NTNU

Norwegian University of
Science and Technology

Value of Virtual Sensing on Offshore Windturbines

Francois Nicolas Emmanuel Million

Civil and Environmental Engineering
Submission date: July 2018
Supervisor: Michael Muskulus, IBM

Norwegian University of Science and Technology
Department of Civil and Environmental Engineering

Abstract

This master thesis project concerns the structural monitoring applied to a windturbine. The algorithm analysed in this master thesis is an extended Kalman Filter called the joint input-state estimation algorithm.

Kalman Filter algorithm is usually fed with a measured signal that is to be filtered and some model parameter. The filtering process is based on a weighted average between the model and the measure at each time step. The weighting process involve calculation of noise covariance matrices. This master thesis is an analysis on Kalman Filtering in case of an operating windturbine, the measured signal correspond to motion data, either displacement, velocity or acceleration taken at a given height. As on a windturbine, the excitation force applied through the rotor can not be exactly known, an extended version of the Kalman Filter called the joint input-state estimation algorithm will be used. This algorithm estimate at the same time the state and the input -excitation force - of the system.

A special attention is given to detection and quantification of the modeling error which is the main personal contribution to this field. A process to include modeling error in calculation of covariance matrices is given and an analysis of the impact of each dynamical model parameter error on the overall estimation error is carried out. Besides, the stability of the algorithm is also a concern in this thesis. Finally, an optimisation of the sensor layout is performed.

As the Kalman Filter is linear, in the first instance algorithm will be analysed for periodic sinusoidal signals as every signal can be seen as the infinite sum of harmonic signals. Then, algorithm will be analysed on a virtual windturbine through FedEm®. Advantage of virtual simulation is that the user can control the importance of stochastic parameters.

Contents

List of figures

List of Tables

Introduction

Developpement of renewable energies and sucess of energy transition is one of the biggest challenge of the 21st century. This challenge relies, among other sources of renewable energy, on a fast and significant increase of wind energy business. Wind energy production can be divided in two families : onshore and offshore. Both present advantages and inconvenients but nowadays, technical reasons bring onshore production to the foreground of wind energy. According to Wind Europe report *Wind in Power 2017*[20], wind energy in Europe is the second largest form of power generation - after gas installations - with a net installed capacity of 168.7 GW, which represent 18% of the total net EU-installed power in 2017. Onshore production represents a net installed power of 153 GW and offshore production reach 15.8 GW. Offshore is still far behind onshore but the net installed power in 2017 can give us an indication on how fast the sector is developping. In 2017, Europe instlled 15.638 GW of new wind power capacity ; 12.484 of which were onshore and 3.154 offshore. Compared to 2016, onshore grew 14.3% while offshore grew 101%. This numbers show how promising the offshore wind sector can be. Also, among other advantages, offshore wind solve the problem of space as the sea space is usually unused. This advantage is very attractive in a world with growing population and growing needs in terms of energy and food. The trend of wind energy from 2005 to 2017 is very encouraging but EU has high-level objectives in terms of renewable energy, which are quite challenging for the sector.

Wind energy challenges

The targets in terms of percentage of renewable in the energy mix are constantly reviewed but several actors, including WindEurope agree to aim for 35% for 2030. Proaction studies are ongoing to determine the necesary today's actions to fulfill this objective. Wind Europe present in a report *Wind energy in Europe: Scenarios for 2030*[21] three different scenarii for wind energy. These scenarii are in line with EU-objectives but they require some breakthrough, especially in offshore wind, to be reached. Indeed, offshore wind is now too expensive compared to onshore wind or to other sources of energy production to be really competitive. Wind Europe expect the offshore wind cost to be below 80€/MWh before 2025 for every windfarm, *i.e.* any depth and any kind of turbine. This cost reduction can be realised at various levels : wind turbine construction, maintenance and operation, lifetime extension... And, as presented in this report, an equilibrium has to be found between these parts of a windturbine project.
The life time extension issue is an essential question because the first offshore windturbine date back to the end of the 1990's, which means that the first offshore windturbine reached or will

soon reach the end of their designed lifetime. A windturbine demanteling is a huge and onerous operation especially since it has never been done for any offshore windturbine. Are all the parts of the turbine obsolete or are some parts reusable ? It is a new operation with big impacts on environement and economy. That's why, at the end of a windturbine lifetime, some specific questions must be studied in detail. Knowing that producing electricity with a turbine creates money but also damages the turbine, one has to wonder if there is an optimal solution between running the windturbine and stopping it to protect it. One may also wonder if it is worth the risk to run the windturbine knowing that it could collapse.
In order to have quantitative answers to these questions, wind turbines are fitted with different kinds of sensors aimed to estimate the windturbine damage all along its lifetime. For economic reasons, number and type of sensors are limited, e.g. a strain gauge is much more expensive and complex to put on a windturbine than an accelerometer. Hence, having an efficient sensor distribution along the windturbine monopile is essential to compute a good damage estimation at relatively low costs. This problematic will be the main topic of the thesis.

Motivation

Extracting acurate data from sensors on a windturbine can have a lot of impacts on energy production costs. Not only can it extend the turbines lifetime thanks to an acurate fatigue damage calculation, but it can also make the maintenance and operation (M & O) more effective. Especially in the field of offshore wind, M & O needs considerable means due to inaccessibility and random - and possibly strong - weather conditions. These operations often represent a large amout of money hence M & O is an important issue in the process of offshore wind cost reduction.
This work intent to provide a better understanding of the filtering processes used for wind turbine monitoring, which could lead to a better estimation of wind turbine displacement and hence a better estimation on fatigue damage.
Besides, the problem of extrapolating data from a sensor distribution is a wide problem in structural engineering. This work can be valuable for other fields than wind turbine, as it is mainly a study of Kalman Filter (KF) algorithm, a filter widely used in dynamic engineering.
As a matter of fact the reason why this thesis is made are both for trying to provide a deeper understanding of dynamical behavior of an offshore wind turbine and also to deepen my knowlege as a master student.

State of the art

The Kalman Filter, first published by Rudholf E. Kalman in 1960 [6] is a very powerful tool for signal processing and for system tracking. Its aim is to filter a signal resulting from a set of deterministic and stochastic processes. Concerning its application on structural monitoring, the filter and its application has been widely analyzed, among others, by E. Lourens and K. Maes whose work is often referenced in this report.

The Kalman filter algorithm is able to compute an output estimation, given a model, measure-

ment data and an input following a process explained through a sketch in Appendix A. However, in most of the real structural dynamic systems such as buildings, bridges or monopiles, the excitation force is caused by several factors and can often be realted to stochastic processes. Therefore, exact excitation forces are rarely well known. Taking into account this important problem in study of structural monitoring, S. Gillijns and B. De Moor developped in 2007 a joint input-state estimation algorithm [2]. This algorithm can be considered as an extended Kalman filter and enable to estimate both the input and the output of the system - excitation force and displacement or velocity or acceleration in case of a dynamic system - through a weight average process between model and measure. This weight tells the algorithm how trustful the model and the measurement are. Computing the correct weight is all the mater of this thesis.
However, the algorithm developed by Gillijns and De Moor assumes that the different stochastic processes are all uncorrolated, which can be - in some cases - too simplistic. In 2016, K. Maes put forward a joint input-state estimation algorithm update in [11] that takes into account the potential correlation between stocastic processes. This most recent algorithm is the one used and analysed in this report.

As explained further, the Kalman Filter is based on a weight average process between measures and a model. Uncertainty can affect both the measures and the model, however, in almost all the Kalman Filter analysis done so far, the modeling uncertainties are neglected. Indeed, in presence of modeling errors, the true dynamic behavior of the system is unknown and all the estmations computed by the algorithm are biased. The influence of modeling error is very wide and complex and hence often assumed to be meaningless for sake of simplicity.
In this report, a modeling uncertainty quantification based on operational modal analysis will be put forward. Furthermore, the different parameters of the dynamic model and their impact on the estimated response of the Kalmen Filter will be analysed one by one. Virtual simulation enables to simply measure the modeling error by difference between simulations with and without model uncertainties.

Objectives

The objectives of this thesis can be cathegorized in studying two main problems. During its lifetime, a windturbine is subject to a lot of random factors which can be taken into account in the filter as stochastic variables.

First, a detail analysis of KF algorithm will aim at having a better understanding of stability and robustness of this filter, face to this stochastic variables and other sources of error. A special care will be given to study the algorithm reaction face to model uncertainties, trying to include their contribution within the joint input-state estimation algorithm.

The second objective is to use this filter for windturbine monopile monitoring and determine if there is an optimal sensor distribution along the monopile which would minimize the total error

on motion. In other words, existence of critical points where data can not be extrabolated will be studied.

Structure

This thesis will be divided in two main parts. In a first instance, a mathematical formulation of the KF applied to a dynamic system will be presented. After giving a general definition of dynamical system, the Kalman Filter algorithm for joint input-state estimation will be analysed closely in terms of uncertainty propagation in order to have a better idea about stability of the filter. For the first part of this thesis all the examples will be given with a single degree of freedom (SDOF) system for sake of simplification.

In a second phase, an example of KF aplication on a monopile will be carried on. After defining a model for the wind turbine time evolution, and after generating motion data through virtual sensing thanks to FedEm software, the performances of the joint input-state algorithm will be shown and discussed. Finally an optimization of the sensor layout on the wind turbine aiming to minimize the estimated output error will be put forward.

Note: All along the report, the joint input-state estimation algorithm applied on a dynamic system will be studied. The system input and output are different from the algorithm input and output. The system input refers to the force applied on the system and the system output is either displacement, velocity or acceleration. Besides, the algorithm input refers to the measurements of the system output and the algorithm output correspond to the filtered system output signals and to the system input.

Chapter 1

Analysis of Kalman Filter algorithm

1.1 Dynamical system model

1.1.1 Equation of motion

Consider a continuous-time equation of motion for a dynamical system with n_{dof} degrees of freedom:

$$\mathbf{M}\ddot{u}(t) + \mathbf{C}\dot{u}(t) + \mathbf{K}u(t) = \mathbf{f}(t) = \mathbf{S_p}(t)\mathbf{p}(t) \tag{1.1}$$

With :
u(t) $\in \mathbb{R}^{n_{dof}}$ the vector of displacement,
$\mathbf{M}$,$\mathbf{C}$ and $\mathbf{K} \in \mathbb{R}^{n_{dof} \times n_{dof}}$ are the mass, damping and stiffness matrices respectively,
$\mathbf{f}$(t) is the excitation vector.

If a number n_p of different forces are applied on the system, the excitation vector is often decomposed into an input force influence matrix $\mathbf{S_p}(t) \in \mathbb{R}^{n_{dof} \times n_p}$ and a vector $\mathbf{p}(t) \in \mathbb{R}^{n_p}$. The vector $\mathbf{p}(t)$ contains the information of the n_p forces time history, and, each column of the matrix $\mathbf{S_p}$ gives the spatial distribution of the load in the corresponding element of vector $\mathbf{p}(t)$. If the load distribution changes in time then $\mathbf{S_p}$ becomes time dependant and is written $\mathbf{S_p}(t)$. In the particular case of a SDOF system with a time dependant excitation applied on the system, the matrix $\mathbf{S_p}$ is simply 1 and the vector $\mathbf{p}(t)$ is the value of the force at time t.

In order to write the governing equation in modal coordinates, the undamped eigenvalue problem [5] corresponding to 1.2 is solved:

$$\mathbf{K\Phi} = \mathbf{M\Phi\Omega}^2 \tag{1.2}$$

Where $\mathbf{\Phi} \in \mathbb{R}^{n_{dof} \times n_{dof}}$ is the matrix formed by the eigenvectors, and $\mathbf{\Omega}$ is a matrix containing the eigenfrequencies ω_j in rad/s on its diagonal.

