
tic
%
% Initial variables
bCn = 0;  % Cn/Ct Computation
Repro = 0;  % 0 = just canonical values, 1 = canonical and ...
            % reproducibility
ReOrdering = 1;  % 0 = off, 1 = on re-order the data so they can be plot ...
            % on gnuplot without looking weird
%
% Types of analysis.
T_Analysis = 2;  % 1 = Azimuth,  2 = Time;
```

100
Order = 2; % 1 = Columns, 2 = Rows; Rows is better :D but i ... need Columns for plots in gnu

% Select file (or files) to process and check how many files do you have or % if there is no file it will show a message.

[Filename, PathName, -] = uigetfile('*.hdf5','MultiSelect', 'on');

if iscell(Filename)
    F=length(Filename);
    filename=Filename;
elseif ischar(Filename)
    F=1;
    filename{1}=Filename;
else
    fprintf('
>> No file selected, ending program.
')
    clear Order PathName Pulse Repro SpecialPlot Stres T_Azimuth bCn Filename
    return
end

fprintf(['
>> ',num2str(F), ' files selected 
'])

% Initiable variables
s=0; % Success file counter
f=0; % Fail file counter
oc=0; % One column
sjit=0; % Jitter exceeds limit
Suc{1,1}=0; % Success file names
Fail{1,1}=0; % Fail file names
OneCol{1,1}=0; % One column
SuccJitter{1,1}=0; % Jitter exceeds limit
% Path for save the summary of the status
ResultsPathAscii = ...
'/Users/Xientikiko/Dropbox/Thesis/Results/NewMEXICO/ascii';

% Routine for each file.
for i=1:F
    NameFile=filename{i};
    DataFile = strcat(PathName, NameFile);

    % Run file processing ...
    auxiliar

    if T_Azimuth == 1
        Status = aux_Azimuth_average(DataFile,bCn,Repro,ReOrdering);
        Add = 'aux_Azimuth_average';
    elseif T_Azimuth == 2
        if Order == 1
            Status = aux_time_series_columns(DataFile,bCn,Repro,ReOrdering);
            Add = 'aux_time_series_columns';
        elseif Order == 2

101
Status = aux_time_series_rows(DataFile,bCn,Repro);
    Add = 'aux_time_series_rows';
end
end

% Evaluate outcome and status of file.
if Status == 1
    fprintf(['>> File ', NameFile, ' Succeed 

']);
s=s+1;
    Succ{s,1}=NameFile;
elseif Status == 2
    fprintf(['>> File ', NameFile, ' Fail cause no TunnelInfo 

']);
f=f+1;
    Fail{f,1}=NameFile;
elseif Status == 3
    fprintf(['>> File ', NameFile, ' Fail cause incorrect lp signal ... 

']);
f=f+1;
    Fail{f,1}=NameFile;
elseif Status == 4
    fprintf(['>> File ', NameFile, ' Succeed as Standstill 

']);
oc=oc+1;
    OneCol{oc,1}=NameFile;
elseif Status == 5
    fprintf(['>> File ', NameFile, ' not procces because is a ... pressure zero run 

']);
sjit=sjit+1;
    SuccJitter{sjit,1}=NameFile;
end

% if Status == 15
% return
end

fprintf(['>> From ',num2str(F), ' files selected 
','...
','num2str(s),' files Succeed 
','num2str(f), ' files failed 
','num2str(oc), ' were Standstill 

']);

% Save summary file with the results of the proccesing.
FileName=['Summary file' Add];
FileNamet=strcat(FileName,'.txt');
file_name = fullfile(ResultsPathAscii,FileNamet);

fid=fopen(file_name,'w');
%fprintf(fid,'This is a List of processed files\n\n');
fprintf(fid,'Number of Analyzed files
Succeed Files
Failed files
Succeed as Standstill

');

% Print summary file with the results of the processing.
fprintf(fid,'Pressure zero run : %d
',sjit);

if Succ{1,1} \neq 0
    fprintf(fid,'\r\n\n');
    fprintf(fid,'Succeed files\r\n');
    [nrows,\_] = size(Succ);
    for row = 1:nrows
        fprintf(fid,'\$s\r\n',Succ{row,:});
    end
    fprintf(fid,'\r\n');
end

if Fail{1,1} \neq 0
    fprintf(fid,'\r\n\n');
    fprintf(fid,'Fail files\r\n');
    [nrows,\_] = size(Fail);
    for row = 1:nrows
        fprintf(fid,'\$s\r\n',Fail{row,:});
    end
    fprintf(fid,'\r\n');
end

if OneCol{1,1} \neq 0
    fprintf(fid,'\r\n\n');
    fprintf(fid,'Standstill \r\n');
    [nrows,\_] = size(OneCol);
    for row = 1:nrows
        fprintf(fid,'\$s\r\n',OneCol{row,:});
    end
    fprintf(fid,'\r\n');
end

if SuccJitter{1,1} \neq 0
    fprintf(fid,'\r\n\n');
    fprintf(fid,'Pressure zero run \r\n');
    [nrows,\_] = size(SuccJitter);
    for row = 1:nrows
        fprintf(fid,'\$s\r\n',SuccJitter{row,:});
    end
    fprintf(fid,'\r\n');
end

fclose(fid);
clear Add ans bCn f F Fail fid file_name filename Filename FileName ...
    FileName i NameFile n rows oc OneCol Order Pulse Repro Result row s ...
    sjit Stres Succ SuccJitter T_Analysis T_Azimuth

toc
A.2 RPD name extraction

```matlab
function [R, P, D] = RPDFilename(Filename);
% function to extract run, polar and dpn number from a Filename
% input is a string
% output is a string
R = '0';
P = '0';
D = '0';
if length(Filename)==0
    return
end
FilenameL = upper(Filename);
i = strfind(FilenameL,'.');
if ~isempty(i)
    FilenameL = FilenameL(1:i(1)-1);
end
ir = strfind(FilenameL,'R');
ip = strfind(FilenameL,'P');
id = strfind(FilenameL,'D');
if ~isempty(ir)
    if ~isempty(ip)
        R = FilenameL(ir(1)+1:ip(1)-1);
    elseif ~isempty(id)
        R = FilenameL(ir(1)+1:id(1) -1);
    else
        R = FilenameL(ir(1):length(FilenameL));
    end
end
if ~isempty(ip)
    if ~isempty(id)
        P = FilenameL(ip(1)+1:id(1)-1);
    elseif ~isempty(id)
        P = FilenameL(ip(1):id(l) -1);
    else
        P = FilenameL ;
    end
end
if numel(P) == 2
    P = strcat('00',P);
else
    P = strcat('0',P);
end
if ~isempty(id)
```

104
D = FilenameL(id(1)+1 : length(FilenameL));
end
A.3 Arrange

function y = sortcoord(source,target1,target2)
% function returns indices of first target values in source ascending
% and in decending order for second target.
% source = 1 x n array with all values
% target1 = 1 x k array with first part of source value
% target2 = 1 x l array with second part of source value
% k + l = n
% function can be used to get sorted coordinates for a closed line x-y plot

for i=1:length(target1)
    y(i)=find(source == target1(i));
end
offs=length(y);

for i=1:length(target2)
    y(offs+i)=find(source == target2(length(target2) + 1 - i));
end
A.4 Azimuth Average

```matlab
function Status = aux_Azimuth_average(DataFile,bCn,Repro,ReOrdering)
% ************************************************************************
% NewMEXICO
% aux_Azimuth_average.m
% Routine to extract and process the information stored in the hdf5
% file, corresponding to the calibrated pressure. This routine is divided
% on three big sections, the setup, the computation and the saving.
% At the end of this routine txt files are save with the information of
% the pressure coefficients, ordered in columns respect the Azimuth
% locations defined.
% It is possible to activate or deactivate the computation of some values
% like Cn's or Ct's, the stress values or just the canonical values, or
% save the txt file, the default option is all turn on, this means:
% bCn = 1
% Repro = 1
% Stres = 1
% bAscii = 1
% The routine save one file for each span position, with the name of the
% file R###P###D###_Span_###
% University of Twente
% February / December 2017
% ************************************************************************
% Programmer: E Prieto

%% Starting settings (SETUP)
% Clear all old data
clear Cp*
clear Pres*
clear Stress*
clear q1*
clear Blade*
clear Cn*
clear Ct*
clear Puls*
clear jit*
clear FileNamet
clear FileNamez
clear RFiles

% Initial variables
Status = 1; % Assume success on process.
PulseOk = 1; % Assume the signal pulse is ok.
bAscii = 1; % 1 - Results are saved in ascii file
```

107
%% Select locations for require directories (SETUP)

% Directory settings

% ResultsPathAscii = Directory to store the azimuth results
ResultsPathAscii = ...
    '/Users/Xientifiko/Dropbox/Thesis/Results/NewMEXICO/ascii/Azimuth_average';

% TunnelInfoPath = Directory to extract the tunnel information (.dat file)
TunnelInfoPath = ...
    '/Users/Xientifiko/Desktop/Thesis/Beehub/home/MexNet/NewMexico/DNW/POL/Steaky_Data';

%% Wind turbine settings (SETUP)

% Wind Turbine Settings

% This values can be modified according to the necessities.

Span = 2.25; % meters
AzStations = [0 60 120 180 240 300]; % Azimuth station in degrees six ... stations default.
Delta=1; % range for the azimuth arc 1 deg default

%% Read the micfile sheet (excel file with information about the sensors) ... (SETUP)

% Excel micfile reading

% Location on computer of the 'micfile' file information, for instance ...
    'K:\documents\micfile2.xls'

Mic=xlsread('/Users/Xientifiko/Desktop/Thesis/Beehub/home/MexNet/NewMexico/NewMexico_datapoints.xlsx','micfile');
Chn1P = Mic(:,3)==1; % 1p sensor channel from mic file.
ChnT = Mic(:,3)==5; % Temperature sensor channel from mic.

% Selection of channels that contain pressure sensors information
% This Following notation is used
% (:,3) = Type of sensor  3 = Pressure
% (:,5) = Span position 25, 35, 60, 82 or 92
% (:,13) = Status of sensor, 1 = working, 0 = Not working
% (:,6) = Number of blade 1,2 or 3

% Type sensor  Section  Status ok?  Blade
PresChan25_1 = find(Mic(:,3)==3 & Mic(:,5)==25 & Mic(:,13)==1 ); % 27 Sensors for this location
PresChan35_1 = find(Mic(:,3)==3 & Mic(:,5)==35 & Mic(:,13)==1 ); % 27 Sensors for this location
PresChan60_1 = find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,13)==1 & ...
    Mic(:,6)==1 ); % 8 Sensors for this location
PresChan60_2 = find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,13)==1 & ...
    Mic(:,6)==2 ); % 24 Sensors for this location
PresChan60_3 = find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,13)==1 & ... 
   Mic(:,6)==3 ); % 4 Sensors for this location
PresChan82_2 = find(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,13)==1 & ... 
   Mic(:,6)==2 ); % 4 Sensors for this location
PresChan82_3 = find(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,13)==1 & ... 
   Mic(:,6)==3 ); % 24 Sensors for this location
PresChan92_3 = find(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,13)==1 ); ... 
   % 25 Sensors for this location

% Read wind tunnel data (SETUP)
% *************************************************************************
% Tunnel Turbine Settings
% *************************************************************************

% Name extraction.
[~,FileName,~] = fileparts(DataFile);
% using function rpdfilename obtain [r], [p] and [d] values
[~, p ,d] = RPDFilename(FileName);
% Filename wind tunnel data POLPPP
NameFile = strcat(TunnelInfoPath,'/POL','p','.DAT');
% Verify if file has representative the TunnelInfo
if exist (NameFile,'file')==2
    fid = fopen(NameFile);
    DataInf = textscan(fid,'%f','HeaderLines',8,'Delimiter',',');
    POLtunneldata = DataInf{1,1};
    POLtunneldata = reshape(POLtunneldata,31,[]);
    POLtunneldata = POLtunneldata';
    D=str2double(d);
    Index_D = POLtunneldata(:,1)==D;
    TunnelData = POLtunneldata(Index_D,:);
% Information contained in the tunneldata file, the "commented" values
% can be activated to obtain the specific information.

% PNT = TunnelData(1); % [-] Data point number
% Time = TunnelData(2); % [hhmmssxxx] Time
Qinf = TunnelData(3); % [Pa] Freestream dynamic ... pressure
RHOinf = TunnelData(4); % [kg/m3] Freestream air density
Vinf = TunnelData(5); % [m/s] Freestream tunnel speed
Beta = TunnelData(6); % [?] Yaw angle
Pitch = TunnelData(7); % [?] Blade pitch angle
Tinf = TunnelData(8); % [? K] Freestream tunnel ... temperature

% Fx_Rot = TunnelData(9); % [N] Force in x-direc ... on (model coordinate system)
% Fy_Rot = TunnelData(10); % [N] Force in y-direc ... on (model coordinate system)
% Fz_Rot = TunnelData(11); % [N] Force in z-direc ... on (model coordinate system)
% Mx_Rot = TunnelData(12); % [Nm] Moment around ... x-axis (model coordinate system)
% My_Rot = TunnelData(13); % [Nm] Moment around ... y-axis (model coordinate system)
% Mz_Rot = TunnelData(14); % [Nm] Moment around ... z-axis (model coordinate system)
N_Rotor = TunnelData(15); % [rpm] Rotor onal speed of rotor

% Lambda = TunnelData(16); % [-] Tip speed ratio of rotor

% X_Trav = TunnelData(17); % [m] x-coordinate of traverse (tunnel coordinate system)

% Y_Trav = TunnelData(18); % [m] y-coordinate of traverse (tunnel coordinate system)

% Pinf = TunnelData(19); % [Pa] Freestream static pressure

% PTinf = TunnelData(20); % [Pa] Freestream total pressure

% dPitot = TunnelData(21); % [Pa] Pitot tube dynamic pressure (velocity verification only)

% VPitot = TunnelData(22); % [m/s] Pitot tube velocity reading (velocity verification only)

% Current = TunnelData(23); % [-] Generator torque (blade-off only)

% Ceil1 = TunnelData(24); % [Pa] Collector static pressure (Figure 17)

% Ceil2 = TunnelData(25); % [Pa] Collector static pressure (Figure 17)

% Star1 = TunnelData(26); % [Pa] Collector static pressure (Figure 17)

% Star2 = TunnelData(27); % [Pa] Collector static pressure (Figure 17)

% Port1 = TunnelData(28); % [Pa] Collector static pressure (Figure 17)

% Port2 = TunnelData(29); % [Pa] Collector static pressure (Figure 17)

% Floor1 = TunnelData(30); % [Pa] Collector static pressure (Figure 17)

% Floor2 = TunnelData(31); % [Pa] Collector static pressure (Figure 17)

fclose(fid);

else

Status = 2; % Fail cause no TunnelInfo

return

end

% Read all available Pressure calibrated data (SETUP)

% ********************************************************************************
% Pressure Calibrated Data *.hdf5 files
% ********************************************************************************

[~,NameFile,~] = fileparts(DataFile);
NameFile = strcat('/',NameFile);
PressureData = h5read(DataFile,NameFile);
[NumSam,NumSensors]=size(PressureData);

% Pulse Analysis (COMPUTATIONS)

% ********************************************************************************
% Pulse Analysis 1p signal
% ********************************************************************************

% Verifying is the 1p signal is correct.

if NumSensors < 161
Status = 5;
return
Pulse = PressureData(:, Mic(Chn1P, 12));
Pulse = Pulse .* -1;
PulseD = diff(Pulse); % Pulse signal PulseD = Pulse(n) - Pulse(n-1)
if max(PulseD) < 3 % If no pulses are present
    if V_inf > 0
        Status = 4; % Either pressure zero run or wrong 1p signal
    else
        Status = 5;
        return
    end
end
PulseOk = 0;
if PulseOk == 1
    PulsMax = (max(PulseD)) / 2;
    [~, PulseI] = findpeaks(PulseD, 'MinPeakHeight', PulsMax); % Pulse at ...
    NoPuls = length(PulseI);
    PulseID = diff(PulseI);
    Rpm = N_Rotor;
    Frot = Rpm * (1/60); % Hz speed estimates
    clear Pulse;
    if (exist('PulseD', 'var') > 0)
        clear PulseD;
    end
else
    Rpm = N_Rotor;
    NoPuls = 2;
end

%% Processing tunnel parameters (COMPUTATIONS)
% ************************************************************************
% Tunnel parameters, for the compute of the qi's
% ************************************************************************
% Calculate temperature
T = mean(PressureData(:, Mic(ChnT, 12))); % Average Temperature data, and ...
    Temperature Constants for the compute of the real Temperature, this ... 
    values could be referring to stagnation T.
cT1 = 1 / 83.4388;
cT0 = -259.74;
T = cT1 * T + cT0;
omega = Rpm / 60 * 2 * pi; % Omega Angular velocity (rad/s)
rhot = RHOinf; % Air density rho

% Cps VER 2
% % Qi per station (Pascales)
qi25 = 0.5 * rhot * ((omega * 0.25 * Span)^2);
qi35 = 0.5 * rhot * ((omega * 0.35 * Span)^2);
qi60 = 0.5 * rhot * ((omega * 0.60 * Span)^2);
% Extraction of azimuth position for pressures (COMPUTATIONS)
% **************************************************************************
% Azimuth Angles computation
% **************************************************************************
% Determine sample indexes for the different blades
AzB1=zeros(1,NumSam);
AzB2=zeros(1,NumSam);
AzB3=zeros(1,NumSam);
PPD=zeros(1,NoPuls-1); % Pulse per Degree
if PulseOk==1
    for i=2:NoPuls
        PPD(i-1)=(360/PulseID(i-1));
        Sample_AzB1=PPD(i-1)*(0:PulseID(i-1)-1); % the term + PPD(i-1) is ...
            an offset to move the values closer to the real value
        AzB1(PulseI(i-1):PulseI(i)-1)=Sample_AzB1;
    end
end
% Additional Azimuth points before the first pulse
Before_1st_pulse=length(AzB1(1:PulseI(1)));
if Before_1st_pulse > PulseID(1)
    PPD=(360/Before_1st_pulse);
    Sample_AzB1=PPD*(0:Before_1st_pulse-1); % Sample for Azimuth ...
        Blade 1 initial
else
    PPD=(360/PulseID(1));
    Sample_AzB1=PPD*(0:PulseID(1)-1); % Sample for Azimuth Blade 1 ...
        final
end
AzB1(1:PulseI(1)-1)=Sample_AzB1(end-Before_1st_pulse+2:end); % Adding ... values to Azimuth Blade before first pulse

% Additional Azimuth points after the last pulse
Last=length(AzB1(PulseI(end):end));
if Last > PulseID(end)
    PPD=(360/Last);
    Sample_AzB1=PPD*(0:Last); % Sample for Azimuth Blade 1 final
else
    PPD=(360/PulseID(end));
    Sample_AzB1=PPD*(0:PulseID(end)-1); % Sample for Azimuth Blade ...
        1 final
end
AzB1(PulseI(end):end)=Sample_AzB1f(2:Last+1); % Adding values to ... 
Azimuth Blade after last pulse

% Correction of Azimuth values, from 0 to 223.9 because the 1p signal
% (is measured always at 223.9 deg)
Ilower = find(AzB1 < 136.1);
Iupper = find(AzB1 > 136.1);
AzB1(Ilower)=AzB1(Ilower)+223.9;
AzB1(Iupper)=AzB1(Iupper)-136.1;

% Azimuth for Blade 2 and Blade 3 for the 3 cases
Index = find(AzB1 > 120 & AzB1 < 240);
AzB2(Index) = AzB1(Index) - 120;
AzB3(Index) = AzB1(Index) + 120;
clear Index

Index = find(AzB1 > 240 & AzB1 < 360);
AzB2(Index) = AzB1(Index) - 120;
AzB3(Index) = AzB1(Index) - 240;
clear Index

Index = find(AzB1 > 0 & AzB1 < 120);
AzB2(Index) = AzB1(Index) + 240;
AzB3(Index) = AzB1(Index) + 120;
clear Index

% Determine sample indexes for the different blades and drives
Bla_AzIdx{1,1} = find(AzB1 < 0 + Delta | AzB1 > 360 - Delta);
Bla_AzIdx{2,1} = find(AzB2 < 0 + Delta | AzB2 > 360 - Delta);
Bla_AzIdx{3,1} = find(AzB3 < 0 + Delta | AzB3 > 360 - Delta);
for k = 2:length(AzStations)
    Bla_AzIdx{1,k} = find(AzB1 < AzStations(k) + Delta & AzB1 > ... 
        AzStations(k) - Delta);
    Bla_AzIdx{2,k} = find(AzB2 < AzStations(k) + Delta & AzB2 > ... 
        AzStations(k) - Delta);
    Bla_AzIdx{3,k} = find(AzB3 < AzStations(k) + Delta & AzB3 > ... 
        AzStations(k) - Delta);
end
else
    Rpm = N_Rotor;
    Frot = 7;
    AzStations = 0;
    Bla_AzIdx_temp = 1:1:length(PressureData);
    Bla_AzIdx{1,1} = Bla_AzIdx_temp;
    Bla_AzIdx{2,1} = Bla_AzIdx_temp;
    Bla_AzIdx{3,1} = Bla_AzIdx_temp;
end

% Pressure extraction at the Azimuth locations (COMPUTATIONS)
% *************************************************************************
% Extraction of the pressure values in the correct azimuth locations
% computed according the azimuth locations and the ∆.
% *************************************************************************
% Initializing variables
Pres25_1 = zeros(length(AzStations), length(Mic(PresChan25_1, 12)));
Pres35_1 = zeros(length(AzStations), length(Mic(PresChan35_1, 12)));
Pres60_1 = zeros(length(AzStations), length(Mic(PresChan60_1, 12)));
Pres60_2 = zeros(length(AzStations), length(Mic(PresChan60_2, 12)));
Pres60_3 = zeros(length(AzStations), length(Mic(PresChan60_3, 12)));
Pres82_2 = zeros(length(AzStations), length(Mic(PresChan82_2, 12)));
Pres82_3 = zeros(length(AzStations), length(Mic(PresChan82_3, 12)));

% Inizialiting varibales
Std_Pres25_1=zeros(length(AzStations),length(Mic(PresChan25_1,12)));  
Std_Pres35_1=zeros(length(AzStations),length(Mic(PresChan35_1,12)));  
Std_Pres60_1=zeros(length(AzStations),length(Mic(PresChan60_1,12)));  
Std_Pres60_2=zeros(length(AzStations),length(Mic(PresChan60_2,12)));  
Std_Pres60_3=zeros(length(AzStations),length(Mic(PresChan60_3,12)));  
Std_Pres82_2=zeros(length(AzStations),length(Mic(PresChan82_2,12)));  
Std_Pres82_3=zeros(length(AzStations),length(Mic(PresChan82_3,12)));  
Std_Pres92_3=zeros(length(AzStations),length(Mic(PresChan92_3,12)));  

% Averaging values
for k=1:length(AzStations)
    Pres25_1(k,:)=mean(PressureData(Bla_Az_Idx{1,k},Mic(PresChan25_1,12)));  
    Pres35_1(k,:)=mean(PressureData(Bla_Az_Idx{1,k},Mic(PresChan35_1,12)));  
    Pres60_1(k,:)=mean(PressureData(Bla_Az_Idx{1,k},Mic(PresChan60_1,12)));  
    Pres60_2(k,:)=mean(PressureData(Bla_Az_Idx{2,k},Mic(PresChan60_2,12)));  
    Pres60_3(k,:)=mean(PressureData(Bla_Az_Idx{3,k},Mic(PresChan60_3,12)));  
    Pres82_2(k,:)=mean(PressureData(Bla_Az_Idx{2,k},Mic(PresChan82_2,12)));  
    Pres82_3(k,:)=mean(PressureData(Bla_Az_Idx{3,k},Mic(PresChan82_3,12)));  
    Pres92_3(k,:)=mean(PressureData(Bla_Az_Idx{3,k},Mic(PresChan92_3,12)));  
end

% Sigma of the Average with standar deviation
for k=1:length(AzStations)
    Std_Pres25_1(k,:)=std(PressureData(Bla_Az_Idx{1,k},Mic(PresChan25_1,12)));  
    Std_Pres35_1(k,:)=std(PressureData(Bla_Az_Idx{1,k},Mic(PresChan35_1,12)));  
    Std_Pres60_1(k,:)=std(PressureData(Bla_Az_Idx{1,k},Mic(PresChan60_1,12)));  
    Std_Pres60_2(k,:)=std(PressureData(Bla_Az_Idx{2,k},Mic(PresChan60_2,12)));  
    Std_Pres60_3(k,:)=std(PressureData(Bla_Az_Idx{3,k},Mic(PresChan60_3,12)));  
    Std_Pres82_2(k,:)=std(PressureData(Bla_Az_Idx{2,k},Mic(PresChan82_2,12)));  
    Std_Pres82_3(k,:)=std(PressureData(Bla_Az_Idx{3,k},Mic(PresChan82_3,12)));  
    Std_Pres92_3(k,:)=std(PressureData(Bla_Az_Idx{3,k},Mic(PresChan92_3,12)));  
end

% Cp's values (COMPUTATIONS)
% *************************************************************************
% Computation of Cps values
% *************************************************************************
% Inizialiting varibales for Cp's
Cp25 = zeros(length(AzStations),length(PresChan25_1));  
Cp35 = zeros(length(AzStations),length(PresChan35_1));  
Cp60_1 = zeros(length(AzStations),length(PresChan60_1));  
Cp60_2 = zeros(length(AzStations),length(PresChan60_2));  
Cp60_3 = zeros(length(AzStations),length(PresChan60_3));  
Cp82_2 = zeros(length(AzStations),length(PresChan82_2));  
Cp82_3 = zeros(length(AzStations),length(PresChan82_3));  
Cp92 = zeros(length(AzStations),length(PresChan92_3));  

% Inizialiting varibales for Errorvalues for Cp's (sigma or std)
err_Cp25 = zeros(length(AzStations),length(PresChan25_1));
err_Cp35 = zeros(length(AzStations),length(PresChan35_1));
err_Cp60_1 = zeros(length(AzStations),length(PresChan60_1));
err_Cp60_2 = zeros(length(AzStations),length(PresChan60_2));
err_Cp60_3 = zeros(length(AzStations),length(PresChan60_3));
err_Cp82_2 = zeros(length(AzStations),length(PresChan82_2));
err_Cp82_3 = zeros(length(AzStations),length(PresChan82_3));
err_Cp92 = zeros(length(AzStations),length(PresChan92_3));

% Computation of Cps
if qi25 ~= 0
    for k=1:length(AzStations)
        % Cps VER 1
        Cp25(k,:) = (Pres25_1(k,:) / (Qinf+qi25));
        Cp35(k,:) = (Pres35_1(k,:) / (Qinf+qi35));
        Cp60_1(k,:) = (Pres60_1(k,:) / (Qinf+qi60));
        Cp60_2(k,:) = (Pres60_2(k,:) / (Qinf+qi60));
        Cp60_3(k,:) = (Pres60_3(k,:) / (Qinf+qi60));
        Cp82_2(k,:) = (Pres82_2(k,:) / (Qinf+qi82));
        Cp82_3(k,:) = (Pres82_3(k,:) / (Qinf+qi82));
        Cp92(k,:) = (Pres92_3(k,:) / (Qinf+qi92));
        % Errors Cp
        err_Cp25(k,:) = (Std_Pres25_1(k,:)) / (Qinf+qi25);
        err_Cp35(k,:) = (Std_Pres35_1(k,:)) / (Qinf+qi35);
        err_Cp60_1(k,:) = (Std_Pres60_1(k,:)) / (Qinf+qi60);
        err_Cp60_2(k,:) = (Std_Pres60_2(k,:)) / (Qinf+qi60);
        err_Cp60_3(k,:) = (Std_Pres60_3(k,:)) / (Qinf+qi60);
        err_Cp82_2(k,:) = (Std_Pres82_2(k,:)) / (Qinf+qi82);
        err_Cp82_3(k,:) = (Std_Pres82_3(k,:)) / (Qinf+qi82);
        err_Cp92(k,:) = (Std_Pres92_3(k,:)) / (Qinf+qi92);
    end
end

% Presentation of results (SAVING RESULTS)
% *************************************************************************
% Presentation of results
% *************************************************************************
% Determine x-axis values
x25 = Mic(PresChan25_1,7);
x35 = Mic(PresChan35_1,7);
x60_1 = Mic(PresChan60_1,7);
x60_2 = Mic(PresChan60_2,7);
x60_3 = Mic(PresChan60_3,7);
x82_2 = Mic(PresChan82_2,7);
x82_3 = Mic(PresChan82_3,7);
x92 = Mic(PresChan92_3,7);

% Which channels have pressure or suctions values
% This Following notation is used
% (:,3) - Type of sensor 3 = Pressure
% (:,5) - Span position 25, 35, 60, 82 or 92
% (:,9) = 1, 0 pressure side -1 suction side
% (:,13) = Work or not work sensor 1 = working, 0 = Not working

% Determination of Cn and Ct (SAVING RESULTS)
% Determine % X-positions pressure side and suction side
% Here the function sortcoord is used to order the positions x on the blade
% to put in the order to print, starting from 0 to 100% for the pressure
% side and then going back from 100% to 0 from the suction side.

x25bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==25 & Mic(:,9)\geq 0 & Mic(:,13)==1,7));
x25on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==25 & Mic(:,9)<0 & Mic(:,13)==1,7));
x25i = sortcoord(x25,x25bo,x25on); % Positions in order
y25 = Mic(PresChan25,8);

if bCn==1
    Cn25=zeros(1,length(AzStations));
    Ct25=zeros(1,length(AzStations));
    for k=1:length(AzStations)
        Cn25(k) = integrateCp(Cp25(k,x25i),x25(x25i),length(x25bo),1,100,0,0);
        Ct25(k) = integrateCp(Cp25(k,x25i),y25(x25i),length(x25bo),1,0,0,0);
    end
end
clear x25bo x25on;

x35bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==35 & Mic(:,9)\geq 0 & Mic(:,13)==1,7));
x35on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==35 & Mic(:,9)<0 & Mic(:,13)==1,7));
x35i = sortcoord(x35,x35bo,x35on);
y35 = Mic(PresChan35,8);

if bCn==1
    Cn35=zeros(1,length(AzStations));
    Ct35=zeros(1,length(AzStations));
    for k=1:length(AzStations)
        Cn35(k) = integrateCp(Cp35(k,x35i),x35(x35i),length(x35bo),1,100,0,0);
        Ct35(k) = integrateCp(Cp35(k,x35i),y35(x35i),length(x35bo),1,0,0,0);
    end
end
clear x35bo x35on;

% x60bo = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==1 & ... Mic(:,9)>0 & Mic(:,13)==1),7));
% x60on = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==1 & ... Mic(:,9)<0 & Mic(:,13)==1),7));
% x60,1i = sortcoord(x60,1,x60bo,x60on);
% clear x60bo x60on;

x60bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==1 & ... Mic(:,9)>0 & Mic(:,13)==1),7));
x60on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==1 & ... Mic(:,9)<0 & Mic(:,13)==1),7));
x60,1i = sortcoord(x60,1,x60bo,x60on);
clear x60bo x60on;

x60bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==2 & Mic(:,9)>0 & ... Mic(:,13)==1),7));
x60on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==2 & Mic(:,9)<0 & ... Mic(:,13)==1),7));
x60,2i = sortcoord(x60,2,x60bo,x60on);
y60 = Mic(PresChan60,2,8);
if bCn==1
    Cn60=zeros(1,length(AzStations));
    Ct60=zeros(1,length(AzStations));
    for k=1:length(AzStations)
        Cn60(k) = ... integrateCp(Cp60.2(k,x60.2i),x60.2(x60.2i),length(x60bo),1,100,0,0);
        Ct60(k) = ... integrateCp(Cp60.2(k,x60.2i),y60(x60.2i),length(x60bo),1,0,0,0);
    end
    clear x60bo x60on;
end

if bCn==1
    Cn82=zeros(1,length(AzStations));
    Ct82=zeros(1,length(AzStations));
    for k=1:length(AzStations)
        Cn82(k) = ... integrateCp(Cp82.3(k,x82.3i),x82.3(x82.3i),length(x82bo),1,100,0,0);
        Ct82(k) = ... integrateCp(Cp82.3(k,x82.3i),y82(x82.3i),length(x82bo),1,0,0,0);
    end
    clear x82bo x82on;
end

if bCn==1
    Cn92=zeros(1,length(AzStations));
    Ct92=zeros(1,length(AzStations));
    for k=1:length(AzStations)
        Cn92(k) = ... integrateCp(Cp92.3(k,x92.3i),x92.3(x92.3i),length(x92bo),1,100,0,0);
        Ct92(k) = ... integrateCp(Cp92.3(k,x92.3i),y92(x92.3i),length(x92bo),1,0,0,0);
    end
    clear x92bo x92on;
end

% x60bo = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==3 & ... Mic(:,9)>=0 & Mic(:,13)==1),7));
% x60on = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==3 & ... Mic(:,9)<0 & Mic(:,13)==1),7));
% x60.3i = sortcoord(x60.3,x60bo,x60on);
% clear x60bo x60on;
% x82bo = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,6)==2 & ... Mic(:,9)>=0 & Mic(:,13)==1),7));
% x82on = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,6)==2 & ... Mic(:,9)<0 & Mic(:,13)==1),7));
% x82.3i = sortcoord(x82.3,x82bo,x82on);
% clear x82bo x82on;
% x92bo = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==3 & ... Mic(:,9)>=0 & Mic(:,13)==1),7));
% x92on = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==3 & ... Mic(:,9)<0 & Mic(:,13)==1),7));
% x92i = sortcoord(x92,x92bo,x92on);
% y92 = Mic(PresChan92.3,8);
if bCn == 1
    Cn92=zeros(1,length(AzStations));
    Ct92=zeros(1,length(AzStations));
    for k=1:length(AzStations)
        Cn92(k) = integrateCp(Cp92(k,x92i),x92(x92i),length(x92bo),1,100,0,0);
        Ct92(k) = integrateCp(Cp92(k,x92i),y92(x92i),length(x92bo),1,0,0,0);
    end
end
clear x92bo x92on;

%% Save results (SAVING RESULTS)
% ************************************************************************
% Time, number of samples, frecuency of sampling etc
% ************************************************************************
if bAscii == 1
    mkdir(fullfile(ResultsPathAscii,FileName));
    % 1rst cross section ...