The displacement vector in modal coordinates $z(t)$ is obtained by applying the coordinates transformation from space coordinates to modal coordinates to displacement vector $u(t)$: $z(t) =$

$\mathbf{\Phi}u(t)$. Premultiplying 1.1 by $\mathbf{\Phi^T}$ yields:

$$\mathbf{\Phi^T M \Phi}\ddot{z}(t) + \mathbf{\Phi^T C \Phi}\dot{z}(t) + \mathbf{\Phi^T K \Phi} z(t) = \mathbf{\Phi^T S_p}(t)\mathbf{p}(t) \tag{1.3}$$

Assuming that mass normalized eigenvectors are used, *i.e.* $\mathbf{\Phi^T M \Phi} = \mathbf{I}$ and using equation 1.2 leads to $\mathbf{\Phi^T K \Phi} = \mathbf{\Omega}^2$. The assumption of proportional damping leads to $\mathbf{\Phi C \Phi^T} = \mathbf{\Gamma}$, where $\mathbf{\Gamma}$ contains the terms $2\xi_j\omega_j$ on its diagonal with ξ the modal damping ratio.
The governing equation of motion in modal coordinates then becomes:

$$\ddot{z}(t) + \mathbf{\Gamma}\dot{z}(t) + \mathbf{\Omega}^2 z(t) = \mathbf{\Phi^T S_p}(t)\mathbf{p}(t) \tag{1.4}$$

1.1.2 Continuous time state-space model

The state vector $\mathbf{x}(t)$ is defined as:

$$\mathbf{x}(t) = \begin{pmatrix} u(t) \\ \dot{u}(t) \end{pmatrix} \tag{1.5}$$

Once the state vector is defined,equation 1.1 can be written in matrix form as:

$$\dot{\mathbf{x}}(t) = \mathbf{A_c}\mathbf{x}(t) + \mathbf{B_c}\mathbf{p}(t) \tag{1.6}$$

With $\mathbf{A_c}$ and $\mathbf{B_c}$ the system matrices defined as $\mathbf{A_c} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{M}^{-1}\mathbf{K} & -\mathbf{M}^{-1}\mathbf{C} \end{bmatrix}$, $\mathbf{B_c} = \begin{bmatrix} \mathbf{0} \\ \mathbf{M}^{-1}\mathbf{S_p} \end{bmatrix}$

The dynamic system contains an observator able to measure n_d quantities. These observed quantities are expressed in the measurement data vector $\mathbf{d}(t) \in \mathbb{R}^{n_d}$. The observed quantities can be expressed as a linear combination of acceleration, velocity and displacement:

$$\mathbf{d}(t) = \mathbf{S_a}\ddot{u}(t) + \mathbf{S_v}\dot{u}(t) + \mathbf{S_d}u(t) \tag{1.7}$$

Where $\mathbf{S_a}$, $\mathbf{S_v}$ and $\mathbf{S_d}$ are the selection matrices. As an example in a SDOF system containing one accelerometer and one strain gauge, *i.e.* measured data contains acceleration and dislacement, $\mathbf{S_a} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$, $\mathbf{S_v} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ and $\mathbf{S_d} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$.

Considering the state vector $\mathbf{x}(t)$ in equation 1.5 and using equation 1.1, the output measured data vector defined in equation 1.7 can also be expressed in state-space form as:

$$\mathbf{d}(t) = \mathbf{G_c}\mathbf{x}(t) + \mathbf{J_c}\mathbf{p}(t) \tag{1.8}$$

Where $\mathbf{G_c} \in \mathbb{R}^{n_d \times n_s}$, $n_s = 2n_{dof}$ is called the output influence matrix and $\mathbf{J_c} \in \mathbb{R}^{n_d \times n_p}$ is the direct transmission matrix.
They are defined as

$$\mathbf{G_c} = \begin{bmatrix} \mathbf{S_d} - \mathbf{S_a}\mathbf{M}^{-1}\mathbf{K} & \mathbf{S_v} - \mathbf{S_a}\mathbf{M}^{-1}\mathbf{K} \end{bmatrix} \tag{1.9}$$

$$\mathbf{J_c} = \begin{bmatrix} \mathbf{S_a}\mathbf{M}^{-1}\mathbf{S_p} \end{bmatrix} \tag{1.10}$$

In case of a model reduction, *i.e.* when the dynamics of the system are expressed by a reduced number n_m of modal coordinates $z(t)$ as $u(t) = \mathbf{\Phi_r} z(t)$, the state vector is transformed accordingly:

$$\mathbf{x}(t) = \begin{bmatrix} \mathbf{\Phi_r} & 0 \\ 0 & \mathbf{\Phi_r} \end{bmatrix} \zeta(t) \tag{1.11}$$

The modal state-space system model is then:

$$\dot{\zeta}(t) = \mathbf{A_c}\zeta(t) + \mathbf{B_c}\mathbf{p}(t) \tag{1.12}$$

$$\mathbf{d}(t) = \mathbf{G_c}\zeta(t) + \mathbf{J_c}\mathbf{p}(t) \tag{1.13}$$

Where $\mathbf{A_c}$, $\mathbf{B_c}$, $\mathbf{G_c}$ and $\mathbf{J_c}$ are defined as:

$$\mathbf{A_c} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{\Omega}^2 & -\mathbf{\Gamma} \end{bmatrix}, \tag{1.14}$$

$$\mathbf{B_c} = \begin{bmatrix} \mathbf{0} \\ \mathbf{\Phi^T S_p} \end{bmatrix}, \tag{1.15}$$

$$\mathbf{G_c} = \begin{bmatrix} \mathbf{S_d \Phi} - \mathbf{S_a \Phi \Omega}^2 & \mathbf{S_v \Phi} - \mathbf{S_a \Phi \Gamma} \end{bmatrix}, \tag{1.16}$$

$$\mathbf{J_c} = \begin{bmatrix} \mathbf{S_a \Phi \Phi^T S_p} \end{bmatrix}. \tag{1.17}$$

1.1.3 Discretization of continuous time state space system model

A sampling frequency of $1/\Delta t$ is used to discretize the continuous time state space system model:

$$\mathbf{x_{k+1}} = \mathbf{A x_k} + \mathbf{B p_k} \tag{1.18}$$

$$\mathbf{d_k} = \mathbf{G x_k} + \mathbf{J p_k} \tag{1.19}$$

With $\mathbf{x_k} = \mathbf{x}(k\Delta t)$, $\mathbf{d_k} = \mathbf{d}(k\Delta t)$, $\mathbf{p_k} = \mathbf{p}(k\Delta t)$, $k = 1,...,N$.
It is shown in [7] that the matrices $\mathbf{A}$ and $\mathbf{B}$ can be expressed as:

$$\mathbf{A} = e^{\mathbf{A_c}\Delta t}\ ,\ \mathbf{B} = \int_0^{\Delta t} e^{\mathbf{A_c}\lambda} d\lambda \mathbf{B_c} = [\mathbf{A} - \mathbf{I}]\mathbf{A_c}^{-1}\mathbf{B_c}\ ,\ \mathbf{G} = \mathbf{G_c}\ ,\ \mathbf{J} = \mathbf{J_c}$$

In case of a small Δt compared to characteristic time of the system, it is possible to use a linear approximation saying that $\dot{\mathbf{x}}_{\mathbf{k+1}} = \dfrac{\mathbf{x_{k+1}} - \mathbf{x_k}}{\Delta t}$ to symplify the expression of $\mathbf{A}$:

$\mathbf{A} = \mathbf{I} + \Delta t \mathbf{A_c}$.

1.2 Kalman Filter and joint input-state estimation

Any kind of dynamic system in real life is subject to known excitations and stochastic unknown excitation. In addition, the sensors composing the system observor contain some noise which can

be very annoying especially in fatigue analysis. Indeed noise in the motion signal can be interpreted as a lot of charge/discharge cycles while it is not. The filtering process and especially the Kalman Filter algorithm aims to estimate accurately the state of the system thanks to a model implemented in the filter through matrices $\mathbf{A}$ and $\mathbf{B}$. This model is theorical and can not describe the system perfectly neither. These errors are accounted for in the system model by stochastic noise process $\mathbf{w_k}$ and measurement noise $\mathbf{v_k}$. When these noises are added to equation 1.18 and 1.19, the time descrete deterministic-stochastic state-space description of the dynamic system is obtained :

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{p}_k + \mathbf{w}_k \tag{1.20}$$

$$\mathbf{d}_k = \mathbf{G}\mathbf{x}_k + \mathbf{J}\mathbf{p}_k + \mathbf{v}_k \tag{1.21}$$

The Kalman Filter algorithm consists in estimating the state $\mathbf{x_k}$ from a set of measures $\mathbf{d_k}$. A state estimator $\hat{\mathbf{x}}_{[k|l]}$ is defined as an estimate of $\mathbf{x_k}$ for a sequence of outputs $\mathbf{d}_{[1,\dots,l]}$. In order to estimate the process error, the error covariance matrix is defined as :

$$\mathbf{P}_{[k|l]} = \mathbb{E}\left[(\mathbf{x_k} - \hat{\mathbf{x}}_{[k|l]})(\mathbf{x}_k - \hat{\mathbf{x}}_{[k|l]})^T\right]. \tag{1.22}$$

The initial state $\hat{\mathbf{x}}_{[0|-1]}$ estimate and the initial error covariance matrix $\mathbf{P}_{[0|-1]}$ are both assumed known. The Kalman Filter algorithm compute the Kalman Gain $\mathbf{K}$ such as the state estimates have the minimum variance and are unbiased. This gain act as a weight that put more importance either on the measure or on the model at each time step. Indeed the Kalman Filter algorithm propagates by computing state estimates $\hat{\mathbf{x}}_{[k+1|k]}$ as follow:

$$\hat{\mathbf{x}}_{[k+1|k]} = \mathbf{A}\hat{\mathbf{x}}_{[k|k-1]} + \mathbf{B}\mathbf{p}_k + \mathbf{K}_k\left(\mathbf{d}_k - \mathbf{G}\hat{\mathbf{x}}_{[k|k-1]} - \mathbf{J}\mathbf{p}_k\right) \tag{1.23}$$

The Kalman gain is computed for each step as follow:

$$\mathbf{K}_k = \mathbf{A}\mathbf{P}_{[k|k-1]}\mathbf{G}^T\left(\mathbf{G}\mathbf{P}_{[k|k-1]}\mathbf{G}^T + R\right)^{-1} \tag{1.24}$$

(Matrix R is defined below). And for each step, the error covariance matrix is updated in two steps, the first step can be seen as the measurement update and the second step as the time update:

$$\mathbf{P}_{[k|k]} = \mathbf{P}_{[k|k-1]} + \mathbf{A}^{-1}\mathbf{K}_k\mathbf{G}\mathbf{P}_{[k|k-1]} \tag{1.25}$$

$$\mathbf{P}_{[k+1|k]} = \mathbf{A}\mathbf{P}_{[k|k]}\mathbf{A}^T + \mathbf{Q} - \mathbf{K}_k\mathbf{S}^T - \mathbf{S}\mathbf{K}_k^T \tag{1.26}$$

Where $\mathbf{Q}$, $\mathbf{R}$ and $\mathbf{S}$ are known noise covariance matrices which can be computed as shown next.

1.2.1 Noise covariance matrices

The process noise vector $\mathbf{w}_k$ accounts for unknown stochastic excitation and modeling errors denoted by $\mathbf{w}_{Sk}$ and $\mathbf{w}_{Ek}$ respectively. The measurement noise accounts for unkown stochastic excitation, modeling errors and measurement errors denoted by $\mathbf{v}_{Sk}$, $\mathbf{v}_{Ek}$ and $\mathbf{v}_{Mk}$ respectively. $\mathbf{w}_{Sk}$ and $\mathbf{v}_{Sk}$ which represent the stochastic excitation can be expressed as:

$$\mathbf{w}_{Sk} = \mathbf{B}'\mathbf{p}_{Sk} \tag{1.27}$$

$$\mathbf{v}_{Sk} = \mathbf{J}'\mathbf{p}_{Sk} \tag{1.28}$$

With $\mathbf{p}_{Sk}$ the unknown vector of stochastic forces, $\mathbf{B}'$ relates the state vector to the vector of stochastic forces and $\mathbf{J}'$ relates the output vector to the vector of stochastic forces.

$\mathbf{w}_{Ek}$ and $\mathbf{v}_{Ek}$ which represent the modeling error can be expressed as:

$$\mathbf{w}_{Ek} = \mathbf{\Delta A}\mathbf{x}_k + \mathbf{\Delta B}\mathbf{p}_k \tag{1.29}$$

$$\mathbf{v}_{Ek} = \mathbf{\Delta G}\mathbf{x}_k + \mathbf{\Delta J}\mathbf{p}_k \tag{1.30}$$

With $\mathbf{\Delta A}$, $\mathbf{\Delta B}$, $\mathbf{\Delta G}$ and $\mathbf{\Delta J}$ the error on $\mathbf{A}$, $\mathbf{B}$, $\mathbf{G}$ and $\mathbf{J}$ matrix resulting from error on stiffness, damping and mass matrices. Determination of these matrices will be subject to a further analysis in section 1.2.2.