    FileNamet = strcat(FileName,'Span25.txt');
    filename = fullfile(ResultsPathAscii,FileName,FileNamet);
    fid=fopen(filename,'w');
    fprintf(fid,'# File : %s
',FileNamet);
    fprintf(fid,'# RPM : %3.1fn',Rpm);
    fprintf(fid,'# Freqrot : %3.2fn',Frot);
    fprintf(fid,'# Omega : %3.2fn',omega);
    fprintf(fid,'# Yaw : %6.2fn',Beta);
    fprintf(fid,'# Pitch : %6.2fn',Pitch);
    fprintf(fid,'# Avg : %3.0fn',NoPuls-2);
    fprintf(fid,'# Vtun : %5.1fn',Vinf);
    fprintf(fid,'# Ttun : %6.2fn',Tinf);
    fprintf(fid,'# rhotun : %7.3fn',rhot);
    %fprintf(fid,'# Pinf : %7.3fn',Patm);
    fprintf(fid,'# Tblade : %6.2fn',T);
    clear Results;
    [r,k]=size(Cp25);
    % Cp values
    Results(1:k,2:r+1)=Cp25(:,x25i)';
    % Error Values
    Results(1:k,r+2:2*r+1)=err_Cp25(:,x25i)';
    % x-Values
    Results(:,1)=x25(x25i);
    if ReOrdering == 1
        Aux1 = mat2cell(Results, [15, 12], 2*r+1);
        Aux2{1,1}=Aux1{2,1};  % pressure side
        Aux2{2,1}=Aux1{1,1};  % suction side
        Results = cell2mat(Aux2);
    end
    fprintf(fid,'
');
    fprintf(fid,'# x = 25
');
    fprintf(fid,'# x = 25
');
if qi25 == 0
    fprintf(fid,'# x, Pa[station]\r\n');
else
    fprintf(fid,'# x, Cp[station] ... Error[station]\r\n');
end
fprintf(fid,'# ');
fprintf(fid,' %8.3f',AzStations);
fprintf(fid,' %8.3f',AzStations);
[p,-]=size(Results);
for i=1:p
    fprintf(fid, '\r\n');
    fprintf(fid, ' %8.3f ',Results(i,:));
    fprintf(fid, '\r\n');
end
if bCn==1
    fprintf(fid,'# Cn ');
    fprintf(fid,' %8.3f',Cn25);
    fprintf(fid, '\r\n');
    fprintf(fid,'# Ct ');
    fprintf(fid,' %8.5f',Ct25);
    fprintf(fid, '\r\n');
end
fclose(fid);
clear fid

%% 2nd cross section ...
***********************************************************************
FileNamet = strcat(FileName,'_Span35.txt');
file_name = fullfile(ResultsPathAscii,FileName,FileNamet);

fid=fopen(file_name,'w');
fprintf(fid,'# File : %s\r\n',FileNamet);
fprintf(fid,'# RPM : %3.1f\r\n',Rpm);
fprintf(fid,'# Freqrot : %3.2f\r\n',Frot);
fprintf(fid,'# Omega : %3.2f\r\n',omega);
fprintf(fid,'# Yaw : %6.2f\r\n',Beta);
fprintf(fid,'# Pitch : %6.2f\r\n',Pitch);
fprintf(fid,'# Avg : %3.0f\r\n',NoPuls-2);
fprintf(fid,'# Vtun : %5.1f\r\n',V_inf);
fprintf(fid,'# Ttun : %6.2f\r\n',Tinf);
fprintf(fid,'# rhotun : %7.3f\r\n',rhot);
%fprintf(fid,'# Pinf : %7.3f\r\n',Patm);
fprintf(fid,'# Tblade : %6.2f\r\n',T);

clear Results;
[r, k]=size(Cp35);
%Cp Values
Results(1:k,2:r+1)=Cp35(:,x35i)';

%C Error Values
Results(1:k,r+2:2*r+1)=err_Cp35(:,x35i)';

%x-Values
Results(:,1)=x35(x35i);
if ReOrdering == 1
    Aux1 = mat2cell(Results, [14, 13], 2*r+1);
    Aux2{1,1}=Aux1{2,1}; % pressure side
    Aux2{2,1}=Aux1{1,1}; % suction side
    Results = cell2mat(Aux2);
end
fprintf(fid,'
');
if qi25 == 0
    fprintf(fid,'# x, Pa[station]
');
else
    fprintf(fid,'# x, Cp[station] ... Error[station]
');
end
fprintf(fid,'# Cn ','
');
fprintf(fid,'# Ct ','
');
fclose(fid);
clear fid
%% 3rd cross section ...

FileNamet = strcat(FileName,'Span60.txt');
file_name = fullfile(ResultsPathAscii,FileName,FileNamet);

fid=fopen(file_name,'w');
fprintf(fid,'# File : %s
',FileName);
fprintf(fid,'# RPM : %3.1f
',Rpm);
fprintf(fid,'# Freqrot : %3.2f
',Frot);
fprintf(fid,'# Omega : %3.2f
',omega);
fprintf(fid,'# Yaw : %6.2f
',Beta);
fprintf(fid,'# Pitch : %6.2f
',Pitch);
fprintf(fid,'# Avg : %3.0f
',NoPuls-2);
fprintf(fid,'# Vtun : %5.1f
',Vinf);
fprintf(fid,'# Ttun : %6.2f
',Tinf);
fprintf(fid,'# rhotun : %7.3f
',rhot);
%fprintf(fid,'# Pinf : %7.3f
',Patm);
fprintf(fid,'# Tblade : %6.2f
',T);

clear Results;
[r, k]=size(Cp60_2);
%Cp Values
Results(1:k,2:r+1)=Cp60(:,x60:2i)';

% Error Values
Results(1:k,r+2:2*r+1)=err_Cp60(:,x60:2i)';

%x-Values
Results(:,1)=x60:2(x60:2i);
if ReOrdering == 1
    Aux1 = mat2cell(Results, [12, 12], 2*r+1); % pressure side
    Aux2{1,1}=Aux1{2,1};
    Aux2{2,1}=Aux1{1,1}; % suction side
    Results = cell2mat(Aux2);
end
fprintf(fid,['\r'n];
if qi25 == 0
    fprintf(fid,'# x, Pa[station]\r\n');
else
    fprintf(fid,'# x,' Cnstasyion] ... Error[station]\r\n');
end
fprintf(fid,'\r
');
if qi25 == 0
    fprintf(fid,'# x, Pa[station]\r\n');
else
    fprintf(fid,'# x, Cnstasyion] ... Error[station]\r\n');
end
if bCn==1
    fprintf(fid,'# Cn ');fprintf(fid,' %8.3f',Cn60);
end
if bRepro == 1
    fprintf(fid,'# Add data to other Blades
    clear Results;
    [r, k]=size(Cp60,1);
    %Cp Values
    Results(1:k,2:r+1)=Cp60_1';
    %x-Values
    Results(:,1)=x60,1;
    fprintf(fid,'\r\n');
    fprintf(fid,'# x = 60 1\r\n');
    if qi25 == 0
        fprintf(fid,'# x, Pa[station]\r\n');
    else
        fprintf(fid,'# x, Pa[station]\r\n');
    end
end
fprintf(fid,'
');
end
fprintf(fid,'
');
fprintf(fid,' %6.0f ',AzStations);
fprintf(fid,'
');
for i=1:k % Here, k is used because length is the largest size
fprintf(fid,'#
');
fprintf(fid,' %8.3f',Results(i,:));
fprintf(fid,'
');
end

clear Results;
[r, k]=size(Cp60_3);
%Cp Values
Results(1:k,2:r+1)=Cp60_3';

%x-Values
Results(:,1)=x60_3;
fprintf(fid,'
');
fprintf(fid,'# x = 60 3
');
if qi25 == 0
fprintf(fid,'# x, Pa[station]\n');
else
fprintf(fid,'# x, Cp[station]\n');
end
fprintf(fid,'#
');
fprintf(fid,' %8.3f',Results(i,:));
fprintf(fid,'
');
end
fclose(fid);
clear fid
% 4th cross section ...
*********************************************************
FileNamet = strcat(FileName,'.Span82.txt');

file name = fullfile(ResultsPathAscii,FileName,FileNamet);

fid=fopen(file_name,'w');
fprintf(fid,'# File : %s\n',FileNamet);
fprintf(fid,'# RPM : %3.1f\n',Rpm);
fprintf(fid,'# Freqrot : %3.2f\n',Frot);
fprintf(fid,'# Omega : %3.2f\n',omega);
fprintf(fid,'# Yaw : %6.2f\n',Beta);
fprintf(fid,'# Pitch : %6.2f\n',Pitch);
fprintf(fid,'# Avg : %3.0f\n',NoPuls-2);
fprintf(fid,'# Vtun : %5.1f\n',Vinf);
fprintf(fid,'# Ttun : %6.2f\n',Tinf);
fprintf(fid,'# rhotun : %7.3f\n',rhot);
%fprintf(fid,'# Pinf : %7.3f\n',Patm);
fprintf(fid,'# Tblade : %6.2f\n',T);
clear Results;
[r, k]=size(Cp82_3);
% Cp Values
Results(1:k,2:r+1)=Cp82_3(:,x82_3i)';
% Error Values
Results(1:k,r+2:2*r+1)=err_Cp82_3(:,x82_3i)';
% x-Values
Results(:,1)=x82_3(x82_3i);
if ReOrdering == 1
    Aux1 = mat2cell(Results, [11, 13], 2*r+1); % Este cambi sucede ...
cuando quito los dos puntos erroneos.
    Aux1 = mat2cell(Results, [11, 11], r+1);
    Aux2{1,1}=Aux1{2,1}; % pressure side
    Aux2{2,1}=Aux1{1,1}; % suction side
    Results = cell2mat(Aux2);
end
fprintf(fid,'r
');
fprintf(fid,'# x = 82
');
if qi25 == 0
    fprintf(fid,'# x, Pa[station]\n');
else
    fprintf(fid,'# x, Cp[station] ... Error[station]\n');
end
fprintf(fid,'r
');
fprintf(fid,'# x = 82 2
');
if qi25 == 0
    fprintf(fid,'# x, Pa[station]\n');
else
    fprintf(fid,'# x, Cp[station] ... Error[station]\n');
end
fprintf(fid,'r
');
fprintf(fid,'# x, Ct');
fprintf(fid,'r
');
fprintf(fid,'# x, Cn');
fprintf(fid,'r
');
if bCn==1
    fprintf(fid,'%8.3f',Cn82);
    fprintf(fid,'r
');
    fprintf(fid,'%8.5f',Ct82);
    fprintf(fid,'r
');
    fprintf(fid,'%8.3f',Cn82);
end
if Repro == 1
    % Add data to other Blades
    clear Results;
    [r, k]=size(Cp82_2);
    % Cp Values
    Results(1:k,2:r+1)=Cp82_2';
    % x-Values
    Results(:,1)=x82_2;
    fprintf(fid,'r
');
    fprintf(fid,'# x = 82 2\n');
    if qi25 == 0
fprintf(fid,'# x, Pa[station]\n');
else
    fprintf(fid,'# x, Cp[station]\n');
end
fprintf(fid,' '); fprintf(fid,'%6.0f ',AzStations);
fprintf(fid,'\n');
for i=1:k % Here, k is used because length is the largest size
    fprintf(fid,'#');
    fprintf(fid,' %8.3f',Results(i,:));
    fprintf(fid,'\n');
end
end
fclose(fid);
clear fid

%% 5th cross section ...

****************************************************
FileNamet = strcat(FileName,'_Span92.txt');
file_name = fullfile(ResultsPathAscii,FileName,FileNamet);

fid=fopen(file_name,'w');
fprintf(fid,'# File : %s\n',FileNamet);
fprintf(fid,'# RPM : %3.1f\n',Rpm);
fprintf(fid,'# Freqrot : %3.2f\n',Frot);
fprintf(fid,'# Omega : %3.2f\n',omega);
fprintf(fid,'# Yaw : %6.2f\n',Beta);
fprintf(fid,'# Pitch : %6.2f\n',Pitch);
fprintf(fid,'# Avg : %3.0f\n',NoPuls-2);
fprintf(fid,'# Vtun : %5.1f\n',Vinf);
fprintf(fid,'# Ttun : %6.2f\n',Tinf);
fprintf(fid,'# rhotun : %7.3f\n',rhot);
fprintf(fid,'# Pinf : %7.3f\n',Patm);
fprintf(fid,'# Tblade : %6.2f\n',T);

clear Results;
[r, k]=size(Cp92);

%Cp Values
Results(1:k,2:r+1)=Cp92(:,x92i)';

% Error Values
Results(1:k,r+2:2*r+1)=errCp92(:,x92i)';

% x-Values
Results(:,1)=x92(x92i);
if ReOrdering == 1
    Aux1 = mat2cell(Results, [12, 13], 2*r+1);
    Aux2{1,1}=Aux1{2,1}; % pressure side
    Aux2{2,1}=Aux1{1,1}; % suction side
    Results = cell2mat(Aux2);
end
fprintf(fid,'\n');
fprintf(fid,'# x = 92\n');
if qi25 == 0
    fprintf(fid,'# x, Pa[station]\n');
else
fprintf(fid,'# x, Cp[station] ...
Error[station]\r\n');
end
fprintf(fid,'# ');
fprintf(fid,' $8.3f$',AzStations);
fprintf(fid,' $8.3f$',AzStations);
fprintf(fid,'\r\n');
[p,~]=size(Results);
for i=1:p
    fprintf(fid, '$8.3f$',Results{i,:});
    fprintf(fid,'\r\n');
end
if bCn==1
    fprintf(fid,'# Cn ');
    fprintf(fid,' $8.3f$',Cn92);
    fprintf(fid,'\r\n');
    fprintf(fid,'# Ct ');
    fprintf(fid,' $8.5f$',Ct92);
    fprintf(fid,'\r\n');
end
fclose(fid);
clear fid;
end
A.5 Time series values with columns arrangement

```matlab
function Status = aux_time_series_columns(DataFile,bCn,Repro,ReOrdering)

% *************************************************************************
% NewMEXICO
% aux_time_series_columns.m
% Routine to extract and process the information stored in the hdf5
% file, corresponding to the calibrated pressure. This routine is divided
% on three big sections, the setup, the computation and the saving.
% At the end of this routine txt files are saved with the information of
% the pressure coefficients, ordered in columns respect the time series
% values.
% It is possible to activate or deactivate the computation of some values
% like Cn's or Ct's, the stress values or just the canonical values, or
% save the txt file, the default option is all turn on, this means:
% bCn = 1
% Repro = 1
% Stres = 1
% bAscii = 1
% The routine save one file for each span position, with the name of the
% file R###P###D###.Span.##
% Univeristy of Twente
% February / December 2017
% *************************************************************************
% Programmer: E Prieto

%% Starting settings (SETUP)

% Clear all old data
clear Cp*
clear Pres*
clear Stress*
clear qi*
clear Blade*
clear Cn*
clear Ct*
clear Puls*
clear jit*
clear FileName1*
clear FileName2*
clear RFiles

% Initial variables
Status = 1; % Assume success on process.
PulseOk = 1; % Assume the lp signal pulse is ok.
bAscii = 1; % 1 = Results are saved in ascii file
```

126
% Select locations for require directories (SETUP)
% *************************************************************************
% Directory settings
% *************************************************************************
% ResultsPathAscii = Directory to store the time series results
ResultsPathAscii = ... /
/Users/Xientifiko/Dropbox/Thesis/Results/NewMEXICO/ascii/Time_series_columns;

% TunnelInfoPath = Directory to extract the tunnel information (.dat file)
TunnelInfoPath = ... /
/Users/Xientifiko/Desktop/Thesis/Beehub/home/MexNet/NewMexico/DNW/POL/Steady_Data;

% Wind turbine settings (SETUP)
% *************************************************************************
% Wind Turbine Settings
% *************************************************************************
Fsam = 5514; % Hz
Span = 2.25; % meters

% Read the micfile sheet (excel file with information about the sensors) ...
% (SETUP)
% Excel micfile reading
% *************************************************************************
% Location on computer of the 'micfile' file information, for instance ...
':documents\micfile2.xls'
Mic=xlsread('/Users/Xientifiko/Desktop/Thesis/Beehub/home/MexNet/NewMexico/NewMexico_datapoints.xlsx','micfile');
Chn1P = Mic(:,3)==1; % 1p sensor channel from mic file.
ChnT = Mic(:,3)==5; % Temperature sensor channel from mic.

% Selection of channels that contain pressure sensors information
% This Following notation is used
% (%,3) = Type of sensor 3 = Pressure
% (%,5) = Span position 25, 35, 60, 82 or 92
% (%,13) = Status of sensor, 1 = working, 0 = Not working
% (%,6) = Number of blade 1,2 or 3

% Type sensor Section Status ok? Blade
PresChan25_1 = find(Mic(:,3)==3 & Mic(:,5)==25 & Mic(:,13)==1 ); % 27 Sensors for this location
PresChan35_1 = find(Mic(:,3)==3 & Mic(:,5)==35 & Mic(:,13)==1 ); % 27 Sensors for this location
PresChan60_1 = find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,13)==1 & ... 
Mic(:,6)==1 ); % 8 Sensors for this location
PresChan60_2 = find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,13)==1 & ... 
Mic(:,6)==2 ); % 24 Sensors for this location
PresChan60_3 = find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,13)==1 & ... 
Mic(:,6)==3 ); % 4 Sensors for this location
PresChan82_2 = find(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,13)==1 & ...
    Mic(:,6)==2 ); % 4 Sensores for this location
PresChan82_3 = find(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,13)==1 & ...
    Mic(:,6)==3 ); % 24 Sensores for this location
PresChan92_3 = find(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,13)==1 ); ...% 26 Sensores for this location

% Read wind tunnel data (SETUP)
% *************************************************************************
% Tunnel Turbine Settings
% *************************************************************************
% Name extraction.
[~,FileName,~] = fileparts(DataFile);
% using function rpdfilename obtain [r], [p] and [d] values
[~, p ,d] = RPDFilename(FileName);
% Filename wind tunnel data POLPPP
NameFile = strcat(TunnelInfoPath,'/POL',p,'.DAT');
% Verify if file has representative the TunnelInfo
if exist (NameFile,'file')==2
    fid = fopen(NameFile);
    DataInf = textscan(fid,'%f','HeaderLines',8,'Delimiter','|');
    POLtunneldata = DataInf{1,1};
    POLtunneldata = reshape(POLtunneldata,31,[]);
    POLtunneldata = POLtunneldata';
    D=str2double(d);
    Index_D = POLtunneldata(:,1)==D;
    TunnelData = POLtunneldata(Index_D,:);
% Information contained in the tunneldata file, the "commented" values
% can be activated to obtain the specific information.
% PNT = TunnelData(1); % [-] Data point number
% Time = TunnelData(2); % [hhmmssxxx] Time
Qinf = TunnelData(3); % [Pa] Freestream dynamic pressure
RHOinf = TunnelData(4); % [kg/m3] Freestream air density
Vinf = TunnelData(5); % [m/s] Freestream tunnel speed
Beta = TunnelData(6); % [?] Yaw angle
Pitch = TunnelData(7); % [?] Blade pitch angle
Tinf = TunnelData(8); % [? K] Freestream tunnel temperature
% Fx_Rot = TunnelData(9); % [N] Force in x-direc ... on (model coordinate system)
% Fy_Rot = TunnelData(10); % [N] Force in y-direc ... on (model coordinate system)
% Fz_Rot = TunnelData(11); % [N] Force in z-direc ... on (model coordinate system)
% Mx_Rot = TunnelData(12); % [Nm] Moment around ... x-axis (model coordinate system)
% My_Rot = TunnelData(13); % [Nm] Moment around ... y-axis (model coordinate system)
% Mz_Rot = TunnelData(14); % [Nm] Moment around ... z-axis (model coordinate system)
N_Rotor = TunnelData(15); % [rpm] Rota onal speed of rotor

128
142  \% Lambda = TunnelData(16); \% [-] Tip speed ratio of rotor
143  \% X_Trav = TunnelData(17); \% [m] x-coordinate of traverse (tunnel coordinate system)
144  \% Y_Trav = TunnelData(18); \% [m] y-coordinate of traverse (tunnel coordinate system)
145  \% Pinf = TunnelData(19); \% [Pa] Freestream static pressure
146  \% PTinf = TunnelData(20); \% [Pa] Freestream total pressure
147  \% dpitot = TunnelData(21); \% [Pa] Pitot tube dynamic pressure (velocity verification only)
148  \% VPitot = TunnelData(22); \% [m/s] Pitot tube velocity reading (velocity verification only)
149  \% Current = TunnelData(23); \% [-] Generator torque (blade-off only)
150  \% Cell1 = TunnelData(24); \% [Pa] Collector static pressure (Figure 17)
151  \% Cell2 = TunnelData(25); \% [Pa] Collector static pressure (Figure 17)
152  \% Start1 = TunnelData(26); \% [Pa] Collector static pressure (Figure 17)
153  \% Start2 = TunnelData(27); \% [Pa] Collector static pressure (Figure 17)
154  \% Port1 = TunnelData(28); \% [Pa] Collector static pressure (Figure 17)
155  \% Port2 = TunnelData(29); \% [Pa] Collector static pressure (Figure 17)
156  \% Floor1 = TunnelData(30); \% [Pa] Collector static pressure (Figure 17)
157  \% Floor2 = TunnelData(31); \% [Pa] Collector static pressure (Figure 17)
158  fclose(fid);
159  else
160     Status = 2; \% Fail cause no TunnelInfo
161     return
162  end
163
164  \% Read all available Pressure calibrated data (SETUP)
165  \% ************************************************************************
166  \% Pressure Calibrated Data *.hdf5 files
167  \% ************************************************************************
168  \% [-,NameFile,] = fileparts(DataFile);
169  NameFile = strcat('/',NameFile);
170  PressureData = h5read(DataFile,NameFile);
171  [NumSam,NumSensors]=size(PressureData);
172
173  \% Extrat time of the run
174  MeasTime = NumSam / Fsam;
175  Tiempos=1/Fsam;
176  Time=0:Tiempo:MeasTime;
177  Time=Time(1:end-1);
% Pulse Analisys (COMPUTATIONS)

% **********************************************************************************************************************************************
% Pulse Analysis 1p signal
% **********************************************************************************************************************************************
% Verifying if the 1p signal is correct.
if NumSensors < 161
    Status = 5;
    return
end

Pulse=PressureData(:,Mic(Chn1P,12));
Pulse = Pulse * -1;
PulseD = diff(Pulse); %Pulse signal PulseD=Pulse(n)-Pulse(n-1)
if max(PulseD)<3 % If no pulses are present
    if V_inf > 0
        Status = 4; % Either pressure zero run or wrong 1p signal
    else
        Status = 5;
        return
    end
else
    PulseOk = 0;
end

if PulseOk==1
    PulsMax=(max(PulseD))/2;
    [~,PulseI] = findpeaks(PulseD,'MinPeakHeight',PulsMax); %Pulse at ... 223.9 azimuth angle
    NoPuls=length(PulseI);
    Rpm = N_Rotor;
    Frot = Rpm*(1/60); % Hz speed estimates
    clear Pulse;
    if (exist('PulseD', 'var')>0)
        clear PulseD;
    end
else
    Rpm = N_Rotor;
    NoPuls = 2;
end

% Extract RPM and Pitch time series from the data
% **********************************************************************************************************************************************
% Time, Rpm and pitch series. and plot of this values.
% **********************************************************************************************************************************************

% RPM extraction and scaling
Rpm_signal = PressureData(:,165);
Rpm_signal=Rpm_signal*50.42;
Rpm_signal_fil=Rpml_signal;
figure(1);
plot(Time, Rpm_signal_fil)
hold on
set(gca,'Xtick',0:1:15)
grid minor
% Pitch extraction filtering and scaling
Pitch_signal = PressureData(:,168);
Pitch_signal=(Pitch_signal*19.00)-49.8;
Pitch_signal_filtr=sgolayfilt(Pitch_signal,1,99);
figure(2);
hold on
grid minor

%% Processing tunnel parameters (COMPUTATIONS)
******************************************************************************
Tunnel parameters, for the comput of the qi's
******************************************************************************

% Calculate temperature
T = mean(PressureData(:,Mic(ChnT,12))); % Average Temperature data, and ...
Tempuratur Constants for the compute of the real Temperature, this ...
values could be referring to stagnation T.
cT1 = 1 / 83.4388;
cT0 = -259.74;
T = cT1*T + cT0;

omega = Rpm/60 * 2 * pi; % Omega Angular velocity (rad/s)
 rho = RH0inf; % Air density rho

%% Cps VER 1
% Qi per station (Pascales)
omega_t = Rpm_signal_filtr/60 * 2 * pi; % Omega Angular velocity (rad/s)
qi25 = 0.5 * rho * ((omega_t * 0.25 * Span).^2);
qi35 = 0.5 * rho * ((omega_t * 0.35 * Span).^2);
qi60 = 0.5 * rho * ((omega_t * 0.60 * Span).^2);
qi82 = 0.5 * rho * ((omega_t * 0.82 * Span).^2);
qi92 = 0.5 * rho * ((omega_t * 0.92 * Span).^2);

% Extration of azimuth position for pressures (COMPUTATIONS)
******************************************************************************
% Azimuth Angles computation
******************************************************************************

% Determine sample indexes for the different blades
AzB1=zeros(1,NumSam);
AzB2=zeros(1,NumSam);
AzB3=zeros(1,NumSam);
PPD=zeros(1,NoPuls-1); % Pulse per Degree
if PulseOk==1
for i=2:NoPuls
PPD(i-1)=360/PulseID(i-1);
Sample_AzB1=PPD(i-1)*0:PulseID(i-1)-1; % the term + PPD(i-1) is ...
an offset to move the values closer to the real value
AzB1(PulseI(i-1):PulseI(i)-1)=Sample_AzB1;
end
% Aditional Azimuth points before the first pulse
Before_1st_pulse=length(AzB1(1:PulseI(1)));  
if Before_1st_pulse > PulseID(1)  
    PPD=(360/Before_1st_pulse);  
    Sample_AzB1i=PPD*(0:Before_1st_pulse-1);  
    % Sample for Azimuth Blade 1 initial
    else  
    PPD=(360/PulseID(1));  
    Sample_AzB1i=PPD*(0:PulseID(1)-1);  
    % Sample for Azimuth Blade 1 final
end  
AzB1(1:PulseI(1)-1)=Sample_AzB1i(end-Before_1st_pulse+2:end);  
% Adding values to Azimuth Blade 1 before first pulse

% Additional Azimuth points after the last pulse
Last=length(AzB1(PulseI(end):end));  
if Last > PulseID(end)  
    PPD=(360/Last);  
    Sample_AzB1f=PPD*(0:Last);  
    % Sample for Azimuth Blade 1 final
    else  
    PPD=(360/PulseID(end));  
    Sample_AzB1f=PPD*(0:PulseID(end)-1);  
    % Sample for Azimuth Blade 1 final
end  
AzB1(PulseI(end):end)=Sample_AzB1f(2:Last+1);  
% Adding values to Azimuth Blade 1 after last pulse

% Correction of Azimuth values, from 0 to 223.9 because the lp signal  
% (is measured always at 223.9 deg)
Ilower = find(AzB1 < 136.1);  
Iupper = find(AzB1 > 136.1);  
AzB1(Ilower)=AzB1(Ilower)+223.9;  
AzB1(Iupper)=AzB1(Iupper)-136.1;

% Azimuth for Blade 2 and Blade 3 for the 3 cases
Index = find(AzB1 > 120 & AzB1 < 240);  
AzB2(Index)=AzB1(Index)-120;  
AzB3(Index)=AzB1(Index)+120;  
clear Index
Index = find(AzB1 > 240 & AzB1 < 360);  
AzB2(Index)=AzB1(Index)-120;  
AzB3(Index)=AzB1(Index)-240;  
clear Index
Index = find(AzB1 > 0 & AzB1 < 120);  
AzB2(Index)=AzB1(Index)+240;  
AzB3(Index)=AzB1(Index)+120;  
clear Index
else  
    if V_inf > 0  
        NumSam = NumSam-1;  
    end  
    NumSam = NumSam+1;  
else
337    NumSam = 1;
338  end
339  Time = 0;
340  AzB1 = 1;
341  AzB2 = AzB1;
342  AzB3 = AzB1;
343  Frot = 7;
344  end
345
346  %% Pressure extraction at the time locations (all) (COMPUTATIONS)
347  % *************************************************************************
348  % Extraction of the pressure values in the correct azimuth locations
349  % computed according the azimuth locations and the ∆.
350  % *************************************************************************
351 352  %% Time location extractions
353  if NumSam > 2
354    Pres25_1(:, :) = (PressureData(:, Mic(PresChan25_1, 12)));
355    Pres35_1(:, :) = (PressureData(:, Mic(PresChan35_1, 12)));
356    Pres60_1(:, :) = (PressureData(:, Mic(PresChan60_1, 12)));
357    Pres60_2(:, :) = (PressureData(:, Mic(PresChan60_2, 12)));
358    Pres60_3(:, :) = (PressureData(:, Mic(PresChan60_3, 12)));
359    Pres82_2(:, :) = (PressureData(:, Mic(PresChan82_2, 12)));
360    Pres82_3(:, :) = (PressureData(:, Mic(PresChan82_3, 12)));
361    Pres92_3(:, :) = (PressureData(:, Mic(PresChan92_3, 12)));
362  else
363    Pres25_1 = mean(PressureData(:, Mic(PresChan25_1, 12)));
364    Pres35_1 = mean(PressureData(:, Mic(PresChan35_1, 12)));
365    Pres60_1 = mean(PressureData(:, Mic(PresChan60_1, 12)));
366    Pres60_2 = mean(PressureData(:, Mic(PresChan60_2, 12)));
367    Pres60_3 = mean(PressureData(:, Mic(PresChan60_3, 12)));
368    Pres82_2 = mean(PressureData(:, Mic(PresChan82_2, 12)));
369    Pres82_3 = mean(PressureData(:, Mic(PresChan82_3, 12)));
370    Pres92_3 = mean(PressureData(:, Mic(PresChan92_3, 12)));
371  end
372
373 374  %% Cps values (COMPUTATIONS)
375  % *************************************************************************
376  % Computation of Cps values
377  % *************************************************************************
378 379  % Initialize variables for Cps
380  Cp25 = zeros(NumSam, length(PresChan25_1));
381  Cp35 = zeros(NumSam, length(PresChan35_1));
382  Cp60_1 = zeros(NumSam, length(PresChan60_1));
383  Cp60_2 = zeros(NumSam, length(PresChan60_2));
384  Cp60_3 = zeros(NumSam, length(PresChan60_3));
385  Cp82_2 = zeros(NumSam, length(PresChan82_2));
386  Cp82_3 = zeros(NumSam, length(PresChan82_3));
387  Cp92 = zeros(NumSam, length(PresChan92_3));
388  % Computation of Cps
% if qi25(1,1) ≠ 0

% Version 1
for k=1:length(PresChan25_1)
    Cp25(:,k) = ((Pres25_1(:,k) ./ (Qinf+qi25(:,1)));
end
for k=1:length(PresChan35_1)
    Cp35(:,k) = ((Pres35_1(:,k) ./ (Qinf+qi35(:,1)));
end
for k=1:length(PresChan60_1)
    Cp60_1(:,k) = ((Pres60_1(:,k) ./ (Qinf+qi60(:,1)));
end
for k=1:length(PresChan60_2)
    Cp60_2(:,k) = ((Pres60_2(:,k) ./ (Qinf+qi60(:,1)));
end
for k=1:length(PresChan60_3)
    Cp60_3(:,k) = ((Pres60_3(:,k) ./ (Qinf+qi60(:,1)));
end
for k=1:length(PresChan82_2)
    Cp82_2(:,k) = ((Pres82_2(:,k) ./ (Qinf+qi82(:,1)));
end
for k=1:length(PresChan82_3)
    Cp82_3(:,k) = ((Pres82_3(:,k) ./ (Qinf+qi82(:,1)));
end
for k=1:length(PresChan92_3)
    Cp92(:,k) = ((Pres92_3(:,k) ./ (Qinf+qi92(:,1)));
end

%% Presentation of results (SAVING RESULTS)
% ************************************************************************
% Presentation of results
% ************************************************************************
% Determine% x-axis values
x25 = Mic(PresChan25_1,7);
x35 = Mic(PresChan35_1,7);
x60_1 = Mic(PresChan60_1,7);
x60_2 = Mic(PresChan60_2,7);
x60_3 = Mic(PresChan60_3,7);
x82_2 = Mic(PresChan82_2,7);
x82_3 = Mic(PresChan82_3,7);
x92 = Mic(PresChan92_3,7);

% Which channels have presSure or suctions values
% This Following notation is used
% (:,3) = Type of sensor 3 = Pressure
% (:,5) = Span position 25, 35, 60, 82 or 92
% (:,9) = 1, 0 presSure side -1 suction side
% (:,13) = Work or not work sensor 1 = working, 0 = Not working

%% Determination of Cn and Ct (SAVING RESULTS)
% Determine % X-positions pressure side and suctions side
% Here the uction sortcoord is used to order the positions x on the blade
% to put in the order to print, starting from 0 to 100% for the pressure
% side and then going back from 100% to 0 from the suctions side.
x25bo = sort(Mic(:,3)==3 & Mic(:,5)==25 & Mic(:,9) >= 0 & Mic(:,13)==1,7));
x25on = sort(Mic(:,3)==3 & Mic(:,5)==25 & Mic(:,9)<0 & Mic(:,13)==1,7));
x25i = sortcoord(x25,x25bo,x25on); % Positions in order
y25 = Mic(PresChan25,1,8);

if bCn==1
    Cn25=zeros(1,NumSam);
    Ct25=zeros(1,NumSam);
    for k=1:NumSam
        Cn25(k) = integrateCp(Cp25(k,x25i),x25(x25i),length(x25bo),1,100,0,0);
        Ct25(k) = integrateCp(Cp25(k,x25i),y25(x25i),length(x25bo),1,0,0,0);
    end
    clear x25bo x25on;
end

x35bo = sort(Mic(:,3)==3 & Mic(:,5)==35 & Mic(:,9) >= 0 & Mic(:,13)==1,7));
x35on = sort(Mic(:,3)==3 & Mic(:,5)==35 & Mic(:,9)<0 & Mic(:,13)==1,7));
x35i = sortcoord(x35,x35bo,x35on);
y35 = Mic(PresChan35,1,8);

if bCn==1
    Cn35=zeros(1,NumSam);
    Ct35=zeros(1,NumSam);
    for k=1:NumSam
        Cn35(k) = integrateCp(Cp35(k,x35i),x35(x35i),length(x35bo),1,100,0,0);
        Ct35(k) = integrateCp(Cp35(k,x35i),y35(x35i),length(x35bo),1,0,0,0);
    end
    clear x35bo x35on;
end

x60bo = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==1 & ...
    Mic(:,9)>=0 & Mic(:,13)==1),7));
x60on = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==1 & ...