The noise processes $\mathbf{w}_k$ and $\mathbf{v}_k$ are assumed to be zero mean and white with known covariance matrices $\mathbf{Q}$, $\mathbf{R}$ and $\mathbf{S}$ defined by:

$$\mathbb{E}\left[\begin{pmatrix}\mathbf{w}_k\\ \mathbf{v}_k\end{pmatrix}\quad\begin{pmatrix}\mathbf{w}_l^T & \mathbf{v}_l^T\end{pmatrix}\right] = \begin{bmatrix}\mathbf{Q} & \mathbf{S}\\ \mathbf{S}^T & \mathbf{R}\end{bmatrix}\delta_{[k-l]} \tag{1.31}$$

The assumptions are that stochastic forces represented by the vector $\mathbf{p}_S$ and the state vector $\mathbf{x}$ are uncorrolated, as well as $\mathbf{p}_S$ and the force history vector $\mathbf{p}$. It is also assumed that $\mathbf{p}_S$, $\mathbf{x}$, and $\mathbf{p}$ are zero mean, this is relevant in the context of a dynamic analysis.

$$\mathbf{Q} = \mathbb{E}(\mathbf{w}_k\mathbf{w}_l^T) = \mathbf{B}'\mathbf{C_{pS}}\mathbf{B}'^T + \mathbf{\Delta A}\mathbf{C_x}\mathbf{\Delta A}^T + \mathbf{\Delta B}\mathbf{C_p}\mathbf{\Delta B}^T + \mathbf{\Delta A}\mathbf{C_{xp}}\mathbf{\Delta B}^T + \mathbf{\Delta B}\mathbf{C_{px}}\mathbf{\Delta A}^T \tag{1.32}$$

With $\mathbf{C_{pS}} \in \mathbb{R}^{n_{ps}\times n_{ps}}$ the covariance matrix of the stocastic forces, $\mathbf{C_x} \in \mathbb{R}^{n_s\times n_s}$ the covariance matrix of the state vector, $\mathbf{C_p} \in \mathbb{R}^{n_p\times n_p}$ the covariance matrix of the force, $\mathbf{C_{px}} = \mathbb{E}(\mathbf{p}_k\mathbf{x}_l^T) = COV(\mathbf{p}_k,\mathbf{x}_l^T) \in \mathbb{R}^{n_p\times n_s}$ and $\mathbf{C_{xp}} = \mathbb{E}(\mathbf{x}_k\mathbf{p}_l^T) = COV(\mathbf{x}_k,\mathbf{p}_l^T) \in \mathbb{R}^{n_s\times n_p}$. Where n_{ps} is the number of stochastic forces acting on the system.

$$\mathbf{S} = \mathbb{E}(\mathbf{w}_k\mathbf{v}_l^T) = \mathbf{B}'\mathbf{C_{pS}}\mathbf{J}'^T + \mathbf{\Delta A}\mathbf{C_x}\mathbf{\Delta G}^T + \mathbf{\Delta B}\mathbf{C_p}\mathbf{\Delta J}^T + \mathbf{\Delta A}\mathbf{C_{xp}}\mathbf{\Delta J}^T + +\mathbf{\Delta B}\mathbf{C_{px}}\mathbf{\Delta G}^T \tag{1.33}$$

$$\mathbf{R} = \mathbb{E}(\mathbf{v}_k\mathbf{w}_l^T) = \mathbf{J}'\mathbf{C_{pS}}\mathbf{J}'^T + \mathbf{\Delta G}\mathbf{C_x}\mathbf{\Delta G}^T + \mathbf{\Delta J}\mathbf{C_p}\mathbf{\Delta J}^T + \mathbf{\Delta G}\mathbf{C_{xp}}\mathbf{\Delta J}^T + +\mathbf{\Delta J}\mathbf{C_{px}}\mathbf{\Delta G}^T + \mathbf{R_M} \tag{1.34}$$

With $\mathbf{R_M} \in \mathbb{R}^{n_d\times n_d}$ the measurement error covariance matrix. $\mathbf{C_{pS}}$, $\mathbf{C_p}$, $\mathbf{C_x}$, $\mathbf{C_{px}}$, $\mathbf{C_{xp}}$ and $\mathbf{R_M}$ act as tuning parameters for the system and depend on the sensors quality, the size of the system and type of input excitation force.
One can note that if modeling uncertainties are neglected, the expression for error covariance matrix given in [12] is found.

1.2.2 Modeling uncertainties

The aim of this section is to put forward a solution to compute matrices $\mathbf{\Delta A}$, $\mathbf{\Delta B}$, $\mathbf{\Delta G}$ and $\mathbf{\Delta J}$ defined in equations 1.29 and 1.30.
These modeling error matrices are the result from a wrong estimation of the dynamical parameters of the system, *i.e.* mass matrix $\mathbf{M}$, stiffness matrix $\mathbf{K}$ and damping matrix $\mathbf{C}$. Thus the modeling errors matrices should be the result of an estimation of error on $\mathbf{M}$, $\mathbf{K}$ and $\mathbf{C}$.

In practice, errors on $\mathbf{K}$, $\mathbf{C}$ and $\mathbf{M}$ matrices can be spotted from errors on natural frequencies ω_j and on damping ratio ξ_j. To do so, natural frequencies and damping ratios for every mode has to be identified from the measured output vector only, without any information on the excitation force. This process is called the omerational modal analysis (OMA) and has been well studied in case of unknown stochastic excitation. Several technics like Peak-picking technique [14] or Natural excitation technique [4] enable the user to find the modal parameter of a system from an output meaured signal, either displacement, velocity or acceleration.
In particular the peak-picking technique uses a frequency analysis and identify peaks corresponding to the natural frequencies. This technique is a SDOF technique but it can be applied on MDOF systems through a singular value decomposition (SDV) of the matrix of power spectra, *i.e.* each mode of the MDOF system is seen as a SDOF system. This advanced version of peak-picking method is known as the Complex Mode Indicator Function (CMIF) [3].

All the OMA techniques presented above assume that the input applied on the structure is a stationary white noise. In a windturbine, the input force applied on the monopile is composed of stochastic excitations which can be considered as white noise due to wind turbulences, but also from an harmonic part due to blade rotation and tower shadow effect. In this case, the OMA methods are not applicable. However, the response to a harmonic excitation can be seen as a virtual eigenmode with zero damping and it can be considered that only white noise excitation is present. This trick makes the OMA possible in presence of harmonic excitation but in order to find the good modal properties of the system, the real eigenfrequencies have to be distinguished from the pseudo eigenfrequencies due to harmonic excitation.
A solution based on peak-picking technique and on statistiacal analysis of the output is presented in [1]. Indeed, the probability density function of a harmonic response is a distribution with two peaks whereas the probability density function of a stochastic structural response is only one peak.

Probability density function of a harmonic response

Let X and Y be two random variables with corresponding probability density functions $f(x)$ and $g(y)$ related by $y = h(x)$. Then according to the fundamental theorem in [17][1]:

$$g(y) = \frac{f(x)}{h'(x)} \tag{1.35}$$

[1]Pages 93-94 of third edition (1991)

Now, for a harmonic response of amplitude a, $y = a\sin(x+\theta)$ or $x = \arcsin\left(\frac{y}{a}\right) - \theta$.
Since $h'(x) = a\cos(x+\theta) = a\cos\left(arcsin(\left(\frac{y}{a}\right)\right) = \sqrt{a^2-y^2}$:

$$g(y) = \frac{f(x)}{\sqrt{a^2-y^2}} \tag{1.36}$$

Taking the density function of X as uniform in the interval $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$[2]:

$$g(y) = \frac{1}{\pi\sqrt{a^2-y^2}} \tag{1.37}$$

The density function above goes to infinity if $y = a$ or if $y = -a$ which means that the density function has two distinct peaks.

Probability density function of a stochastic response

In case of stochastic loading, the structure is loaded by a large number of independant loading sources which can be seen as a large number of stochastic variables. The central limit theorem states that any linear combination of a large number of stochastic variables tends to a normal distribution. Thus, the probability density function of a stochastic response will show only one peak as a gaussian distribution.

After computing the modal response for each mode, *i.e.* for each peak detected in the frequency domain decomposition of the signal, R. Brincker and al. in [1] are able to distinguish the type of mode corresponding to each peak. In case of a harmonic mode, no further information will be taken from the response, but in case of structural mode, the eigenfrequency and the damping ratio will be identified thanks to a classic SDOF peak-picking method.
In reality, this method based on statistiacal analysis of the output is consistent only if the harmonic frequencies are well separated from the structural eigen frequencies. In the case of a windturbine, special care is given to avoid resonance in the design process. Hence, such an assuption is quite realistic for a windturbine.

Once the eigen frequencies and the damping ratios are identified from the measured output signal, it can be compared to the parameters of the model and thus identify and quantify the modeling error.

1.2.3 Joint input-state estimation algorithm

In this most basic form of the Kalman filter, it is assumed that the excitation forces $\mathbf{p}_k$ are known. This assumption may be relevant for systems where the known excitation forces are significantly higher than the stochastic forces applyed on the system but as our goal is to apply this Kalman Filter on a wind turbine monopile, this assumption is not acceptable. Indeed, the wind may be

[2]The equation $y = a\sin(x+\theta)$ has one root in the interval $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$ for any θ

very turbulent, even more in front of an obstacle like a wind turbine. It is of course possible to have a mean value of the force applied on the structure thanks to anemometers but the actual wind acting on the wind turbine can not be known exactly. Moreover, on offshore wind turbines the waves add a significant stochastic excitation to the whole structure. Thus, it is necessary to consider the force as a unknown variable that is to be estimated.

The original algorithm called the joint input-state estimation algorithm is found in [9] but contrary to the assumption made in it, the noise vectors $\mathbf{w}_k$ and $\mathbf{v}_k$ are correlated as it is done in [11] because they both account for excitation forces and modeling errors.

As done in section 1.2, a state estimate $\hat{\mathbf{x}}_{[k|l]}$ is defined as an estimate of $\mathbf{x}_k$. In addition to the basic KF algorithm, an input estimate $\hat{\mathbf{p}}_{[k|l]}$ is defined as an estimate of $\mathbf{p}_k$.

The error covariance matrix $\mathbf{P}_{[k|l]}$ is defined for both state and input and are denoted by $\mathbf{P_x}$ and $\mathbf{P_p}$ respectively. $\mathbf{P_x}$ is defined in 1.22 and $\mathbf{P_p}$ is defined similarly:

$$\mathbf{P}_{\mathbf{p}[k|l]} = \mathbb{E}\left[(\mathbf{p}_k - \hat{\mathbf{p}}_{[k|l]})(\mathbf{p}_k^T - \hat{\mathbf{p}}_{[k|l]}^T)\right] \tag{1.38}$$

Input and state are obviously correlated signals, thus a cross covariance matrix $\mathbf{P_{xp}}$ has to be defined:

$$\mathbf{P}_{\mathbf{xp}[k|l]} = \mathbf{P}_{\mathbf{px}[k|l]}^T = \mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_{[k|l]})(\mathbf{p}_k^T - \hat{\mathbf{p}}_{[k|l]}^T)\right] \tag{1.39}$$

In addition, a vector of output quantities $\mathbf{d_e}(t)$ that are to be identified - which is a combination of acceleration, velocity and displacement of the system - is introduced as:

$$\mathbf{d_e}(t) = \mathbf{S_{de,a}}\ddot{u}(t) + \mathbf{S_{de,v}}\dot{u}(t) + \mathbf{S_{de,d}}u(t) \tag{1.40}$$

Where $\mathbf{S_{de,a}}$, $\mathbf{S_{de,v}}$ and $\mathbf{S_{de,d}}$ are the selection matrices, similar to equation 1.7 except that $\mathbf{d_e}(t)$ represents the output quantities that are to be identified and not the output quantities measured. Similarly to what is done in section 1.1.2, equation 1.40 can be transform into its state-space form. After adding a measurement noise to this output vector, it is expressed as:

$$\mathbf{d}_{\mathbf{e}k} = \mathbf{G_e}\mathbf{x}_k + \mathbf{J_e}\mathbf{p}_k + \mathbf{v}_{\mathbf{e}k} \tag{1.41}$$

Matrices $\mathbf{G_e}$ and $\mathbf{J_e}$ are obtained from equations 1.9 and 1.10 by replacing selection matrices of output measured quantities $\mathbf{S_a}$, $\mathbf{S_v}$ and $\mathbf{S_d}$ by the selection matrices of output quantities to be identified $\mathbf{S_{de,a}}$, $\mathbf{S_{de,v}}$ and $\mathbf{S_{de,d}}$. The measurement noise vector $\mathbf{v}_{\mathbf{e}k}$ accounts for stochastic excitation and modeling errors and is assumed to be zero mean and white with known covariance matrices $\mathbf{R_e}$ and $\mathbf{R_c}$:

$$\mathbb{E}\left[\mathbf{v}_{\mathbf{e}k}\mathbf{v}_{\mathbf{e}l}^T\right] = \mathbf{R_e}\delta_{[k-l]}, \text{ and } \mathbb{E}\left[\mathbf{v}_{\mathbf{e}k}\mathbf{v}_l^T\right] = \mathbf{R_c}\delta_{[k-l]} \tag{1.42}$$

Finally, an output vector estimate $\hat{\mathbf{d}}_\mathbf{e}$ is defined and the error covariance matrix corresponding to

output estimate is defined similarly to other error covariance matrices :

$$\mathbf{P}_{\mathbf{de}[k|l]} = \mathbb{E}\left[(\mathbf{d}_{\mathbf{e}k} - \hat{\mathbf{d}}_{\mathbf{e}[k|l]})(\mathbf{d}_{\mathbf{e}k}^T - \hat{\mathbf{d}}_{\mathbf{e}[k|l]}^T)\right] \tag{1.43}$$

Joint input-state estimation algorithm

The algorithm presented below is taken from [11]. The initial state estimate vector $\hat{\mathbf{x}}_{[0|-1]}$ and its error covariance matrix $\mathbf{P}_{\mathbf{x}[0|-1]}$ are assumed to be known. The algorithm proceeds by computing the input, the state estimate, and the output estimate recursively in four steps, the input estimation step, the measurement update, the time update and the output estimation step:

Step 1: Input estimation

$$\tilde{\mathbf{R}}_k = \mathbf{G}\mathbf{P}_{\mathbf{x}[k|k-1]}\mathbf{G}^T + \mathbf{R} \tag{1.44}$$

$$\mathbf{M}_k = \left(\mathbf{J}^T\tilde{\mathbf{R}}_k^{-1}\mathbf{J}\right)^{-1}\mathbf{J}^T\tilde{\mathbf{R}}_k^{-1} \tag{1.45}$$

$$\hat{\mathbf{p}}_{[k|k]} = \mathbf{M}_k\left(\mathbf{d}_k - \mathbf{G}\hat{\mathbf{x}}_{[k|k-1]}\right) \tag{1.46}$$

$$\mathbf{P}_{\mathbf{p}[k|k]} = \left(\mathbf{J}^T\tilde{\mathbf{R}}_k^{-1}\mathbf{J}\right)^{-1} \tag{1.47}$$

Step 2: Measurement update

$$\mathbf{K}_k = \mathbf{P}_{\mathbf{x}[k|k-1]}\mathbf{G}^T\tilde{\mathbf{R}}_k^{-1} \tag{1.48}$$

$$\hat{\mathbf{x}}_{[k|k]} = \hat{\mathbf{x}}_{[k|k-1]} + \mathbf{K}_k\left(\mathbf{d}_k - \mathbf{G}\hat{\mathbf{x}}_{[k|k-1]} - \mathbf{J}\hat{\mathbf{p}}_{[k|k]}\right) \tag{1.49}$$

$$\mathbf{P}_{\mathbf{x}[k|k]} = \mathbf{P}_{\mathbf{x}[k|k-1]} - \mathbf{K}_k\left(\tilde{\mathbf{R}}_k - \mathbf{J}\mathbf{P}_{\mathbf{p}[k|k]}\mathbf{J}^T\right)\mathbf{K}_k \tag{1.50}$$

$$\mathbf{P}_{\mathbf{xp}[k|k]} = \mathbf{P}_{\mathbf{px}[k|k]}^T = -\mathbf{K}_k\mathbf{J}\mathbf{P}_{\mathbf{p}[k|k]} \tag{1.51}$$

Step 3: Time update

$$\hat{\mathbf{x}}_{[k+1|k]} = \mathbf{A}\hat{\mathbf{x}}_{[k|k]} + \mathbf{B}\hat{\mathbf{p}}_{[k|k]} \tag{1.52}$$

$$\mathbf{N}_k = \mathbf{A}\mathbf{K}_k\left(\mathbf{I}_{n_d} - \mathbf{J}\mathbf{M}_k\right) + \mathbf{B}\mathbf{M}_k \tag{1.53}$$

$$\mathbf{P}_{\mathbf{x}[k+1|k]} = \begin{bmatrix}\mathbf{A} & \mathbf{B}\end{bmatrix}\begin{bmatrix}\mathbf{P}_{\mathbf{x}[k|k]} & \mathbf{P}_{\mathbf{xp}[k|k]} \\ \mathbf{P}_{\mathbf{px}[k|k]} & \mathbf{P}_{\mathbf{p}[k|k]}\end{bmatrix}\begin{bmatrix}\mathbf{A}^T \\ \mathbf{B}^T\end{bmatrix} + \mathbf{Q} - \mathbf{N}_k\mathbf{S}^T - \mathbf{S}\mathbf{N}_k^T \tag{1.54}$$

Step 4: Output estimate

$$\mathbf{d}_{\mathbf{e}[k|k]} = \mathbf{G}_\mathbf{e}\hat{\mathbf{x}}_{[k|k]} + \mathbf{J}_\mathbf{e}\hat{\mathbf{p}}_{[k|k]} \tag{1.55}$$

$$\mathbf{Z}_k = \mathbf{G}_\mathbf{e}\mathbf{K}_k\left(\mathbf{I} - \mathbf{J}\mathbf{M}_k\right) + \mathbf{J}_\mathbf{e}\mathbf{M}_k \tag{1.56}$$

$$\mathbf{P}_{\mathbf{de}[k|k]} = \mathbf{G}_\mathbf{e}\mathbf{P}_{\mathbf{x}[k|k]}{\mathbf{G}_\mathbf{e}}^T + \mathbf{J}_\mathbf{e}\mathbf{P}_{\mathbf{p}[k|k]}{\mathbf{J}_\mathbf{e}}^T + \mathbf{G}_\mathbf{e}\mathbf{P}_{\mathbf{xp}[k|k]}{\mathbf{J}_\mathbf{e}}^T + \mathbf{J}_\mathbf{e}\mathbf{P}_{\mathbf{px}[k|k]}{\mathbf{G}_\mathbf{e}}^T + \mathbf{R}_\mathbf{e} - \mathbf{Z}_k{\mathbf{R}_\mathbf{c}}^T - \mathbf{R}_\mathbf{c}\mathbf{Z}_k^T \tag{1.57}$$

$\mathbf{M}_k$ and $\mathbf{K}_k$ are the gain matrix for input estimation and state estimation respectively. They are determined such as the input estimate $\hat{\mathbf{p}}_k$ and the state estimate $\hat{\mathbf{x}}_k$ are minimum variance and

unbiased.
The ucertainties on force and state estimate are quantified by the trace of error covariance matrices, $\mathbf{P}_{\mathbf{p}[k|k]}$ and $\mathbf{P}_{\mathbf{x}[k|k]}$ respectively. Similarly, the uncertainty on output estimated quantity is quantified by the trace of output covariance matrix $\mathbf{P}_{\mathbf{d_e}[k|k]}$.

The MATLAB® script for this algorithm in the simple case of a SDOF system is presented in Appendix B.

1.2.4 Algorithm results

For reasons of system identifiablity presented in [10], measured quantities are subject to some constraints. As the algorithm is computing both input and state, the measurements have to satisfy criteria of obervability, controlability and direct inversion.
For a SDOF system, the criterion of direct inversion requires that the output vector contains at least one acceleration measurement. In addition, avoiding marginally stable transmission zeros (see section 1.3) requires that the output vector contains at least one displacement measurement.

In the first instance, some results of the algorithm with satisfied instentaneous system inversion conditions will be presented: a SDOF system, with a mass $m = 10kg$, a dashpot $c = 2kg/s$ and a stiffness $k = 1000N/m$ is considered. First, a very short and high force shaped like a peak with an amplitude of $100N$ is applied to the system. Then a sinusoidal excitation force of amplitude $F_0 = 20N$ and a pulsation $\omega_0 = 7rad/s$ which is most likely the type of load seen by a windturbine is applied on the system. The algorithm is run in abscence of stochastic forces and with insignificant measurement noise[3]. The initial state vector $\hat{\mathbf{x}}_{[0|-1]}$ and the initial error covariance matrix $\mathbf{P}_{\mathbf{x}[0|-1]}$ are both assigned to zero value. In order to simulate the measured quantities, the dynamical equation 1.1 of the system is numerically solved and some atificial white and zero mean noise is added to the solution.

Figures 1.1 and 1.2 shows that in abscence of stochastic excitation, modeling errors and with almost no measurement error, the algorithm is able to reconstruct almost perfectly the actual force applied on the system and the system state.
The "true displacement" is the numerical solution to the dynamic equation 1.1 applied to the SDOF system described above and the "true force" is the force implemented in the dynamical equation resolution.

The measurement - which is in this example the "true" output - is now disturbed with some gaussian noise. σ_M,$d = 10^{-3}m$ and σ_M,$a = 10^{-2}m/s^2$ are the standart deviation of measurement

[3]The standart deviation of measurement noise can not be set strictly to zero because we see in the joint input-state estimation algorithm that inverse of $\tilde{\mathbf{R}}$ has to be computed, which means that the matrix can not be singular. To avoid this problem, we set a standart deviation of 10^{-7} for the noise in displacement and acceleration measurement. This problem is not so important as the aim of the algorithm is, among other, to filter measurement noise.

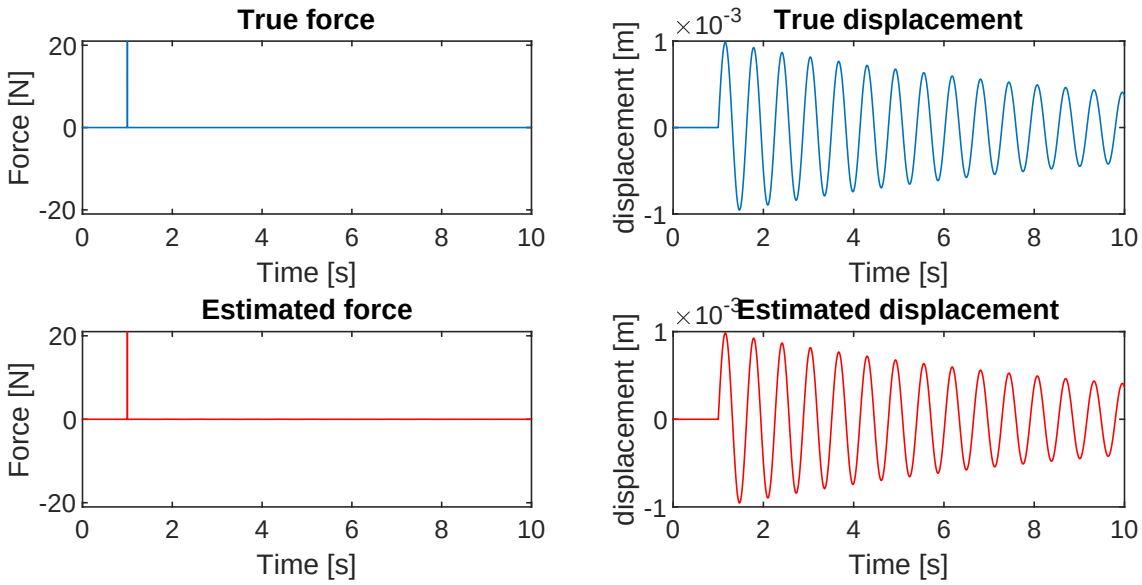


Fig. 1.1 : Response to peak excitation

noise for the displacement and the acceleration respectively. A zero mean and white stochastic force acting on the system with a standart deviation $\sigma_{p_s} = 2N$ is also considered. Finally for this example as well, the initial state vector $\hat{\mathbf{x}}_{[0|-1]}$ and the initial error covariance matrix $\mathbf{P}_{\mathbf{x}[0|-1]}$ are both assigned to zero value. Figure 1.3 shows the capacity to remove the noise from the measured output. For this example, the filter shows very satisfying results in terms of strain estimation. The uncertaity on output estimate, $tr(\mathbf{P}_{\mathbf{d}_e}) = 2.8 \times 10^{-8}\ m^2$ is almost insignificant. On another hand, the force estimate is less accurate, the uncertainty on the force estimate, $tr(\mathbf{P}_{\mathbf{p}}) = 4.04\ N^2$. This result correspond to a stochastic force with a standart deviation of 2N. Besides, force estimation can be important for operating but for fatigue calculation, only strain estimation which are in this case very accurate are important.