    Mic(:,9)<0 & Mic(:,13)==1),7));
x60i = sortcoord(x601,x60bo,x60on);
clear x60bo x60on;

if bCn==1
    Cn60=zeros(1,NumSam);
    Ct60=zeros(1,NumSam);
    for k=1:NumSam
        Cn60(k) = integrateCp(Cp60(k,x602),x602(x602),length(x60bo),1,100,0,0);
\[
Ct60(k) = \ldots
\]
\[
\text{integrateCp}(Cp60,2(k,x60,2i),y60(x60,2i),\text{length}(x60bo),1,0,0,0);
\]
end
end
clear x60bo x60on;

\% x60bo = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==3 & ...
Mic(:,9)\geq0 & Mic(:,13)==1),7));
\% x60on = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==3 & ...
Mic(:,9)<0 & Mic(:,13)==1),7));
\% x60,3i = sortcoord(x60,3,x60bo,x60on);
\% clear x60bo x60on;

\% x82bo = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,6)==2 & ...
Mic(:,9)\geq0 & Mic(:,13)==1),7));
\% x82on = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,6)==2 & ...
Mic(:,9)<0 & Mic(:,13)==1),7));
\% x82,2i = sortcoord(x82,3,x82bo,x82on);
\% clear x82bo x82on;

\% x82bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,6)==3 & Mic(:,9)\geq0 & ...
Mic(:,13)==1,7));
\% x82on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,6)==3 & Mic(:,9)<0 & ...
Mic(:,13)==1,7));
\% x82,3i = sortcoord(x82,3,x82bo,x82on);
\% y82 = Mic(PresChan82,3,8);

if bCn==1
Cn82=zeros(1,NumSam);
Ct82=zeros(1,NumSam);
for k=1:NumSam
Cn82(k) = ... 
\text{integrateCp}(Cp82,2(k,x82,3i),x82,3(x82,3i),\text{length}(x82bo),1,100,0,0);
Ct82(k) = ... 
\text{integrateCp}(Cp82,2(k,x82,3i),y82(x82,3i),\text{length}(x82bo),1,0,0,0);
end
clear x82bo x82on;

if bCn == 1
Cn92=zeros(1,NumSam);
Ct92=zeros(1,NumSam);

\% x92bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==3 & Mic(:,9)\geq0 & ...
Mic(:,13)==1,7));
\% x92on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==3 & Mic(:,9)<0 & ...
Mic(:,13)==1,7));
\% x92,2i = sortcoord(x92,x92bo,x92on);
\% y92 = Mic(PresChan92,3,8);

\% x92bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==3 & Mic(:,9)\geq0 & ...
Mic(:,13)==1,7));
\% x92on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==3 & Mic(:,9)<0 & ...
Mic(:,13)==1,7));
\% x92,3i = sortcoord(x92,3,x92bo,x92on);
\% y92 = Mic(PresChan92,3,8);

\% x92bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==2 & ...
Mic(:,9)\geq0 & Mic(:,13)==1,7));
\% x92on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==2 & ...
Mic(:,9)<0 & Mic(:,13)==1,7));
\% x92,2i = sortcoord(x92,3,x92bo,x92on);
\% clear x92bo x92on;

\% x92bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==2 & Mic(:,9)\geq0 & ...
Mic(:,13)==1,7));
\% x92on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==2 & Mic(:,9)<0 & ...
Mic(:,13)==1,7));
\% x92,3i = sortcoord(x92,3,x92bo,x92on);
\% y92 = Mic(PresChan92,3,8);

\% x92bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==2 & Mic(:,9)\geq0 & ...
Mic(:,13)==1,7));
\% x92on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==2 & Mic(:,9)<0 & ...
Mic(:,13)==1,7));
\% x92,2i = sortcoord(x92,3,x92bo,x92on);
\% clear x92bo x92on;
for k=1:NumSam
    Cn92(k) = integrateCp(Cp92(k,x92i),x92(x92i),length(x92bo),1,100,0,0);
    Ct92(k) = integrateCp(Cp92(k,x92i),y92(x92i),length(x92bo),1,0,0,0);
end
clear x92bo x92on;

%% Save results (SAVING RESULTS)
% *************************************************************************
% Time, number of samples, frequency of sampling etc
% *************************************************************************
if bAscii == 1
    mkdir(fullfile(ResultsPathAscii,FileName));
    % 1rst cross section ...
    %*************************************************************************
    FileNamet = strcat(FileName,'_Span25.txt');
    file_name = fullfile(ResultsPathAscii,FileName,FileNamet);
    fid=fopen(file_name,'w');
    fprintf(fid,'# File : %s 
',FileNamet);
    fprintf(fid,'# RPM : %3.1fr
',Rpm);
    fprintf(fid,'# Freqrot : %3.2fr
',Frot);
    fprintf(fid,'# Omega : %3.2fr
',omega);
    fprintf(fid,'# Yaw : %6.2fr
',Beta);
    fprintf(fid,'# Pitch : %6.2fr
',Pitch);
    fprintf(fid,'# Avg : %3.0fr
',NoPuls-2);
    fprintf(fid,'# Vtun : %5.1fr
',Vinf);
    fprintf(fid,'# Ttun : %6.2fr
',Tinf);
    fprintf(fid,'# rhotun : %7.3fr
',rhot);
    fprintf(fid,'# Pinf : %7.3fr
',Patm);
    fprintf(fid,'# Tblade : %6.2fr
',T);
    clear Results;
    [r,k]=size(Cp25);
    % Cp values
    Results(1:k,2:r+1)=Cp25(:,x25i)';
    % x-locations
    Results(:,1)=x25(x25i);
    if ReOrdering == 1
        Aux1 = mat2cell(Results, [15, 12], r+1);
        Aux2{1,1}=Aux1{2,1}; % pressure side
        Aux2{2,1}=Aux1{1,1}; % suction side
        Results = cell2mat(Aux2);
    end
    fprintf(fid,'\r
');
    fprintf(fid,'% x = 25\r
');
    if qi25(1,1) == 0
        fprintf(fid,'% x, Pa(station)\r
');
    else
        fprintf(fid,'% x, Cp(station)\r
');
    end
    fprintf(fid,'% Time ');%Time Azi B1 ');
    fprintf(fid,’ %8.5f ’,Time);
fprintf(fid,'\r\n');
fprintf(fid,'# Azi B1 ');#Time Azi B1 ');
fprintf(fid,' %8.2f ',AzB1);
fprintf(fid,'\r\n');
fprintf(fid,'# RPM ');%rpm_signal_filtr);
fprintf(fid,' %8.2f ',Rpm_signal_filtr);
fprintf(fid,'\r\n');
fprintf(fid,'# Pitch ');%pitch_signal_filtr);
fprintf(fid,' %10.3f',Results(i,:));
fprintf(fid,'\r\n');
if bCn==1
    fprintf(fid,' %10.3f',Cn25);
    fprintf(fid,' %10.5f',Ct25);
end
fclose(fid);
clear fid
%% 2nd cross section ...
fileNamet = strcat(FileName,'_Span35.txt');
file_name = fullfile(ResultsPathAscii,FileName,FileNamet);
fid=fopen(file_name,'w');
fprintf(fid,'# File : %s\r\n',FileNamet);
fprintf(fid,'# RPM : %3.1f\r\n',Rpm);
fprintf(fid,'# Freqrot : %3.2f\r\n',Frot);
fprintf(fid,'# Omega : %3.2f\r\n',omega);
fprintf(fid,'# Yaw : %6.2f\r\n',Beta);
fprintf(fid,'# Pitch : %6.2f\r\n',Pitch);
fprintf(fid,'# Avg : %3.0f\r\n',NoPuls-2);
fprintf(fid,'# Vtun : %5.1f\r\n',V_inf);
fprintf(fid,'# Ttun : %6.2f\r\n',T_inf);
fprintf(fid,'# rhotun : %7.3f\r\n',rhot);
%fprintf(fid,'# Pinf : %7.3f\r\n',Patm);
fprintf(fid,'# Tblade : %6.2f\r\n',T);

clear Results;
[r,k]=size(Cp35);
%Cp Values
Results(1:k,2:r+1)=Cp35(:,x35i)';
%x-Values
Results(:,1)=x35(x35i);
if ReOrdering == 1
    Aux1 = mat2cell(Results, [14, 13], r+1);
    Aux2{1,1}=Aux1[2,1]; % pressure side
Aux2{2,1}=Aux1{1,1};  % suction side

Results = cell2mat(Aux2);

fprintf(fid, '\n');
fprintf(fid, '# x = 35 \n');
if qi25(1,:) == 0
    fprintf(fid, '# x, Pa(station) \n');
else
    fprintf(fid, '# x, Cp(station) \n');
end
fprintf(fid, '# x, fpp \n');
[p,-]=size(Results);
for i=1:p
    fprintf(fid, ' %10.3f',Results(i,:));
end
if bCn==1
    fprintf(fid, '\n');
    fprintf(fid, '# Cn \n');
    fprintf(fid, '%10.3f',Cn35);
    fprintf(fid, '\n');
    fprintf(fid, '# Ct \n');
    fprintf(fid, '%10.3f',Ct35);
end
fclose(fid);
clear fid

% 3rd cross section ...

FileNamet = strcat(FileName,'Span60.txt');
file_name = fullfile(ResultsPathAscii,FileName,FileNamet);

fid=fopen(file_name,'w');
fprintf(fid, '# File : %s \n',FileNamet);
fprintf(fid, '# RPM : %3.1f \n',Rpm);
fprintf(fid, '# Freqrot : %3.2f \n',Frot);
fprintf(fid, '# Omega : %3.2f \n',omega);
fprintf(fid, '# Yaw : %6.2f \n',Beta);
fprintf(fid, '# Pitch : %6.2f \n',Pitch);
fprintf(fid, '# Avg : %3.0f \n',NoPuls-2);
fprintf(fid, '# Vtun : %5.1f \n',Vinf);
fprintf(fid, '# Ttun : %6.2f \n',Tinf);
fprintf(fid, '# rhotun : %7.3f \n',rhot);
%fprintf(fid, '# Pinf : %7.3f \n',Patm);
fprintf(fid,'# Tblade : %6.2f\n',T);
clear Results;
[r, k]=size(Cp60_2);
% Cp Values
Results(1:k,2:r+1)=Cp60_2(:,x60_2i)';
% x-locations
Results(:,1)=x60_2(x60_2i);
if ReOrdering == 1
    Aux1 = mat2cell(Results, [12, 12], r+1);
    Aux2[1,1]=Aux1[2,1]; % pressure side
    Aux2[2,1]=Aux1[1,1]; % suction side
    Results = cell2mat(Aux2);
end
fprintf(fid,'\n');
fprintf(fid,'# x = 60\n');
if qi25(1,1) == 0
    fprintf(fid,'# x, Pa[station]\n');
else
    fprintf(fid,'# x, Cp[station]\n');
end
fprintf(fid,'# Time ');%Time Azi B1 ');
fprintf(fid,' %8.5f ',Time);
fprintf(fid,'\n');
fprintf(fid,'# Azi B2 ');%Time Azi B2 ');
fprintf(fid,' %8.2f ',AzB2);
fprintf(fid,'\n');
fprintf(fid,'# RPM ');%Rpm_signal_filtri;
fprintf(fid,' %8.2f ',Rpm_signal_filtri);
fprintf(fid,'\n');
[p,] = size(Results);
for i=1:p
    fprintf(fid,' %10.3f',Results(i,:));
    fprintf(fid,'\n');
end
if bCn==1
    fprintf(fid,'\n');
    fprintf(fid,'# Cn ');%Cn_1';
    fprintf(fid,' %10.3f',Cn60);
    fprintf(fid,'\n');
    fprintf(fid,'# Ct ');%Ct_1';
    fprintf(fid,' %10.5f',Ct60);
    fprintf(fid,'\n');
end
if Repro == 1
% Add data to other Blades
clear Results;
[r, k]=size(Cp60_1);
% Cp Values
Results(1:k,2:r+1)=Cp60_1';
% x-Values
Results(:,1)=x60_1;
fprintf(fid, '\r\n');
fprintf(fid, '# x = 60 1\r\n');
if qi25 == 0
  fprintf(fid, '# x, Pa[station]\r\n');
else
  fprintf(fid, '# x, Cp[station]\r\n');
end
fprintf(fid, '# Time ');
fprintf(fid, '%8.5f ',Time);
fprintf(fid, '\r\n');
fprintf(fid, '# Azi B2 ');fprintf(fid, '%8.2f ',AzB2);
fprintf(fid, '\r\n');
for i=1:k % Here, k is used because length is the largest size
  fprintf(fid,' %8.3f',Results(i,:));
  fprintf(fid,');
  fprintf(fid, '%10.3f',Results(i,:));
  fprintf(fid, '\r\n');
end
clear Results;
[r,k]=size(Cp60_3);
% Cp Values
Results(1:k,2:r+1)=Cp60_3';
% x-Values
Results(:,1)=x60_3;
fprintf(fid,' \r\n');
fprintf(fid, '# x = 60 3\r\n');
if qi25(1,1) == 0
  fprintf(fid, '# x, Pa[station]\r\n');
else
  fprintf(fid, '# x, Cp[station]\r\n');
end
fprintf(fid, '# Time ');
fprintf(fid, '%8.5f ',Time);
fprintf(fid, '\r\n');
fprintf(fid, '# Azi B3 ');fprintf(fid, '%8.2f ',AzB3);
fprintf(fid, '\r\n');
for i=1:k % Here, k is used because length is the largest size
  fprintf(fid,' %8.3f',Results(i,:));
  fprintf(fid,');
  fprintf(fid, '%10.3f',Results(i,:));
  fprintf(fid, '\r\n');
end
fclose(fid);
clear fid

%% 4th cross section ...
*********************************************************
FieldName = strcat(FileName, '_Span82.txt');
file_name = fullfile(ResultsPathAscii,FileName,FieldName);
fid=fopen(file_name,'w');
fprintf(fid,'# File : %s
', FileNamet);
fprintf(fid,'# RPM : %3.1f
', Rpm);
fprintf(fid,'# Freqrot : %3.2f
', Frot);
fprintf(fid,'# Omega : %3.2f
', omega);
fprintf(fid,'# Yaw : %6.2f
', Beta);
fprintf(fid,'# Pitch : %6.2f
', Pitch);
fprintf(fid,'# Avg : %3.0f
', NoPuls-2);
fprintf(fid,'# Vtun : %5.1f
', Vinf);
fprintf(fid,'# Ttun : %6.2f
', Tinf);
fprintf(fid,'# rhotun : %7.3f
', rhot);
%fprintf(fid,'# Pinf : %7.3f
', Patm);
fprintf(fid,'# Tblade : %6.2f
', T);

clear Results;
[r, k]=size(Cp82);

%Cp Values
Results(1:k,2:r+1)=Cp82(:,x8i);

%x-locations
Results(:,1)=x82(x8i);
if ReOrdering == 1
    Aux1 = mat2cell(Results, [11, 13], r+1);
    Aux2{1,1}=Aux1{2,1}; % pressure side
    Aux2{2,1}=Aux1{1,1}; % suction side
    Results = cell2mat(Aux2);
end
fprintf(fid,'

# x = 82

if qi25(1,1) == 0
    fprintf(fid,'# x, Pa[station]

else
    fprintf(fid,'# x, Cp[station]

end
fprintf(fid,'# Time ');%Time Azi B1 ');
fprintf(fid,' %8.5f ',Time);
fprintf(fid,'

# Azi B3 ');
fprintf(fid,' %8.2f ',AzB3);
fprintf(fid,'

# RPM ');
fprintf(fid,' %8.2f ',Rpm_signal_filtr);
fprintf(fid,'

# Pitch ');
fprintf(fid,' %8.2f ',Pitch_signal_filtr);
fprintf(fid,'

for i=1:p
    fprintf(fid,' %10.3f',Results(i,:));
    fprintf(fid,'

end
if bCn==1
    fprintf(fid,'

    fprintf(fid,'# Cn ');
fprintf(fid,' %10.3f',Cn82);
    fprintf(fid,'

142
fprintf(fid,'# Ct ');
fprintf(fid,' %10.5f',Ct82);
fprintf(fid,' \\n');
end

if Repro == 1
    % Add data to other Blades
    clear Results;
    [r, k]=size(Cp82,2);
    %Cp Values
    Results(1:k,2:r+1)=Cp82';
    % x-Values
    Results(:,1)=x82;
    fprintf(fid,' \\
');
    fprintf(fid,'# x = 82 2 \\
');
    if qi25(1,1) == 0
        fprintf(fid,'# x, Pa[station]\n');
    else
        fprintf(fid,'# x, Cp[station]\n');
    end
    fprintf(fid,'# Time ');
    fprintf(fid,' %8.5f ',Time);
    fprintf(fid,' \\
');
    fprintf(fid,'# Azi B2 ');%
    fprintf(fid,' %8.2f ',AzB2);
    fprintf(fid,' \\
');
    for i=1:k % Here, k is used because length is the largest size
        fprintf(fid,'#');
        fprintf(fid,' %10.3f',Results(i,:));
        fprintf(fid,' \\
');
    end
end
fclose(fid);
%
% 5th cross section ...