Hence, for a SDOF system the algorithm is very performant when modeling errors are assumed negligible. In reality, it is very unlikely that the complex dynamic model used for the wind turbine monopile is perfect. Indeed, even if values of stiffness, mass and damping are accurately measured, the finite element process will introduce some error, especially as the main deformation on the monopile is bending. Thus, the response of the algorithm in attendance of modeling errors is analysed below.

In presence of modeling errors, the system described by equations 1.20 and 1.21 does not represent the true dynamic behavior of the structure anymore. The force and the state obtained from the joint input-state estimation algorithm are no longer minimum variance and unbiased. The force and state covariance matrices do not correspond to the true error on the biased estimates. Similarly, in the presence of modelling errors, the estimated output covariance matrix $\mathbf{P}_{\mathbf{d}_e[k|k]}$ does not correspond to the true error on the biased output estimate. Hence it is not possible to use the

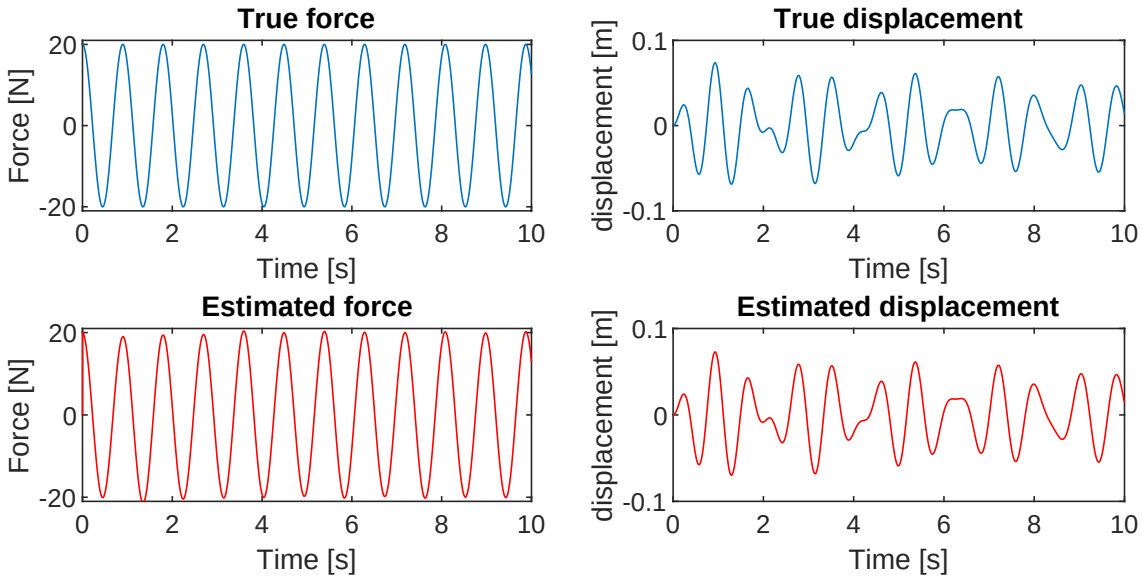


Fig. 1.2 : Response to sinusoidal excitation

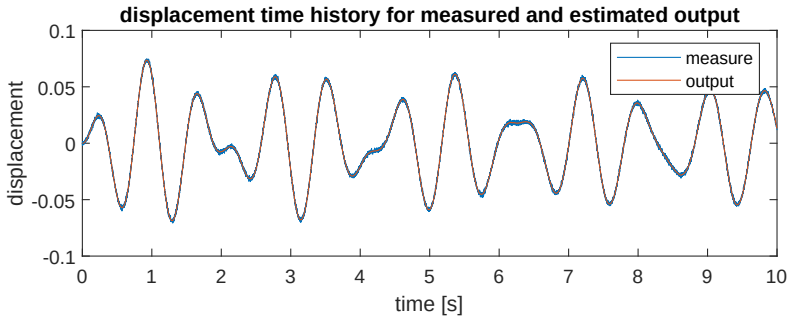


Fig. 1.3 : Response to sinusoidal excitation with measurement noise and stochastic forces

error covariance matrix to estimate the error on input and output estimate.
However, the advantage of virtual sensing is that every parameter is controllable, such as measurement noise, standart deviation of stochastic forces and modeling error matrices $\mathbf{\Delta A}$ and $\mathbf{\Delta B}$. Thereupon, it is possible to see how the algorithm reacts to modeling errors by comparing estimated output with and without modeling error. It is important to note that the modeling error analysis can not be decoupled from the other sources of error. Indeed, if the measurement noise and stochastic forces are set to zero, it means that the measured output quantity is the exact solution and the algorithm will rely one hundred percent on the measurement and the imformation given by the model will be totally useless. The process to quantify modeling error is described below :

- A reference output estimate with no modeling error is computed with a set of parameters that describes the stochastic forces and the measurement noise.
- While keeping the same measurement noise[4], the joint input-state estimation algorithm is

[4]The measurement noise is made randomly everytime that the algorithm is run. To be sure that the modeling

run with modeling errors on the stiffness k, the daming c and the mass m.

- The standart deviation of the difference between the two signals σ_{model} is used to quantify the modeling error on the output estimate.

The set of parameters for stochastic forces and measurement noise is the same than on the previous examples, *i.e.* σ_M,$d = 10^{-3}m$ and σ_M,$a = 10^{-2}m/s^2$ the standart deviation of measurement noise for the displacement and the acceleration respectively, with a stochastic force of standart deviation $\sigma_{p_s} = 2N$ applyed on the system. The results for different values of error on mass and on stiffness for a damping error $err_c = 0\%$ are presented in table 1.1.

$\sigma_{model} \times 10^{-3}$ $[m]$	1%	5%	10%	20%	50%	80%
1%	0.3	1.6	2.7	3.8	4.5	4.7
5%	1.8	2.7	3.4	4.0	4.6	4.7
10%	3.1	3.5	3.9	4.2	4.6	4.8
20%	4.1	4.2	4.4	4.5	4.7	4.8
50%	4.8	4.8	4.8	4.9	4.9	4.9
80%	5.0	5.0	5.0	5.0	5.0	5.0

Table 1.1: Values of $\sigma_{model}[m]$ for different relative error on mass (rows) and stiffness (columns) with no damping error

The results for different values of error on mass and on stiffness for a daming error $err_c = 20\%$ are presented in table 1.2.

$\sigma_{model} \times 10^{-3}$ $[m]$	1%	5%	10%	20%	50%	80%
1%	0.8	1.8	2.8	3.8	4.5	4.7
5%	2.0	2.8	3.4	4.0	4.6	4.7
10%	3.2	3.5	3.9	4.2	4.6	4.8
20%	4.1	4.2	4.4	4.5	4.7	4.8
50%	4.8	4.8	4.8	4.9	4.9	4.9
80%	5.0	5.0	5.0	5.0	5.0	5.0

Table 1.2: Values of σ_{model} for different relative error on mass (rows) and stiffness (columns) for a relative damping error of 20%

Different conclusions can be drawn from this tables. First, the algorithm stays quite stable for small modeling errors, indeed, the range of σ_{model} is always 10^{-3} which is equal to the standart deviation of the noise in the displacement measurement, there is no unstable phenomenon with

error is observed without any other measurement noise, the measure has to stay exactely the same when the reference output estimate is computed and when the response with modeling error is computed.

the modeling errors concerning the output estimation.

Then, we can rank in order of importance in the algorithm the three dynamical parameters mass m, damping c and stiffness k. The difference between m and k is slight in the results presented but as the mass m is involved more significantely[5] than stiffness k in force estimation, the mass is stated as the most important dynamical parameter that is to be measured as accuratly as possible. Then comes the stiffness k, to which the output estimate is also sensitive and finally, the damping c has less effect on the modeling error, hence it is not necessary to use resources to measure the damping c very acurately.

Last but not least, the modeling error tends to a limit which seems to be 5.0×10^{-3} for the set of parameters considered in this example. This is indeed due to the fact that the joint input-state estimation algorithm is based on a principle of weight mean between the measure and the model. Thus, once the model starts to be very unacurate, all the weight is put on the measure. The output is then not absurd but if the modeling errors are too high, the key concept of filtering is lost.

Concerning the input estimate, the algorithm does not behaves as well as it does with the output estimate. The process presented above to compute σ_{model} is done once again to compute $\sigma_{input_{mod}}$, which is the uncertainty induced by modeling error on the input estimation. Results are presented in table 1.3

$\sigma_{input_{mod}}$ $[N]$	1%	5%	10%	20%	50%	80%
1%	0.4237	0.8178	1.4578	3.5794	12.6907	22.5118
5%	1.2614	1.7721	2.6486	5.1496	14.4666	24.3161
10%	2.4979	3.2481	4.4205	7.2140	16.7032	26.5793
20%	5.9303	7.0255	8.4759	11.5368	21.2168	31.1266
50%	18.9496	20.2274	21.8352	25.0772	34.9273	44.8676
80%	32.6549	33.9559	35.5862	38.8582	48.7376	58.6744

Table 1.3: Values of $\sigma_{input_{mod}}$ for different relative error on mass (rows) and stiffness (columns) for a relative damping error of 0%

The increase of model errors has dire consequences on the input estimate. Contrary to the output estimate, the joint input-state estimation algorithm does not include any force measurement that can prevent the model error from becoming very large. Indeed, when the algorithm include a measurement of the quantity that is to be estimated, in case of high modeling error, the weighted average process can put all the weight on the measure and not on the model. However, when the algorithm does not include any measurement of the quantity that is to be estimated, both the modeling term and the measurement term that acts in the weighted average process are biased. One can see in figure 1.4 that with 10% relative error on mass and stiffness, the shape of the estimated force is totally different from the true force.

[5]The gain matrix $\mathbf{M}_k$ used for input estimation is computed with matrix J which contains the mass m but not the stiffness k.

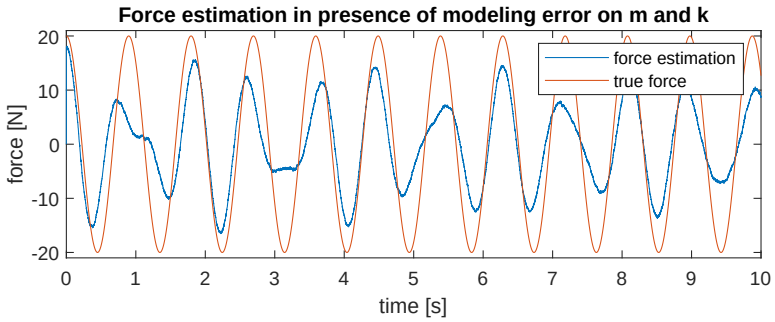


Fig. 1.4 : Force estimation in presence of 10% relative error on mass, 10% relative error on stiffness and 0% relative error on damping

The results on model error in force estimation confirm that the algorithm is more sensitive to error on mass m than on stiffness k.

In order to keep a good filtering performance, the input has to be computed accurately as it is used in the measurement update for the state. Then a criteria on $\sigma_{input_{mod}}$ can be set: the standart deviation of the uncertainty introduced by modeling errors $\sigma_{input_{mod}}$ has to be at least 10 times lower than the standart deviation of the stochastic forces applied on the structure. This criterion ensures that the stochastic forces remain predominant in the process of force estimation.