FILENAME = strcat(FileName,'_Span92.txt');
file_name = fullfile(ResultsPathAscii,FileName,FILENAME);
if fid=fopen(file_name,'w');
    fprintf(fid,'# File : %s\n',FILENAME);
    fprintf(fid,'# RPM : %3.1f \n',Rpm);
    fprintf(fid,'# Freqrot : %3.2f \n',Frot);
    fprintf(fid,'# Omega : %3.2f \n',omega);
    fprintf(fid,'# Yaw : %6.2f \n',Beta);
    fprintf(fid,'# Pitch : %6.2f \n',Pitch);
    fprintf(fid,'# Avg : %3.0f \n',NoPuls-2);
    fprintf(fid,'# Vtun : %5.1f \n',V_inf);
    fprintf(fid,'# Ttun : %6.2f \n',T_inf);
    fprintf(fid,'# rhotun : %7.3f \n',rhot);
    fprintf(fid,'# Pinf : %7.3f \n',Patm);
    fprintf(fid,'# Tblade : %6.2f \n',T);
end
fclose(fid);
clear Results;
[r, k]=size(Cp92);

%Cp Values
Results(1:k,2:r+1)=Cp92(:,x92i)';

%x-locations
Results(:,1)=x92(x92i);
if ReOrdering == 1
    Aux1 = mat2cell(Results, [12, 13], r+1);
    Aux2{1,1}=Aux1{2,1}; % pressure side
    Aux2{2,1}=Aux1{1,1}; % suction side
    Results = cell2mat(Aux2);
end

%Az-station
%xLocations=x25(x25i)';
fprintf(fid, '\n');
fprintf(fid, '# x = 92\n');
if qi25(1,1) == 0
    fprintf(fid, '# x, Pa\n');
else
    fprintf(fid, '# x, Cp\n');
end
fprintf(fid, '# Time ');%Time Azi B1 ');
fprintf(fid, ' %8.5f ',Time);
fprintf(fid, '\n');
fprintf(fid, '# Azi B3 ');%Time Azi B1 ');
fprintf(fid, ' %8.2f ',AzB3);
fprintf(fid, '\n');
fprintf(fid, '# RPM ');%Rpm_signal_phi);
fprintf(fid, ' %8.2f ',Pitch_signal_filr);
fprintf(fid, '\n');
[p, -] = size(Results);
for i=1:p
    fprintf(fid, ' %10.3f',Results(i,:));
    fprintf(fid, '\n');
end
if bCn==1
    fprintf(fid, '\n');
    fprintf(fid, '# Cn ');%Cn92);
    fprintf(fid, ' %10.5f',Ct92);
    fprintf(fid, '\n');
    fclose(fid); clear fid;
end
A.6  Time series values with rows arrangement

```matlab
function Status = aux_time_series_rows(DataFile,bCn,Repro)

% NewMEXICO
% aux_time_series_rows.m
%
% Routine to extract and proccess the information storaged in the hdf5
% file, corresponding to the calibrated pressure. This routine is divided
% on three big sections, the setup, the computation and the saving.
%
% At the end of this routine txt files are save with the information of
% the pressure coefficients. ordered in rows respect the time series
% values.
%
% It is possible to activate or deactivate the computation of some values
% like Cn's or Ct'a, the stress values or just the cannonical values, or
% save the txt file, the default option is all turn on, this means:
%
% bCn = 1
% Repro = 1
% Stres = 1
% bAscii = 1
%
% The routine save one file for each span position, with the name of the
% file R###P###D###_Span_##
%
% Univeristy of Twente
% February / December 2017
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Programmer: E Prieto
%
% % Starting settings (SETUP)

% Clear all old data
clear Cp*
clear Pres*
clear Stress*
clear qi*
clear Blade*
clear Cn*
clear Ct*
clear Puls*
clear jit*
clear FileNamet
clear FileNamez
clear RFiles
%
% % Initial variables
Status = 1;  % Assume success on proccess.
PulseOk = 1;  % Assume the lp singal pulse is ok.
bAscii = 1;  % 1 - Results are saved in ascii file
```
%% Select locations for require directories (SETUP)

% *************************************************************************
% Directory settings
% *************************************************************************

ResultsPathAscii = Directory to store the time series results

ResultsPathAscii = ...
'/Users/Xientifiko/Dropbox/Thesis/Results/NewMEXICO/ascii/Time_series_rows';

TunnelInfoPath = Directory to extract the tunnel information (.dat file)

TunnelInfoPath = ...
'/Users/Xientifiko/Desktop/Thesis/Beehub/home/MexNet/NewMexico/DNW/POL/Steady_Data';

%% Wind turbine settings (SETUP)

% *************************************************************************
% Wind Turbine Settings
% *************************************************************************

Fsam = 5514; % Hz
Span = 2.25; % meters

%% Read the micfile sheet (excel file with information about the sensors) ... (SETUP)

% *************************************************************************
% Excel micfile reading
% *************************************************************************

%Location on computer of the 'micfile' file information, for instance ...
'K:\documents\micfile2.xls'

Mic=xlsread('/Users/Xientifiko/Desktop/Thesis/Beehub/home/MexNet/NewMexico/NewMexico_datapoints.xlsx','micfile');

Chn1P = Mic(:,3)==1; % 1p sensor channel from mic file.
ChnT = Mic(:,3)==5; % Temperature sensor channel from mic.

% *************************************************************************
% Selection of channels that contain pressure sensors information
% This Following notation is used
% (:,3) = Type of sensor 3 = Pressure
% (:,5) = Span posistion 25, 35, 60, 82 or 92
% (:,13) = Status of sensor, 1 = working, 0 = Not working
% (:,6) = Number of blade 1,2 or 3
% *************************************************************************

% Type sensor Section Status ok? Blade
PresChan25_1 = find(Mic(:,3)==3 & Mic(:,5)==25 & Mic(:,13)==1 ); % 27 Sensores for this location
PresChan35_1 = find(Mic(:,3)==3 & Mic(:,5)==35 & Mic(:,13)==1 ); % 27 Sensores for this location
PresChan60_1 = find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,13)==1 & ...
    Mic(:,6)==1 ); % 8 Sensores for this location
PresChan60_2 = find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,13)==1 & ...
    Mic(:,6)==2 ); % 24 Sensores for this location
PresChan60_3 = find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,13)==1 & ...
    Mic(:,6)==3 ); % 4 Sensores for this location
PresChan82.2 = find(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,13)==1 & ...  
    Mic(:,6)==2 );  % 4 Sensors for this location
PresChan82.3 = find(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,13)==1 & ...  
    Mic(:,6)==3 );  % 24 Sensors for this location
PresChan92.3 = find(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,13)==1 );  ...  
    % 26 Sensors for this location

% Read wind tunnel data (SETUP)
% Tunnel Turbine Settings
% Name extraction.
[~,FileName,~] = fileparts(DataFile);
% using function rpdfilename obtain [r], [p] and [d] values
 [~, p ,d] = RPDFilename(FileName);
% Filename wind tunnel data POLPPP
NameFile = strcat(TunnelInfoPath,'/POL',p,'.DAT');
% Verify if file has representative the TunnelInfo
if exist (NameFile,'file')==2
  fid = fopen(NameFile);
  DataInf = textscan(fid,'%f','HeaderLines',8,'Delimiter','|');
  POLtunneldata = DataInf{1,1};
  POLtunneldata = reshape(POLtunneldata,31,[]);
  POLtunneldata = POLtunneldata';
  D=str2double(d);
  Index_D = POLtunneldata(:,1)==D;
  TunnelData = POLtunneldata(Index_D,:);

% Information contained in the tunneldata file, the "commented" values
% can be activated to obtain the specific information.
% PNT = TunnelData(1);  % [-] Data point number
% Time = TunnelData(2);  % [hhmmssxxx] Time
  Qinf = TunnelData(3);  % [Pa] Freestream dynamic pressure
  RHOinf = TunnelData(4);  % [kg/m3] Freestream air density
  Vinf = TunnelData(5);  % [m/s] Freestream tunnel speed
  Beta = TunnelData(6);  % [?] Yaw angle
  Pitch = TunnelData(7);  % [?] Blade pitch angle
  Tinf = TunnelData(8);  % [? K] Freestream tunnel temperature
  % Fx_Rot = TunnelData(9);  % [N] Force in x-direc on (model coordinate system)
  % Fy_Rot = TunnelData(10);  % [N] Force in y-direc on (model coordinate system)
  % Fz_Rot = TunnelData(11);  % [N] Force in z-direc on (model coordinate system)
  % Mx_Rot = TunnelData(12);  % [Nm] Moment around x-axis (model coordinate system)
  % My_Rot = TunnelData(13);  % [Nm] Moment around y-axis (model coordinate system)
  % Mz_Rot = TunnelData(14);  % [Nm] Moment around z-axis (model coordinate system)
  N_Rotor = TunnelData(15);  % [rpm] Rota onal speed of rotor
% Lambda = TunnelData(16); % [-] Tip speed ratio of rotor
% X.Trav = TunnelData(17); % [m] x-coordinate of traverse (tunnel coordinate system)
% Y.Trav = TunnelData(18); % [m] y-coordinate of traverse (tunnel coordinate system)
% Pinf = TunnelData(19); % [Pa] Freestream static pressure
% PTotal = TunnelData(20); % [Pa] Freestream total pressure
% dPitot = TunnelData(21); % [Pa] Pitot tube dynamic pressure (velocity verification only)
% VPitot = TunnelData(22); % [m/s] Pitot tube velocity reading (velocity verification only)
% Current = TunnelData(23); % [-] Generator torque (blade-off only)
% Ceil1 = TunnelData(24); % [Pa] Collector static pressure (Figure 17)
% Ceil2 = TunnelData(25); % [Pa] Collector static pressure (Figure 17)
% Star1 = TunnelData(26); % [Pa] Collector static pressure (Figure 17)
% Star2 = TunnelData(27); % [Pa] Collector static pressure (Figure 17)
% Port1 = TunnelData(28); % [Pa] Collector static pressure (Figure 17)
% Port2 = TunnelData(29); % [Pa] Collector static pressure (Figure 17)
% Floor1 = TunnelData(30); % [Pa] Collector static pressure (Figure 17)
% Floor2 = TunnelData(31); % [Pa] Collector static pressure (Figure 17)
fclose(fid);
else
    Status = 2; % Fail cause no TunnelInfo
    return
end

%% Read all available Pressure calibrated data (SETUP)
% ************************************************************************
% Pressure Calibrated Data *.hdf5 files
% ************************************************************************
[~,NameFile,~] = fileparts(DataFile);
NameFile = strcat('/',NameFile);
PressureData = h5read(DataFile,NameFile);
[NumSam,NumSensors]=size(PressureData);

MeasTime = NumSam / Fsam;
Tiempos=1/Fsam;
Time=0:Tiempos:MeasTime;
Time=Time(1:end-1);

%% Extract RPM and Pitch time series from the data
% ************************************************************************
%dRPM = ...
% Time, Rpm and pitch series. and plot of this values.
% *************************************************************************

% RPM extraction and scaling
Rpm_signal = PressureData(:,165);
Rpm_signal=Rpm_signal*50.42;
figure(1);
plot(Time, Rpm_signal);
hold on
set(gca,'Xtick',0:1:15)
grid minor

% Pitch extraction filtering and scaling
Pitch_signal = PressureData(:,168);
Pitch_signal=(Pitch_signal*19)-49.8;
Pitch_signal_filtr=sgolayfilt(Pitch_signal,1,99);
figure(2);
plot(Time, Pitch_signal_filtr);
hold on
grid minor

%% Pulse Analisys (COMPUTATIONS)
%*************************************************************************

% Pulse Analysis 1p signal
%*************************************************************************
% Verifying is the 1p signal is correct.
if NumSensors < 161
    Status = 5;
    return
end
Pulse=PressureData(:,Mic(Chn1P,12));
Pulse = Pulse * -1;
PulseD = diff(Pulse); % Pulse signal PulseD=Pulse(n)-Pulse(n-1)
if max(PulseD)<3 % If no pulses are present
    if V_inf > 0
        Status = 4; % Either pressure zero run or wrong 1p signal
    else
        Status = 5;
        return
    end
else
    PulseOk = 0;
end
if PulseOk==1
    PulsMax=(max(PulseD))/2;
    [~,PulseI] = findpeaks(PulseD,'MinPeakHeight',PulsMax); % Pulse at ...
    azimuth angle
    NoPuls=length(PulseI);
PulseID = diff(PulseI);
Rpm = NRotor;
Frot = Rpm*(1/60); % Hz speed estimates
clear Pulse;
if (exist('PulseD', 'var')>0)
    clear PulseD;
end
else
    Rpm = N_Rotor;
    NoPuls = 2;
end

%% Processing tunnel parameters (COMPUTATIONS)
%******************************************************************************
% Tunnel parameters, for the comput of the qi's
%******************************************************************************

%% Calculating Temperature
T = mean(PressureData(:,Mic(ChnT,12))); % Average Temperature data, and ... Temperatur Constants for the compute of the real Temperature, this ... values could be referring to stagnation T.
cT1 = 1 / 83.4388;
cT0 = -259.74;
T = cT1*T + cT0;

omega = Rpm/60 * 2 * pi; % Omega Angular velocity (rad/s)
rho = RHOinf; % Air density rho

%% Cps VER 1
% % Qi per station (Pascales)
omega_t = Rpm_signal_filtr/60 * 2 * pi; % Omega Angular velocity (rad/s)
q125 = 0.5 * rho * ((omega_t * 0.25 * Span).^2);
q135 = 0.5 * rho * ((omega_t * 0.35 * Span).^2);
q160 = 0.5 * rho * ((omega_t * 0.60 * Span).^2);
q182 = 0.5 * rho * ((omega_t * 0.82 * Span).^2);
q192 = 0.5 * rho * ((omega_t * 0.92 * Span).^2);

%% Extraction of azimuth position for pressures (COMPUTATIONS)
%******************************************************************************
% Azimuth Angles computation
%******************************************************************************

% Determine sample indexes for the different blades
AzB1=zeros(1,NumSam);
AzB2=zeros(1,NumSam);
AzB3=zeros(1,NumSam);
PPD=zeros(1,NoPuls-1); % Pulse per Degree
if PulseOk==1
    for i=2:NoPuls
        PPD(i-1)=(360/PulseID(i-1));
        Sample_AzB1=PPD(i-1) *(0:PulseID(i-1)-1); % % the term + PPD(i-1) is ... an offset to move the values closer to the real value
        AzB1(PulseI(i-1):PulseI(i)-1)=Sample_AzB1;
    end
    % Adicional Azimuth points before the first pulse
    Before_lst_pulse=length(AzB1(1:PulseI(1)));
    if Before_lst_pulse > PulseID(1)
PPD=(360/Before 1st pulse);
Sample_AzB1i=PPD*(0:Before 1st pulse-1); % Sample for Azimuth Blade 1 initial
else
PPD=(360/PulseID(1));
Sample_AzB1i=PPD*(0:PulseID(1)-1); % Sample for Azimuth Blade 1 final
end
AzB1(1:PulseI(1)-1)=Sample_AzB1i(end-Before 1st pulse+2:end); % Adding values to Azimuth Blade before first pulse
else
PPD=(360/PulseID(1));
Sample_AzB1i=PPD*(0:PulseID(1)-1); % Sample for Azimuth Blade 1 final
end
AzB1(PulseI(1):end)=Sample_AzB1i(2:Last+1); % Adding values to Azimuth Blade 1 after last pulse

% Correction of Azimuth values, from 0 to 223.9 because the 1p signal (is measured always at 223.9 deg)
Ilower = find(AzB1 < 136.1);
Iupper = find(AzB1 > 136.1);
AzB1(Ilower)=AzB1(Ilower)+223.9;
AzB1(Iupper)=AzB1(Iupper)-136.1;

% Azimuth for Blade 2 and Blade 3 for the 3 cases
Index = find(AzB1 > 120 & AzB1 < 240);
AzB2(Index)=AzB1(Index)-120;
AzB3(Index)=AzB1(Index)+120;
clear Index
Index = find(AzB1 > 240 & AzB1 < 360);
AzB2(Index)=AzB1(Index)-120;
AzB3(Index)=AzB1(Index)-240;
clear Index
Index = find(AzB1 > 0 & AzB1 < 120);
AzB2(Index)=AzB1(Index)+240;
AzB3(Index)=AzB1(Index)+120;
clear Index
else
if Vtun > 0
NumSam = NumSam-1;
NumSam = NumSam+1;
else
NumSam = 1;
end
Time = 0;
AzB1 = 1;
AzB2 = AzB1;
AzB3 = AzB1;
end

%% Pressure extraction at the time locations (all) (COMPUTATIONS)
% *************************************************************************
% Extraction of the pressure values in the correct azimuth locations
% computed according the azimuth locations and the ∆.
% *************************************************************************

% Time location extractions
if NumSam > 2
    Pres25_1(:, :) = (PressureData(:, Mic(PresChan25_1, 12)));
    Pres35_1(:, :) = (PressureData(:, Mic(PresChan35_1, 12)));
    Pres60_1(:, :) = (PressureData(:, Mic(PresChan60_1, 12)));
    Pres60_2(:, :) = (PressureData(:, Mic(PresChan60_2, 12)));
    Pres60_3(:, :) = (PressureData(:, Mic(PresChan60_3, 12)));
    Pres82_2(:, :) = (PressureData(:, Mic(PresChan82_2, 12)));
    Pres82_3(:, :) = (PressureData(:, Mic(PresChan82_3, 12)));
    Pres92_3(:, :) = (PressureData(:, Mic(PresChan92_3, 12)));
else
    Pres25_1 = mean(PressureData(:, Mic(PresChan25_1, 12)));
    Pres35_1 = mean(PressureData(:, Mic(PresChan35_1, 12)));
    Pres60_1 = mean(PressureData(:, Mic(PresChan60_1, 12)));
    Pres60_2 = mean(PressureData(:, Mic(PresChan60_2, 12)));
    Pres60_3 = mean(PressureData(:, Mic(PresChan60_3, 12)));
    Pres82_2 = mean(PressureData(:, Mic(PresChan82_2, 12)));
    Pres82_3 = mean(PressureData(:, Mic(PresChan82_3, 12)));
    Pres92_3 = mean(PressureData(:, Mic(PresChan92_3, 12)));
end

%% Cp's values (COMPUTATIONS)
% *************************************************************************
% Computation of Cps values
% *************************************************************************

% Initialize variables for Cp's
Cp25 = zeros(NumSam, length(PresChan25_1));
Cp35 = zeros(NumSam, length(PresChan35_1));
Cp60_1 = zeros(NumSam, length(PresChan60_1));
Cp60_2 = zeros(NumSam, length(PresChan60_2));
Cp60_3 = zeros(NumSam, length(PresChan60_3));
Cp82_2 = zeros(NumSam, length(PresChan82_2));
Cp82_3 = zeros(NumSam, length(PresChan82_3));
Cp92 = zeros(NumSam, length(PresChan92_3));

% %Computation of Cps
% if qi25(1,1) ≠ 0
% % Version 1
for k=1:length(PresChan25_1)
    Cp25(:,k) = ((Pres25_1(:,k))./(Qinf+qi25(:,1)));
end
for k=1:length(PresChan35_1)
    Cp35(:,k) = ((Pres35_1(:,k))./(Qinf+qi35(:,1)));
end
for k=1:length(PresChan60_1)
    Cp60_1(:,k) = ((Pres60_1(:,k))./(Qinf+qi60(:,1)));
end
for k=1:length(PresChan60_2)
    Cp60_2(:,k) = ((Pres60_2(:,k))./(Qinf+qi60(:,1)));
end
for k=1:length(PresChan60_3)
    Cp60_3(:,k) = ((Pres60_3(:,k))./(Qinf+qi60(:,1)));
end
for k=1:length(PresChan82_2)
    Cp82_2(:,k) = ((Pres82_2(:,k))./(Qinf+qi82(:,1)));
end
for k=1:length(PresChan82_3)
    Cp82_3(:,k) = ((Pres82_3(:,k))./(Qinf+qi82(:,1)));
end
for k=1:length(PresChan92_3)
    Cp92(:,k) = ((Pres92_3(:,k))./(Qinf+qi92(:,1)));
end

%% Presentation of results (SAVING RESULTS)
% *************************************************************************
% Presentation of results
% *************************************************************************
% Determine % x-axis values
x25 = Mic(PresChan25_1,7);
x35 = Mic(PresChan35_1,7);
x60_1 = Mic(PresChan60_1,7);
x60_2 = Mic(PresChan60_2,7);
x60_3 = Mic(PresChan60_3,7);
x82_2 = Mic(PresChan82_2,7);
x82_3 = Mic(PresChan82_3,7);
x92 = Mic(PresChan92_3,7);
% Which channels have pressure or suction values
% This Following notation is used
% (;,3) = Type of sensor 3 = Pressure
% (;,5) = Span position 25, 35, 60, 82 or 92
% (;,9) = 1, 0 pressure side -1 suction side
% (;,13) = Work or not work sensor 1 = working, 0 = Not working
% Determination of Cn and Ct (SAVING RESULTS)
% Determine % X-positions pressure side and suction side
% Here the function sortcoord is used to order the positions x on the blade
% to put in the order to print, starting from 0 to 100% for the pressure
% side and then going back from 100% to 0 from the suction side.
x25bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==25 & Mic(:,9)>=0 & Mic(:,13)==1,7));
x25on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==25 & Mic(:,9)<0 & Mic(:,13)==1,7));

153
x25i = sortcoord(x25,x25bo,x25on); % Positions in order
y25 = Mic(PresChan25,1,8);

if bCn==1
    Cn25=zeros(1,NumSam);
    Ct25=zeros(1,NumSam);
    for k=1:NumSam
        Cn25(k) = integrateCp(Cp25(k,x25i),x25(x25i),length(x25bo),1,100,0,0);
        Ct25(k) = integrateCp(Cp25(k,x25i),y25(x25i),length(x25bo),1,0,0,0);
    end
clear x25bo x25on;
endif bCn==1
    Cn35=zeros(1,NumSam);
    Ct35=zeros(1,NumSam);
    for k=1:NumSam
        Cn35(k) = integrateCp(Cp35(k,x35i),x35(x35i),length(x35bo),1,100,0,0);
        Ct35(k) = integrateCp(Cp35(k,x35i),y35(x35i),length(x35bo),1,0,0,0);
    end
clear x35bo x35on;
end
% x60bo = sort(Mic(find(Mic(:,3)==3 & Mic(:,6)==60 & Mic(:,9)>=0 & Mic(:,13)==1),7));
% x60on = sort(Mic(find(Mic(:,3)==3 & Mic(:,6)==60 & Mic(:,9)<0 & Mic(:,13)==1),7));
% x60_1i = sortcoord(x60_1,x60bo,x60on);
% clear x60bo x60on;

x60bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,9)>=0 & Mic(:,13)==1),7));
x60on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,9)<0 & Mic(:,13)==1),7));
x60_2i = sortcoord(x60_2,x60bo,x60on);
y60 = Mic(PresChan60,2,8);

if bCn==1
    Cn60=zeros(1,NumSam);
    Ct60=zeros(1,NumSam);
    for k=1:NumSam
        Cn60(k) = ... integrateCp(Cp60_2(k,x60_2i),x60_2(x60_2i),length(x60bo),1,100,0,0);
        Ct60(k) = ... integrateCp(Cp60_2(k,x60_2i),y60(x60_2i),length(x60bo),1,0,0,0);
    end
end
clear x60bo x60on;

% x60bo = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==3 & ... 
    Mic(:,9)>0 & Mic(:,13)==1),7));
% x60on = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==60 & Mic(:,6)==3 & ... 
    Mic(:,9)<0 & Mic(:,13)==1),7));
% x60,3i = sortcoord(x60,3,x60bo,x60on);
% clear x60bo x60on;

% x82bo = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,6)==2 & ... 
    Mic(:,9)>0 & Mic(:,13)==1),7));
% x82on = sort(Mic(find(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,6)==2 & ... 
    Mic(:,9)<0 & Mic(:,13)==1),7));
% x82,2i = sortcoord(x82,3,x82bo,x82on);
% clear x82bo x82on;

x82bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,6)==3 & Mic(:,9)\n    >= 0 & ... 
    Mic(:,13)==1,7));
x82on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==82 & Mic(:,6)==3 & Mic(:,9)<\n    0 & ... 
    Mic(:,13)==1,7));
x82,3i = sortcoord(x82,3,x82bo,x82on);
y82 = Mic(PresChan82,3,8);

if bCn==1
    Cn82=zeros(1,NumSam);
    Ct82=zeros(1,NumSam);
    for k=1:NumSam
        Cn82(k) = ... 
            integrateCp(Cp82(k,x82,3i),x82(x82,3i),length(x82bo),1,100,0,0);
        Ct82(k) = ... 
            integrateCp(Cp82(k,x82,3i),y82(x82,3i),length(x82bo),1,0,0,0);
    end
end
clear x82bo x82on;

x92bo = sort(Mic(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==3 & Mic(:,9)\n    >= 0 & ... 
    Mic(:,13)==1,7));
x92on = sort(Mic(Mic(:,3)==3 & Mic(:,5)==92 & Mic(:,6)==3 & Mic(:,9)<\n    0 & ... 
    Mic(:,13)==1,7));
x92i = sortcoord(x92,x92bo,x92on);
y92 = Mic(PresChan92,3,8);

if bCn == 1
    Cn92=zeros(1,NumSam);
    Ct92=zeros(1,NumSam);
    for k=1:NumSam
        Cn92(k) = integrateCp(Cp92(k,x92,3i),x92(x92,3i),length(x92bo),1,100,0,0);
        Ct92(k) = integrateCp(Cp92(k,x92,3i),y92(x92,3i),length(x92bo),1,0,0,0);
    end
end

155
%% Save results (SAVING RESULTS)
% *************************************************************************
% Time, number of samples, frequency of sampling etc
% *************************************************************************
if bAscii == 1
    mkdir(fullfile(ResultsPathAscii,FileName));

    FileNamet = strcat(FileName,'_Span25.txt');
    file_name = fullfile(ResultsPathAscii,FileName,FileNamet);

    fid=fopen(file_name,'w');
    fprintf(fid,'# File : %s\n',FileNamet);
    fprintf(fid,'# RPM : %3.1f\n',Rpm);
    fprintf(fid,'# Freqrot : %3.2f\n',Frot);
    fprintf(fid,'# Omega : %3.2f\n',omega);
    fprintf(fid,'# Yaw : %6.2f\n',Beta);
    fprintf(fid,'# Pitch : %6.2f\n',Pitch);
    fprintf(fid,'# Avg : %3.0f\n',NoPuls-2);
    fprintf(fid,'# Vtun : %5.1f\n',V_inf);
    fprintf(fid,'# Ttun : %6.2f\n',T_inf);
    fprintf(fid,'# rhotun : %7.3f\n',rhot);
%    fprintf(fid,'# Pinf : %7.3f\n',Patm);
    fprintf(fid,'# Tblade : %6.2f\n',T);

    clear Results;
    [r,k]=size(Cp25);
    % Cp values
    Results(1:r,5:k+4)=Cp25(:,x25i);
    % x-Values
    Results(:,1)=Time';
    % Azimuth Values Blade 1
    Results(:,2)=AzB1';

    % RPM
    Results(:,3)=Rpm_signal_filtri;

    % Pitch
    Results(:,4)=Pitch_signal_filtri;

    if bCn==1
        % Cn-values
        Results(:,k+3)=Cn25';
        % Ct-values
        Results(:,k+4)=Ct25';
        % Max-Pressure-values
        Results(:,k+5)=Pmax25';
    end

end
% Az-station

xLocations=x25(x25i)';
fprintf(fid, '\r\n');
fprintf(fid, '# x = 25\r\n');
if qi25(1,1) == 0
  fprintf(fid, '# x, Pa[station]\r\n');
else
  fprintf(fid, '# x, Cp[station]\r\n');
end
fprintf(fid, '# x, %', xLocations, ' \r\n');
if bCn==1
  fprintf(fid, ' Cn Ct Pmax \r\n');
end
fprintf(fid, '\r\n');
[p,­]=size(Results);
for i=1:p
  fprintf(fid, ' %9.3f', Results(i,:));
  fprintf(fid, '\r\n');
end
fclose(fid);

%% 2nd cross section ...

% fprintf(fid,'# Pinf : %7.3f \r\n',Patm);

fid=fopen(file_name,'w');
fprintf(fid, '# File : %s',FileName,FileNamet);
fprintf(fid, '# RPM : %3.1f \r\n',Rpm);
fprintf(fid, '# Freqrot : %3.2f \r\n',Frot);
fprintf(fid, '# Omega : %3.2f \r\n',omega);
fprintf(fid, '# Yaw : %6.2f \r\n',Beta);
fprintf(fid, '# Pitch : %6.2f \r\n',Pitch);
fprintf(fid, '# Avg : %3.0f \r\n',NoPuls-2);
fprintf(fid, '# Vtun : %5.1f \r\n',Vinf);
fprintf(fid, '# Ttun : %6.2f \r\n',Tinf);
fprintf(fid, '# rhotun : %7.3f \r\n',rhot);
% clear Results;
[r,k]=size(Cp35);
%Cp Values
Results(1:r,5:k+4)=Cp35(:,x35i);
% x-Values
Results(:,1)=Time';
% Azimuth Values Blade 1
Results(:,2)=AzB1';
% RPM
Results(:,3)=Rpm_signal_filtr;
% Pitch
Results(:,4)=Pitch_signal_filtri;

if bCn==1
    % Cn-values
    Results(:,k+3)=Cn35';
    % Ct-values
    Results(:,k+4)=Ct35';
    % Max-Pressure-values
    Results(:,k+5)=Pmax35';
end

% Az-station
xLocations=x25(x25i)';
fprintf(fid,'r
');
fprintf(fid,'# x = 35\n');
if qi25(1,1) == 0
    fprintf(fid,'# x, Pa[station]\n');
else
    fprintf(fid,'# x, Cp[station]\n');
end
fprintf(fid,'# Time Azi B1 RPM Pitch ');
fprintf(fid,' %7.2f ',xLocations);
if bCn==1
    fprintf(fid,'# Cn Ct Pmax ');
end
fprintf(fid,'r
');
[p,¬]=size(Results);
for i=1:p
    fprintf(fid,' %9.3f',Results(i,:));
    fprintf(fid,'r
');
end
fclose(fid);

%%% 3rd cross section ...

FileNamet = strcat(FileName,'.Span60.txt');
file_name = fullfile(ResultsPathAscii,FileName,FileNamet);

fid=fopen(file_name,'w');
fprintf(fid,'r
');
fprintf(fid,'# File : %s\n',FileNamet);
fprintf(fid,'# RPM : %3.1f\n',Rpm);
fprintf(fid,'# Freqrot : %3.2f\n',Frot);
fprintf(fid,'# Omega : %3.2f\n',omega);
fprintf(fid,'# Yaw : %6.2f\n',Beta);
fprintf(fid,'# Pitch : %6.2f\n',Pitch);
fprintf(fid,'# Avg : %3.0f\n',NoPuls-2);
fprintf(fid,'# Vtun : %5.1f\n',Vinf);
fprintf(fid,'# Ttun : %6.2f\n',Tinf);
fprintf(fid,'# rhotun : %7.3f\n',rhot);
%fprintf(fid,'# Pinf : %7.3f\n',Patm);
fprintf(fid,'# Tblade : %6.2f\n',T);

clear Results;
[r, k]=size(Cp60_2);
%Cp Values
Results(1:r,5:k+4)=Cp60_2(:,x60_2i);
% x-Values
Results(:,1) = Time';
% Azimuth Values Blade 1
Results(:,2) = AzB2';
% RPM
Results(:,3) = Rpm_signal_filtr;
% Pitch
Results(:,4) = Pitch_signal_filtr;
if bCn == 1
  % Cn-values
  Results(:,k+3) = Cn60';
  % Ct-values
  Results(:,k+4) = Ct60';
  % Max-Pressure-values
  Results(:,k+5) = Pmax60';
end
% Az-station
xLocations = x60^2(x60^2i)';
fprintf(fid,'\n');
fprintf(fid,'# x = 60 \n');
if qi25(1,1) == 0
  fprintf(fid,'# x, Pa[station] \n');
else
  fprintf(fid,'# x, Cp[station] \n');
end
fprintf(fid,'# Time Azi B1 RPM Pitch ');
fprintf(fid,' %7.2f ',xLocations);
if bCn == 1
  fprintf(fid,' Cn Ct Pmax ');
end
fprintf(fid,'\n');
[p, ~] = size(Results);
for i = 1:p
  fprintf(fid,' %9.3f ',Results(i,:));
fprintf(fid,'\n');
end
if Repro == 1
  % Add data to other Blades
  clear Results;
  [r, k] = size(Cp60,1);
  %Cp Values
  Results(:,1:r,3:k+2) = Cp60(:,1);
  % x-Values
  Results(:,1) = Time';
  % Azimuth Values Blade 1
  Results(:,2) = AzB1';
  % Az-station
  xLocations = x60,1;
  fprintf(fid,'\n');
  fprintf(fid,'# x = 60 r\n');
if qi25(1,1) == 0
    fprintf(fid,'# x, Pa[station]\r\n');
else
    fprintf(fid,'# x, Cp[station]\r\n');
end
fprintf(fid,'# Time Azi B1 ');
fprintf(fid,' %7.2f ',xLocations);
for i=1:r % Here, k is used because length is the largest size
    fprintf(fid,' %8.3f',Results(i,:));
    fprintf(fid,'#');
    fprintf(fid,' %9.3f',Results(i,:));
end
clear Results;
[r,k]=size(Cp60_3);
%Cp Values
Results(1:r,3:k+2)=Cp60_3;
%x-Values
Results(:,1)=Time';
%Azimuth Values Blade 1
Results(:,2)=AzB3';
%Az-station
xLocations=x60_3;
fprintf(fid,'\r
');
if qi25(1,1) == 0
    fprintf(fid,'# x, Pa[station]\r\n');
else
    fprintf(fid,'# x, Cp[station]\r\n');
end
fprintf(fid,'# Time Azi B3 ');
fprintf(fid,' %7.2f ',xLocations);
for i=1:r % Here, k is used because length is the largest size
    fprintf(fid,' %8.3f',Results(i,:));
    fprintf(fid,'#');
    fprintf(fid,' %9.3f',Results(i,:));
end
fclose(fid);

% 4th cross section ...