1.3 Stability of filtering process

It has been seen that in the example of SDOF system, when the output measured vector contains acceleration and displacement and when the ouptut vector to be estimated contains only displacement, the algorithm is stable against measurement errors, stochastic forces and modeling errors. In this part, stability criteria will be analysed deeper and not only for SDOF systems. This part is not a personal contribution, as this problem is studied in [10] but it is still explained there for sake of completeness.

1.3.1 Identifiability conditions

In the joint input-state estimation algorithm, the input and output quantities are estimated from an output measured vector and a model. Hence it is necessary that the measurement contains information on all the quantities that are to be estimated. The identifiability condition requires observability and direct inverstability.

Observability

System observability requires that all the states are observed in the system output. An observability matrix is defined:

$$\mathcal{O} = \begin{bmatrix} \mathbf{G} \\ \mathbf{GA} \\ \vdots \\ \mathbf{GA}^{n_s-1} \end{bmatrix} \tag{1.58}$$

The system described by equations 1.20 and 1.21 is observable if and only if $\text{rank}(\mathcal{O}) = n_s$ with n_s the number of system state.
This criteria is automatically satisfied in a SDOF system as long as $\mathbf{G}$ is non-null. For a multiple degree of freedom (MDOF) system, when a modally reduced order model is used, observability requires that all modes considered in the model contribute to the measured quantity.

Direct inversion

Direct invertibility requires that the input can be estimated from the output without any time delay. The necessary and sufficient condition for direct invertibility is, for a system described by equations 1.20 and 1.21, $\text{rank}(\mathbf{J}) = n_p$ with n_p the number of input forces that are to be estimated. For a SDOF system with only one force applying on it, the direct inversion is satisfied as long as $\mathbf{J}$ is non-null. According to 1.10, $\mathbf{S}_a$ has to be non-null which means that the output measure vector should contain at least one acceleration measurement. For a MDOF system, when a modally reduced order model is used, this direct inversion criteria requires two conditions:

- The number of input forces to be identified n_p has to be lower or equal to the number of modes used in the model n_m.
- The number of acceleration measurements $n_{d,a}$ has to be greater or equal to the number of input forces to be estimated n_p.

Proofs for the statements above can be found in [10] and [13].

Controlability

When unknown ambiant forces such as wind loads are acting on the structure, the location and the spatial distribution of these forces are not well known. In this case, the joint input-state estimation algorithm is applied to identify a set of forces acting at predefined locations, *i.e.* $\mathbf{S_p}$ is a parameter set by the user. Then the estimated forces are not the true force but equivalent forces that compensates for any source of vibration. The unknown vibration source generally excites all modes, so the estimated forces should be able to do the same. This requires that all the states of the system can be controlled by the system input.
The system controlability is tested by the controllability matrix $\mathcal{C}$:

$$\mathcal{C} = \begin{bmatrix} \mathbf{B} & \mathbf{AB} & \dots & \mathbf{A}^{n_s-1}\mathbf{B} \end{bmatrix} \tag{1.59}$$

The system described by equations 1.20 and 1.21 is controllable if and only if $\text{rank}(\mathcal{C}) = n_s$, with n_s the number of system states.

For a MDOF system and for a modally reduced order model, if all damped natural frequencies are different, controllability requires that the matrix $\mathbf{S_p\Phi}$ does not contain any zero columns.

These invertibility conditions are necessary but not sufficient to guarentee that the forces and the system states can be estimated in the presence of noise. The system also needs to satisfy stability and uniqueness conditions to estimate the input and output correctly.

1.3.2 Stability and uniqueness conditions

It is now assumed that the system is totally observable, controlable and that it satisfy the direct invertibility conditions. In this section, the properties and the rank of Rosenbrock's system matrix will be analyzed ([16],[19]):

$$\mathcal{R} = \begin{bmatrix} \mathbf{A} - \lambda_j \mathbf{I} & \mathbf{B} \\ \mathbf{G} & \mathbf{J} \end{bmatrix} \tag{1.60}$$

$\lambda_j \in \mathbb{C}$ is called a finite transmission zero of the system if

$$\text{rank}(\mathcal{R}) \leq n_s + min(n_p,n_d) \tag{1.61}$$

The transmission zero of a system depends on all four system matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{G}$ and $\mathbf{J}$ and are found by solving the generalized eigenvalue problem:

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{G} & \mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{[0]} \\ \mathbf{p}_{[0]} \end{bmatrix} = \begin{bmatrix} \lambda_j \mathbf{I} \\ \mathbf{0} \end{bmatrix} \tag{1.62}$$

When a modally reduced order model is used, the mode shapes Φ_j, the natural frequencies ω_j and the damping ratio ξ_j determine the location of the transmission zero in the complex plan. The location of the transmission zero also depends on the type, number and location of the sensors used for the measured output vector. If a transmission zero is located inside the unit circle, *i.e.* $|\lambda_j| < 1$, the transmission zero is called stable. If $|\lambda_j| = 1$, the transmission zero is called marginally stable. If $|\lambda_j| > 1$, the transmission zero is called unstable.

The following theorem is stated and proved in [10]:
Theorem: *If only acceleration and/or velocity measurements are included in the output vector, there will always be at least one purely real transmission zero $\lambda_j = 1$ marginally stable.*

Indeed, a transmission zero $\lambda_j = 1$ correspond to a constant excitation force. Both acceleration and velocity measurement are insensitive to constant excitation, hence the input can not be computed exactly with only velocity and acceleration measurement. Eventhough in dynamical studies the constant forces and displacement are not important, they can be responsible for system inversion instability.
To avoid the marginally stable transmission zero, it is stated in [10] that the number of displacement measurement has to be greater or equal to the number of forces n_p that are to be estimated.

The proposition on uniqueness is stated in [10] is presented below:

Proposition: *The input of a system with at least one finite transmission zero cannot be uniquely reconstructed.*

Indeed, if the system has a finite transmission zero, then by definition (equation 1.62) it exists an initial input and an initial state for which the system output is zero. As a consequence, the input cannot be uniquely reconstructed from the measured output.

To ilustrate the stability and uniqueness criteria, the joint input-state estimation algorithm is run with a measured output vector containing only acceleration. The system is the same than the one presented in section 1.2.4 but without displacement measurement. Matrices **G** and **J** and the error covariance matrices changes consequently. The estimated force and the estimated displacement are compared to the results in section 1.2.4 in figure 1.5

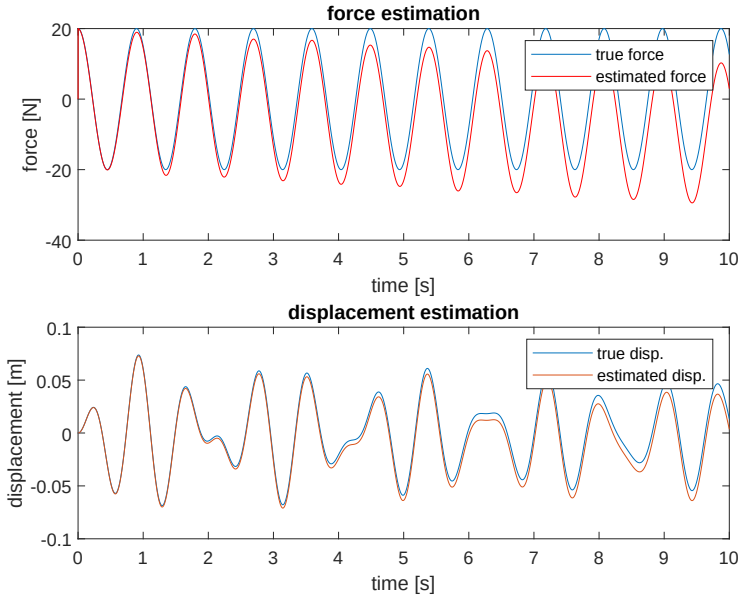


Fig. 1.5 : Response to sinusoidal excitation and estimation of this excitation force with only acceleration measurement

Both input and output estimations are shifting from the good solution. Then the force and the displacement are not computed perfectly. If the aim of the process is to calculate the fatigue damage to the structure, the important data are the charge/discharge cycles which can be identified for example thanks to a rainflow counting method. Three cycles parameters are factored: the range of the charge/discharge cycle, its mean value, and the number of cycles. The range and the number of cycles will be the same in both cases but the mean value will change.

However, as the shape of the signal is the same in both cases, it is feasible to redress the signal so that its mean value would be zero before proceeding with the rainflow counting process. In practice, accelerometers are very cheap compared to strain gauges and much easier to set up on a

wind turbine. Thus, estimating the signal without displacement measurement, and then proceed to an average adjustement may be a good economical solution for monitoring.

Chapter 2

Dynamic study of an offshore windturbine

2.1 Windturbine model

In this chapter, an offshore windturbine will be modeled by a tube-shaped beam clamped at one end and free at the other end, with a tip mass M_{tip} at the free end representing the nacelle and the rotor of the wind turbine. The beam will be submited to different kind of loads:

- The main load applied at the tip of the beam is caused by the lift component of the force applied on the blades. This force is the only force that is to be estimated with the joint input-state estimation algorithm.
- Light stochastic loads are applied all along the emerged part of the beam. This stochastic forces are caused by the wind on the structure, other than the blades.
- Slightly stronger stochastig loads are applied all along the immersed part of the beam in order to imitate the action of the stream and of the vortex induced vibrations (VIV) on the structure.
- Finally a stronger stochastic force is applied at the waterline level which accounts for VIV and for stronger forces due to the waves such as slaming loads in case of plunging waves.

The beam itself is modeled by an Euler-Bernoulli model in 2D. Then, the motion in only one dimension will be studied. The system is sketched in figure 2.1.

A modal analysis will be carried out on this beam. First, the eigenfrequencies and the mode shape have to be determined. The equilibrium equation of an Euler-Bernoulli beam can be written as [15]:

$$\rho A w_{,tt} + EI w_{,xxxx} = 0 \tag{2.1}$$

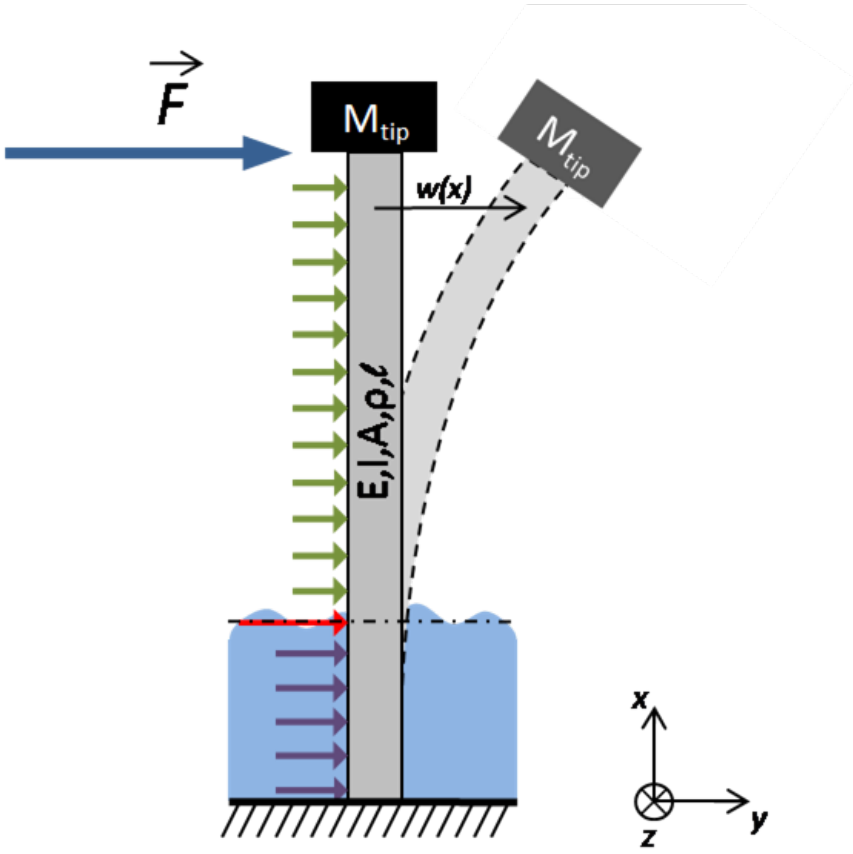


Fig. 2.1 : Sketch of the studied system - Arrows on the left side of the beam represents stochastic forces applying on the structure.