FileNamet = strcat(FileName, '_Span82.txt');
file_name = fullfile(ResultsPathAscii, FileName, FileNamet);

fid=fopen(file_name,'w');
fprintf(fid,'# File : %s\r\n',FileName);
fprintf(fid,'# RPM : %3.1f\r\n',Rpm);
fprintf(fid,'# Freqrot : %3.2f\r\n',Frot);
fprintf(fid,'# Omega : %3.2f\r\n',omega);
fprintf(fid,'# Yaw : %6.2f\r\n',Beta);
fclose(fid);
fprintf(fid,'# Pitch : %6.2f
',Pitch);
fprintf(fid,'# Avg : %3.0f
',NoPuls-2);
fprintf(fid,'# Vtun : %5.1f
',Vinf);
fprintf(fid,'# Ttun : %6.2f
',Tinf);
fprintf(fid,'# rhotun : %7.3f
',rhot);
fprintf(fid,'# Pinf : %7.3f
',Patm);
fprintf(fid,'# Tblade : %6.2f
',T);
clear Results;
[r,k]=size(Cp82);
%Cp Values
Results(1:r,5:k+4)=Cp82(:,x82i);
%x-Values
Results(:,1)=Time';
% Azimuth Values Blade 3
Results(:,2)=AzB3';
% RPM
Results(:,3)=Rpm_signal_filtr;
% Pitch
Results(:,4)=Pitch_signal_filtr;
if bCn==1
% Cn-values
Results(:,k+3)=Cn82';
% Ct-values
Results(:,k+4)=Ct82';
% Max-Pressure-values
Results(:,k+5)=Pmax82';
end
% Az-station
xLocations=x82(x82i)';
fprintf(fid,'
');
if qi25(1,1) == 0
fprintf(fid,'# x, Pa[station]
');
else
fprintf(fid,'# x, Cp[station]
');
end
fprintf(fid,'# x = 82
');
fprintf(fid,'# Time Azi B1 RPM Pitch ');
if Repro == 1
 fprintf(fid,' Cn Ct Pmax ');
end
fprintf(fid,' %9.3f',Results(i,:));
fprintf(fid,'
');
end
fprintf(fid,'
');
clear Results;
[r, k]=size(Cp82);
% Cp Values
Results(1:r,3:k+2)=Cp82;
% x-Values
Results(:,1)=Time';
% Azimuth Values Blade 1
Results(:,2)=AzB2';
% Az-station
xLocations=x82;
fprintf(fid,'
');
if qi25(1,1) == 0
fprintf(fid,'# x, Pa[station]
');
else
fprintf(fid,'# x, Cp[station]
');
end
fprintf(fid,'# Time Azi B2 
');
for i=1:r % Here, k is used because length is the largest size
fprintf(fid,'#');
fprintf(fid,' %9.3f',Results(i,:));
fprintf(fid,'
');
end
fclose(fid);

%% 5th cross section ...
*********************************************************
FileName = strcat(FileName,'Span92.txt');
file_name = fullfile(ResultsPathAscii,FileName,FileName);

fid=fopen(file_name,'w');
fprintf(fid,'# File : %s
',FileName);
fprintf(fid,'# RPM : %3.1f
',Rpm);
fprintf(fid,'# Freqrot : %3.2f
',Frot);
fprintf(fid,'# Omega : %3.2f
',omega);
fprintf(fid,'# Yaw : %6.2f
',Beta);
fprintf(fid,'# Pitch : %6.2f
',Pitch);
fprintf(fid,'# Avg : %3.0f
',NoPuls-2);
fprintf(fid,'# Vtun : %5.1f
',Vinf);
fprintf(fid,'# Ttun : %6.2f
',Tinf);
fprintf(fid,'# rhotun : %7.3f
',rhot);
% fprintf(fid,'# Pinf : %7.3f
',Patm);
fprintf(fid,'# Tblade : %6.2f
',T);

clear Results;
[r, k]=size(Cp92);
% Cp Values
Results(1:r,5:k+4)=Cp92(:,x92i);
% x-Values
Results(:,1)=Time';
% Azimuth Values Blade 1
Results(:,2)=AzB3';
% RPM
Results(:,3)=Rpm_signal_filtr;

% Pitch
Results(:,4)=Pitch_signal_filtr;

if bCn==1
    % Cn-values
    Results(:,k+3)=Cn92';
    % Ct-values
    Results(:,k+4)=Ct92';
    % Max-Pressure-values
    Results(:,k+5)=Pmax92';
end

% Az-station
xLocations=x92(x92i)';
fprintf(fid,'%r\n');
fprintf(fid,'# x = %r\n');
if qi25(1,1) == 0
    fprintf(fid,'# x, Pa[station]\r\n');
else
    fprintf(fid,'# x, Cp[station]\r\n');
end
fprintf(fid,'# Time Azi B1 RPM Pitch ');% Time Azi B1 RPM Pitch 
if bCn==1
    fprintf(fid,' Cn Ct Pmax ');% Cn Ct Pmax 
end
fprintf(fid,' %7.2f ',xLocations);
if bCn==1
    fprintf(fid,' %9.3f',Results(i,:));% Time Azi B1 RPM Pitch 
end
fclose(fid);
B Complete correlation of configurations and files for the New MEXICO experiment

During the New MEXICO experiment a vast amount of information was recorded, and this information if organised in a test matrix showed in the Table B2. The order of the columns follows to a large extent the test schedule in chronological order. The measurement apparatus used for each part of the test is shown in the first row. The row label as Model configuration indicates whether roughness, add-ons or other features have been applied to the model. The Table B1 give an explanation of the numbering in this row. Finally, the operational condition section shows an overview of the pitch angles, yaw angles, rotational speed and tunnel speeds that have been applied for each part of the test. The full description of the experimental order and setup can be found in [66].

Table B1: Model configuration legends for the New MEXICO

<table>
<thead>
<tr>
<th>Legend number</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Roughness on full blade</td>
</tr>
<tr>
<td>1</td>
<td>Guerney flaps long</td>
</tr>
<tr>
<td>2</td>
<td>Guerney flaps short</td>
</tr>
<tr>
<td>3</td>
<td>Outboard blade clean</td>
</tr>
<tr>
<td>4</td>
<td>Spoilers</td>
</tr>
<tr>
<td>5</td>
<td>Serrations</td>
</tr>
<tr>
<td>6</td>
<td>Pitch misalignment B2 (−20°)</td>
</tr>
<tr>
<td>7</td>
<td>Oil flow: sensors taped off</td>
</tr>
<tr>
<td>99</td>
<td>Blade off</td>
</tr>
</tbody>
</table>
Table B2: Correlation of data for cases and files of the New MEXICO experiment

<table>
<thead>
<tr>
<th>Test type</th>
<th>Velocity verification</th>
<th>Load vs Velocity</th>
<th>Standstill (pressure)</th>
<th>Axial flow (pressure)</th>
<th>PIV</th>
<th>Dynamic inflow</th>
<th>Yawed flow (pressure)</th>
<th>Blad add-ons</th>
<th>Pitch misalignment</th>
<th>Flowviz</th>
<th>Blade-off</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMO</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>(x)</td>
<td>(x)</td>
</tr>
<tr>
<td>Balance</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PIV (+pitot)</td>
<td>axial</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PIV transverse</td>
<td>radial</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mics</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Array</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Apparatus       | Pitch angle [°]        | 90              | -5.3 → 1.7            | -2.3 → 90             | -5.3 → 1.7 | -2.3       | -2.3, 0.7              | -5.3 → 1.7 | -5.3 → 20         | -2.3, 73.6 | NA         |
|                 | Yaw angle [°]          | 0               | 0                     | 90 → 30               | 0            | -30,0,30   | 0,15,30               | -30 → 45     | 0                 | 0                 | -30 → 30   |
|                 | Rot Speed [rpm]        | 0               | 0                     | 324, 424              | 0            | 324, 424   | 424                   | 0,324,424    | 324               | 0,324,424 | 0.324,424  |
|                 | $U_\infty$ [ms]        | 10 → 30         | 7.5 → 24              | 30                    | 5 → 30       | 10,15,24   | 10,15,18,24           | 5 → 30       | -5 → 15          | 15,18,30 | 10 → 30    |

† Legend clarification in Table B1
Table C1: Specifications of Kulite® XCQ-95-062-5A.

<table>
<thead>
<tr>
<th>INPUT</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Pressure Range</td>
<td>35 kPa (5 psiA)</td>
</tr>
<tr>
<td>Measuring range</td>
<td>40 kPa to 106 kPa Absolute (5.8 to 15.37 psiA)</td>
</tr>
<tr>
<td>Operational Mode</td>
<td>Absolute</td>
</tr>
<tr>
<td>Over Pressure</td>
<td>3 times rated pressure</td>
</tr>
<tr>
<td>Burst Pressure</td>
<td>4 times rated pressure</td>
</tr>
<tr>
<td>Pressure Media</td>
<td>All non conductive, non corrosive Liquids or Gas</td>
</tr>
<tr>
<td>Rated Electrical</td>
<td>Excitation 10 VDC/AC</td>
</tr>
<tr>
<td>Maximum Electrical Excitation</td>
<td>15 VDC/AC</td>
</tr>
<tr>
<td>Input Impedance</td>
<td>800 (Min.)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OUTPUT</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Output Impedance</td>
<td>1000 ohms (Nom.)</td>
</tr>
<tr>
<td>Full Scale Output (FSO)</td>
<td>90 mV (Nom.)</td>
</tr>
<tr>
<td>Output at 14.5 psiA (100KPaA)</td>
<td>270 mV (Nom)</td>
</tr>
<tr>
<td>Residual Unbalance</td>
<td>±3% FSO</td>
</tr>
<tr>
<td>Combined Non-Linearity and Hysteresis</td>
<td>±0.25% FS BFSL</td>
</tr>
<tr>
<td>Hysteresy</td>
<td>Less Than 0.1% (typ.)</td>
</tr>
<tr>
<td>Repeatability</td>
<td>±0.1%</td>
</tr>
<tr>
<td>Resolution</td>
<td>Infinite</td>
</tr>
<tr>
<td>Natural Frequency (KHz)</td>
<td>&gt; 150 KHz</td>
</tr>
<tr>
<td>Acceleration Sensitivity % FS/g Perpendicular</td>
<td>0.002</td>
</tr>
<tr>
<td>Transverse</td>
<td>0.0005</td>
</tr>
<tr>
<td>Insulation Resistance</td>
<td>50 Megohm Min. at 100 VDC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ENVIRONMENTAL</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating Temperature Range</td>
<td>-55 °C to +120 °C</td>
</tr>
<tr>
<td>Compensated Temperature Range</td>
<td>+10 °C to +65 °C</td>
</tr>
<tr>
<td>Thermal Zero Shift</td>
<td>±1%/55 °C (typ.)</td>
</tr>
<tr>
<td>Thermal Sensitivity Shift</td>
<td>±1%/55 °C (typ.)</td>
</tr>
<tr>
<td>Steady Acceleration</td>
<td>10.000 g (max.)</td>
</tr>
<tr>
<td>Linear Vibration</td>
<td>10 – 2000 Hz Sine, 100 g max.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PHYSICAL</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Electrical Connection</td>
<td>4 leads AWG 38 (dia including Teflon insulator 0.23 mm). Length between transducer and TC module to be specified later. Length after module to be specified later.</td>
</tr>
<tr>
<td>Housing length</td>
<td>2.54 mm (0.1&quot;)</td>
</tr>
<tr>
<td>Compensation module</td>
<td>2.8 mm dia × 25.4 mm long (0.110 × 1&quot; long)</td>
</tr>
<tr>
<td>Weight</td>
<td>0.2 Gram (Nom.) Excluding Module and Leads</td>
</tr>
<tr>
<td>Diaphragm protection</td>
<td>B-screen</td>
</tr>
<tr>
<td>Sensing Principle</td>
<td>Fully Active Four Arm Wheatstone Bridge Diffused into Silicon Diaphragm</td>
</tr>
</tbody>
</table>
## D Example of file generated with Matlab routine

Table D1: Example of file generated with Matlab routines.

<table>
<thead>
<tr>
<th># File</th>
<th>R52P81D940_Span25.txt</th>
</tr>
</thead>
<tbody>
<tr>
<td># RPM</td>
<td>424.1</td>
</tr>
<tr>
<td># Freqrot</td>
<td>7.08</td>
</tr>
<tr>
<td># Omega</td>
<td>44.52</td>
</tr>
<tr>
<td># Yaw</td>
<td>30.01</td>
</tr>
<tr>
<td># Pitch</td>
<td>-2.30</td>
</tr>
<tr>
<td># Avg</td>
<td>33</td>
</tr>
<tr>
<td># Vtun</td>
<td>15.0</td>
</tr>
<tr>
<td># Ttun</td>
<td>293.20</td>
</tr>
<tr>
<td># rhotun</td>
<td>1.205</td>
</tr>
<tr>
<td># Tblade</td>
<td>23.38</td>
</tr>
<tr>
<td>x = 25</td>
<td></td>
</tr>
<tr>
<td>x, Cp[station] Error[station]</td>
<td></td>
</tr>
<tr>
<td>#</td>
<td>0.000 60.000 120.000 180.000 240.000 300.000 0.000 60.000 120.000 180.000 240.000 300.000</td>
</tr>
<tr>
<td>89.650</td>
<td>-0.085 -0.018 0.181 0.226 0.161 0.043 -0.256 -0.257 -0.251 -0.248 -0.248 -0.255</td>
</tr>
<tr>
<td>80.540</td>
<td>0.525 0.531 0.696 0.754 0.736 0.641 -0.255 -0.254 -0.256 -0.252 -0.254 -0.256</td>
</tr>
<tr>
<td>69.730</td>
<td>0.222 0.194 0.366 0.331 0.342 0.299 -0.251 -0.252 -0.255 -0.255 -0.256 -0.259</td>
</tr>
<tr>
<td>60.900</td>
<td>0.006 -0.022 -0.017 -0.057 0.020 0.072 -0.258 -0.259 -0.257 -0.258 -0.259 -0.259</td>
</tr>
<tr>
<td>50.590</td>
<td>-0.004 -0.049 -0.208 -0.384 -0.201 0.026 -0.259 -0.259 -0.257 -0.258 -0.258 -0.258</td>
</tr>
<tr>
<td>40.740</td>
<td>-0.117 -0.235 -0.583 -0.864 -0.624 -0.249 -0.256 -0.256 -0.258 -0.258 -0.259 -0.259</td>
</tr>
<tr>
<td>33.180</td>
<td>0.518 0.448 0.100 -0.256 0.052 0.478 -0.259 -0.260 -0.258 -0.258 -0.255 -0.259</td>
</tr>
<tr>
<td>15.720</td>
<td>0.299 0.331 0.082 -0.308 -0.060 0.274 -0.258 -0.260 -0.256 -0.258 -0.258 -0.261</td>
</tr>
<tr>
<td>3.330</td>
<td>0.165 0.454 0.833 0.715 0.680 0.440 -0.259 -0.258 -0.257 -0.258 -0.260 -0.258</td>
</tr>
<tr>
<td>0.950</td>
<td>-0.324 0.114 1.028 1.280 1.013 0.271 -0.256 -0.253 -0.256 -0.261 -0.259 -0.256</td>
</tr>
<tr>
<td>0.190</td>
<td>-1.264 -0.773 0.544 1.170 0.735 -0.413 -0.252 -0.246 -0.243 -0.259 -0.255 -0.252</td>
</tr>
<tr>
<td>0.000</td>
<td>-2.215 -1.815 -0.352 0.616 0.107 -1.247 -0.246 -0.242 -0.232 -0.249 -0.247 -0.247</td>
</tr>
<tr>
<td>0.200</td>
<td>-2.838 -2.651 -1.392 -0.283 -0.743 -2.005 -0.246 -0.239 -0.223 -0.248 -0.250 -0.247</td>
</tr>
<tr>
<td>0.940</td>
<td>-3.639 -3.637 -3.637 -3.640 -3.643 -3.639 -0.258 -0.258 -0.257 -0.258 -0.258 -0.258</td>
</tr>
<tr>
<td>2.540</td>
<td>-2.383 -2.302 -1.947 -1.347 -1.469 -1.930 -0.248 -0.248 -0.248 -0.251 -0.255 -0.255</td>
</tr>
<tr>
<td>5.300</td>
<td>-2.012 -2.038 -2.197 -1.865 -1.845 -1.973 -0.239 -0.243 -0.240 -0.253 -0.257 -0.257</td>
</tr>
<tr>
<td>10.570</td>
<td>-1.416 -1.558 -1.877 -1.714 -1.558 -1.452 -0.253 -0.252 -0.245 -0.254 -0.259 -0.257</td>
</tr>
<tr>
<td>15.640</td>
<td>-1.930 -1.964 -2.259 -2.300 -2.030 -1.778 -0.255 -0.252 -0.246 -0.253 -0.256 -0.257</td>
</tr>
<tr>
<td>22.490</td>
<td>-1.245 -1.345 -1.770 -1.914 -1.566 -1.213 -0.255 -0.252 -0.246 -0.253 -0.256 -0.257</td>
</tr>
<tr>
<td>29.670</td>
<td>-0.669 -0.801 -1.340 -1.527 -1.154 -0.711 -0.256 -0.248 -0.249 -0.256 -0.257 -0.259</td>
</tr>
<tr>
<td>38.580</td>
<td>-0.622 -0.711 -1.224 -1.426 -1.109 -0.708 -0.258 -0.253 -0.243 -0.250 -0.258 -0.257</td>
</tr>
<tr>
<td>50.660</td>
<td>-0.711 -0.758 -1.093 -1.281 -1.031 -0.722 -0.254 -0.250 -0.252 -0.257 -0.258 -0.259</td>
</tr>
<tr>
<td>62.040</td>
<td>-0.273 -0.358 -0.411 -0.567 -0.386 -0.182 -0.252 -0.247 -0.252 -0.257 -0.258 -0.256</td>
</tr>
<tr>
<td>72.110</td>
<td>-0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264</td>
</tr>
<tr>
<td>84.190</td>
<td>-0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264</td>
</tr>
<tr>
<td>91.990</td>
<td>-0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264 -0.264</td>
</tr>
</tbody>
</table>