Where ρ is the volumic mass of the beam, A the surface of its cross section,l its length, E its Young modulus and I its quadratic moment.

Four boundary conditions are needed to describe the model correctly [8]:

$$w(0,t) = 0 \tag{2.2}$$

$$w_{,x}(0,t) = 0 \tag{2.3}$$

$$w_{,xx}(l,t) = 0 \tag{2.4}$$

$$EIw_{,xxx}(l,t) = M_{tip}w_{,t}t(l,t) \tag{2.5}$$

Equations 2.2 and 2.3 come from the fact that there is nore displacement neither rotation at the clamped end of the beam. Equation 2.4 expresses the abscence of bending moment at the free end of the beam. Equation 2.5 comes from the fundamental dynamic principle applied at the end of the beam which tells that the shear force is equal to the tip mass times the tip mass acceleration.

As vibration modes are studied, a solution of the following form will be searched:

$$w(x,t) = W(x)e^{i\omega t} \tag{2.6}$$

Subsituting equation 2.6 in equation 2.1:

$$-\omega^2 \rho A W(x) + EIW''''(x) = 0 \tag{2.7}$$

The boundary conditions can be expressed as:

$$W(0) = 0\ ,\ W'(0) = 0\ ,\ W''(l) = 0\ ,\ W'''(l) = -\frac{M_{tip}}{\rho A} W''''(l) \tag{2.8}$$

Equations 2.7 and 2.8 form the eigenvalue problem.

It is assumed that the solution of 2.7 can be written as:

$$W(x) = Be^{\tilde{\beta}x} \tag{2.9}$$

Substituting equation 2.9 in 2.7, and simplifying by $Be^{\tilde{\beta}x}$:

$$-\omega^2 \rho A + EI\tilde{\beta}^4 = 0 \tag{2.10}$$

2.10 has then four solution on $\tilde{\beta}$:

$$\tilde{\beta} = \pm \left[\frac{\omega^2 \rho A}{EI}\right]^{\frac{1}{4}} = \pm\beta \quad , \quad \tilde{\beta} = \pm\, i \left[\frac{\omega^2 \rho A}{EI}\right]^{\frac{1}{4}} = \pm\, i\beta \tag{2.11}$$

Thus $W(x)$ can be written as:

$$W(x) = B_1 \cosh(\beta x) + B_2 \sinh(\beta x) + B_3 \cos(\beta x) + B_4 \sin(\beta x) \tag{2.12}$$

Using 2.12 and the boundary conditions, it can be stated that:

$$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ \cosh(\beta l) & \sinh(\beta l) & -\cos(\beta l) & -\sin(\beta l) \\ \sinh(\beta l) + \mu\beta\cosh(\beta l) & \cosh(\beta l) + \mu\beta\sinh(\beta l) & \sin(\beta l) + \mu\beta\cos(\beta l) & \mu\beta\sin(\beta l) - \cos(\beta l) \end{bmatrix} \times \begin{bmatrix} B_1 \\ B_2 \\ B_3 \\ B_4 \end{bmatrix} = \overline{0} \tag{2.13}$$

With $\mu = \dfrac{M_{tip}}{\rho A}$

The equation 2.13 has to be true for non-trivial solutions, *i.e.* for $\begin{bmatrix} B_1 & B_2 & B_3 & B_4 \end{bmatrix}^T \neq \overline{0}$, which means that the determinant of the first matrix has to vanish:

$$\begin{vmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ \cosh(\beta l) & \sinh(\beta l) & -\cos(\beta l) & -\sin(\beta l) \\ \sinh(\beta l)+\mu\beta\cosh(\beta l) & \cosh(\beta l)+\mu\beta\sinh(\beta l) & \sin(\beta l)+\mu\beta\cos(\beta l) & \mu\beta\sin(\beta l)-\cos(\beta l) \end{vmatrix} = 0 \tag{2.14}$$

$$\Leftrightarrow \frac{1+\cos(\beta l)\cosh(\beta l)}{\beta l(\sin(\beta l)\cosh(\beta l)-\cos(\beta l)\sinh(\beta l)} = \frac{M_{tip}}{\rho A l} \tag{2.15}$$

Numerical methods are used to solve the characteristic equation 2.15 and obtain the n_m first natural frequencies of the system. The number of modes n_m is determined by the user.

Solving partially equation 2.13 for B_2, B_3 and B_4 gives :

$$B_2 = -B_1\frac{\cosh(\beta l)+\cos(\beta l)}{\sinh(\beta l)+\sin(\beta l)} \quad , \quad B_3 = -B_1 \quad , \quad B_4 = B_1\frac{\cosh(\beta l)+\cos(\beta l)}{\sinh(\beta l)+\sin(\beta l)} \tag{2.16}$$

These results are used in equation 2.12 and for each circular natural frequency β_n[1], the coresponding mode shape is numerically computed as:

$$W_n(x) = B_1\left[\left(\frac{\cosh(\beta_n l)+\cos(\beta_n l)}{\sinh(\beta_n l)+\sin(\beta_n l)}\right)(\sin(\beta_n x)-\sinh(\beta_n x))+\cosh(\beta_n x)-\cos(\beta_n x)\right] \tag{2.17}$$

The eigen frequencies ω_n are computed from equation 2.11 and the corresponding mode shapes Φ_n are computed from equation 2.17.

2.1.1 Mode shapes normalization

It is assumed in the system description in section 1.1.1 that the mode shape matrix $\mathbf{\Phi}$ satisfies the orthonomralization condition, *i.e.* $\mathbf{\Phi M \Phi^T} = \mathbf{I}$. Hence the value of B_1 is not important and can be arbitrary set e.g. to $B_1 = 1$.

For the normalization process only, the mass matrix $\mathbf{M}$ is defined using lumped mass and finite element method. The beam is divided in n_{disc} elements, each one with a local mass matrix

$$M^e = \frac{1}{2}\rho A l\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{2.18}$$

The global mass matrix is obtained by merging all the element matrices.

The eigenvectors are normalized by:

$$\Phi_n = \Phi_n/\sqrt{\Phi_n \mathbf{M} \Phi_n^T} \tag{2.19}$$

[1] solutions of equation 2.15

The four first eigen frequencies and their corresponding mode shapes are represented in figure 2.2. The parameters used for this simulation are: length $l = 107.6m$, volmic mass $\rho = 8500kg/m^3$, young modulus $E = 2.1 \times 10^{11} Pa$, Poisson ratio $\nu = 0.3$, diameter $D = 6m$ and thikness $t = 0.06m$. Matrix $\mathbf{\Phi}$ is obtained from the concatenation of all the mode shapes.

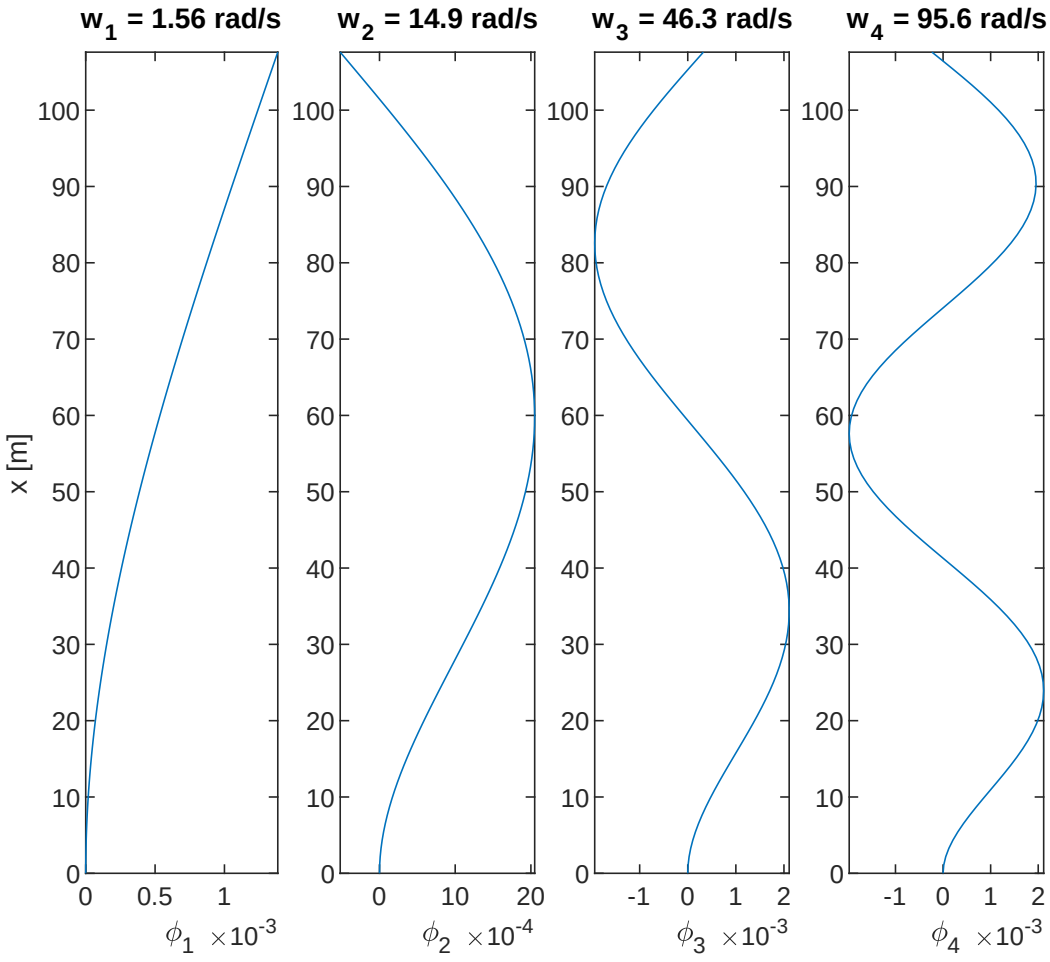


Fig. 2.2 : Four first eigen frequencies and their corresponding mode shapes for a set of dynamical parameters

2.1.2 Modal damping

As the assumption of proportional damping holds, $\mathbf{\Gamma}$ is a diagonal matrix containing the values of modal damping ratios for each mode. For a wind turbine, damping is the combination of two phenomena: the structural damping and the aerodynamic damping. Thus the damping ratios depend on the wind speed. However, as shown in section 1.2.4 the modeling error on damping values does not have so much impact on the overall estimation error. Hence, in this report, no further

attention will be paid to wind turbine damping.

Values for damping ratios for the three first modes are taken from [12] and come from measurements on a Vestas V90 3MW located in the belgium north sea with a height of 72 m above lowest astronomical tide, a water depth of 24 m and a diameter of 5 m.

$$\mathbf{\Gamma} = \begin{bmatrix} 2 \times 1.86\omega_1 & 0 & 0 \\ 0 & 2 \times 1.38\omega_2 & 0 \\ 0 & 0 & 2 \times 0.56\omega_3 \end{bmatrix} \% \tag{2.20}$$

2.1.3 Model reduction

Strictly speaking, the number of modes should be equal to the number of degrees of freedom of the system. However, only the low frequency modes have a physical meaning [18].

If the structure is excited by harmonic forces with a frequency within a bandwidth $\Delta\omega$, its response is dominated by the modes whose natural frequency belongs to the excitation bandwidth.

The structural response can be splited into two kinds of modes: the modes whose frequency belongs to the excitation bandwidth which are responding dynamically and the modes with higher frequency which are responding in a quasi-static manner.

In the context of dynamical study, the three first modes will be taken into account in the implementation of joint input-state estimation algorithm.

Finally system model matrices **A**, **B**, **J** and **G** are computed from equations 1.14, 1.15, 1.16 and 1.17.

2.2 Data aquisition

The data used in the joint input-state estimation algorithm will be taken from the finite element software specialized in offshore structures dynamic analysis FedEm®.

Different simulations of the joint input-state estimation algorithm will be carried out :

- First the algorithm response will be analyzed when the output measured vector contains one displacement measurement, one acceleration measurement and when only one excitation force is to be estimated. In this case the identifiability, stability and uniqueness conditions are fulfilled.

- Then the algorithm response will be analyzed when the uniqueness condition expressed in 1.3.2 is lacking, *i.e.* when the measured output vector does not contain any displacement or strain measurement.
  This case is particularely interesting as the accelerometers are cheap sensors and are easy to install on a structure, whereas the displacement sensors or strain gauges often have to be installed during the building process and are much more onerous.

- Finally, an optimization of the sensor layout will be put forward by comparing the output estimate error for different arrengements of accelerometers and displacement sensors.

A windturbine with a monopile which has the same dynamical properties than in 2.1 is implemented in FedEm®. The windturbine is loaded with a constant wind of $8m/s$. A sampling rate of $1250Hz$ is used in the data aquisition. In the system described above, only one force is to be estimated. Thus, the system identifiability, stability and uniqueness conditions require that the measured output vector contains at least one displacement and one acceleration measurement.

An example of measured output vector is shown in figures 2.3 and 2.4. The displacement measurement is taken at 97 m, and the acceleration measurement is taken 64 m above the mudline.

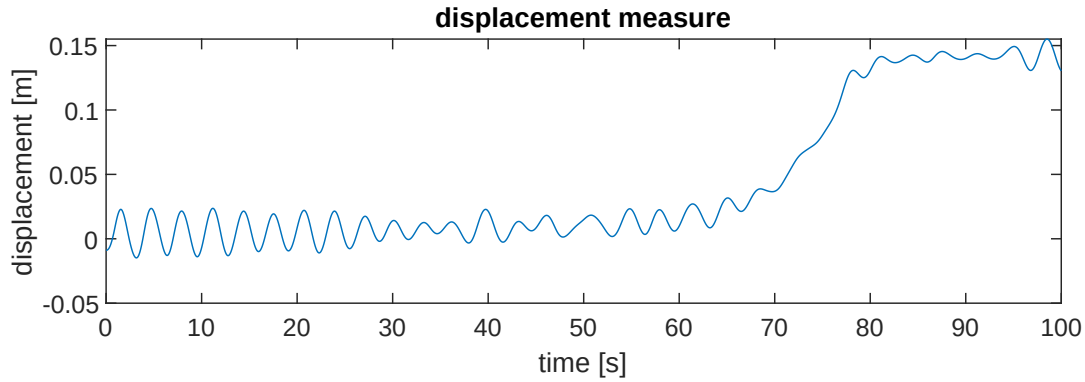


Fig. 2.3 : Displacement measure at altitude $z = 97\ m$

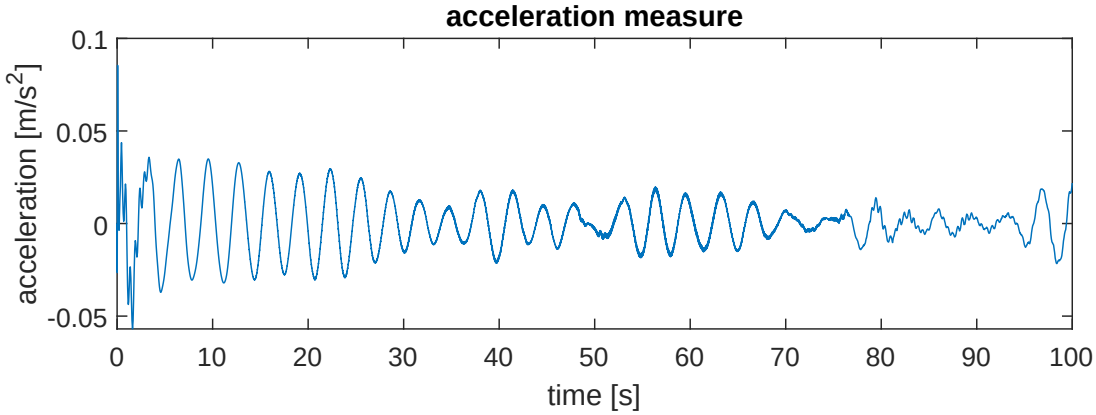


Fig. 2.4 : Acceleration measure at altitude $z = 64\ m$

2.3 Algorithm response analysis

2.3.1 Transfer functions of the system

The characteristics of the algorithm presented in section 1.2.3 depend on the dynamic behavior of the system and on the sensor layout. In order to analyse the impact of the sensors arrangement on the estimation error, the system transfer function relating Fourier transform of input to Fourier transform of output is computed.

The system transfer function H_{dp} is defined in the Laplace domain by :

$$H_{dp}(s) = \sum_{m=1}^{n_m} \frac{s^q \phi_{dm} \phi_{pm}}{s^2 + 2\xi_m \omega_m s + \omega_m^2} \tag{2.21}$$

With s the Laplace variable, ξ_m and ω_m are the modal damping ratio and the natural frequency corresponding to mode m. ϕ_{dm} and ϕ_{pm} are the components of mode shape corresponding to mode m taken respectively at the sensor(s) and at the force(s) location. The integer q equals 0 for displacement measurement, 1 for velocity and 2 for acceleration.
Replacing s by $i\omega$, the transfer function in frequency domain is obtained. Figures 2.5 respectively 2.6 show the transfer function obtained for displacement sensor and acceleration sensors respectively at 40 m above the mudline and at the tip of the beam.

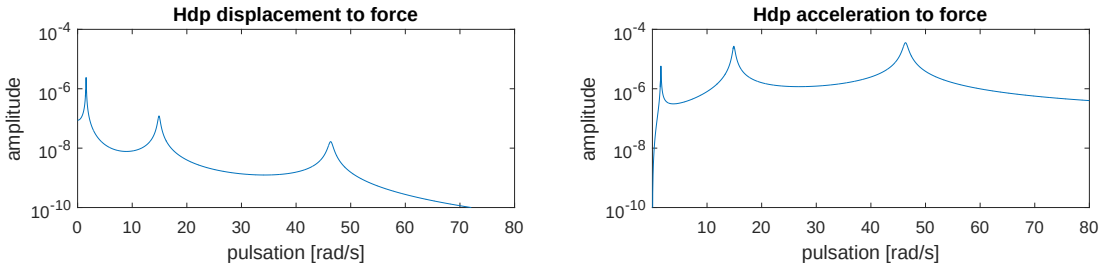


Fig. 2.5 : Transfer function relating Fourier transform of the displacement measure to Fourier transform of the force (left) and Fourier transform of the acceleration measure to Fourier transform of the force (right). Both acclerometer and displacement sensors are 40 m above the mudline

Three peaks corresponding to the eigenvalues can be identified for both sensor heights, however, the abscence of clear dips in figure 2.5 shows that existence of antiresonance frequencies depends on the sensor position on the monopile.

It has been shown in [11] that the estimation error depends on the zeros stability of the transfer function because the estimation error is maximum at the antiresonance frequencies. Hence, in case of existence of antiresonance frequencies, a special care will be given to unstable zero analysis in the optimization of the sensor layout in section 2.3.3.

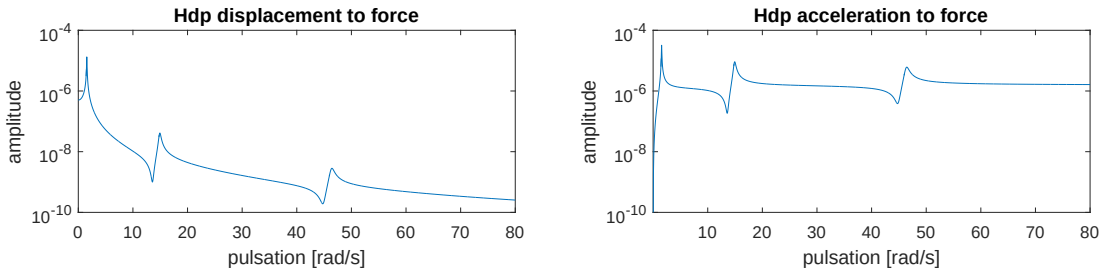


Fig. 2.6 : Transfer function relating Fourier transform of the displacement measure to Fourier transform of the force (left) and Fourier transform of the acceleration measure to Fourier transform of the force (right). Both acclerometer and displacement sensors are at the tip of the monopile.

Finally, the system transfer function is useful to check the discrete time system stability: the discrete time system is stable only if the poles are located inside the unit circle.
A system with n_m modes has $2n_m$ poles occuring in complex conjugate pairs given by:

$$\lambda_{m1,2} = -\omega_m \xi_m \pm i\omega_m \sqrt{1-\xi_m^2} \tag{2.22}$$

The poles of the transfer function $\lambda_{m1,2}$ of the continuous time system are related to the poles of the transfer function of the discrete time system $\lambda_{Dm1,2}$ as follows[2]:

$$\lambda_{Dm1,2} = \exp(\lambda_{m1,2}\Delta t) \tag{2.23}$$

The poles of the system depend only on its dynamical properties, *i.e.* its eigenvalues and the damping ratios.
The poles of the system presented in 2.1 are shown on the complex plan in figure 2.7.

The discrete time system is then stable and the joint input-state estimation can be performed. However, it is found that beyond a limit on the discretization time step Δt, system matrices - epecially $\mathbf{A}$ - are badly conditionned, which leads to important numerical error. For the system described in section 2.1, the maximum limit for time step is found to be 8×10^{-4} s.

2.3.2 Estimation results

The joint input-state estimation algorithm takes the model equations from section 2.1 and the data from section 2.2 and compute the input and output estimation through a weight average process. The weighting is based on the covariance matrices $\mathbf{Q}$, $\mathbf{S}$ and $\mathbf{R}$ defined in section 1.2.1.

The stochastic forces are assumed to be zero mean and white with a standart deviation of:

[2]Under zero-order hold assumption

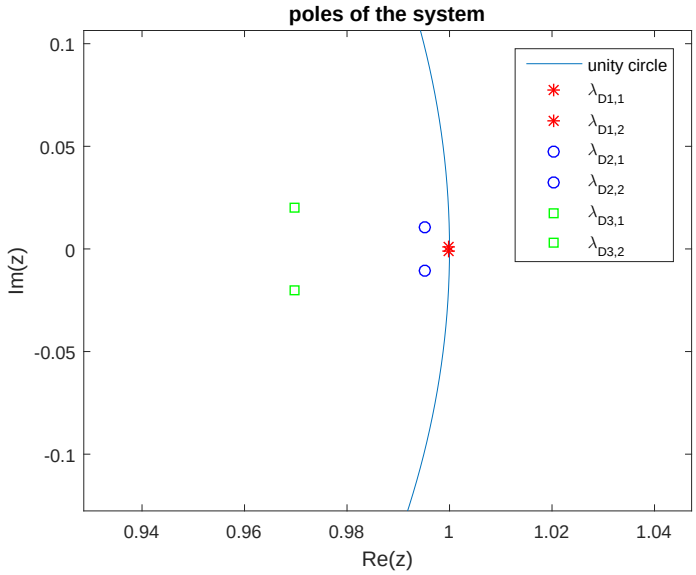


Fig. 2.7 : Poles of the system in complex plan compared to unity circle

- $100N$ for the forces located above the waterline
- $1000N$ for the forces located under the waterline
- $5000N$ for the force at the waterline ment to represent the eventual high loads induced by slamming waves.

The assumption of zero mean and white noise stochastic forces is, in practice, not very relevant as the waves, the wind and the underwater stream often have one dominant direction. However, in case of colored stochastic forces, the input and state estimates are no longer minimum variance and unbiased and the error covariance matrix of the extrapolated output vector $\mathbf{P_{de}}$ defined in equation 1.57 does not correspond to the true error on the output estimate. Algorithm response in case of non zero mean stochastic forces is beyond the scope of this thesis, however it would be a good track of research to make the algorithm results even more accurate and trustful.

Matrices $\mathbf{C_p}$, $\mathbf{C_x}$, $\mathbf{C_xp}$ and $\mathbf{C_px}$ which account for the modeling error in the expression of covariance matrices depends on the input force and the state variance. Therefore, a two steps process is necessary to estimate the input and the state taking into account the modeling errors:

1. First, the joint input-state estimation algorithm is run without any modeling error, *i.e.* $\mathbf{C_p}$, $\mathbf{C_x}$, $\mathbf{C_xp}$ and $\mathbf{C_px}$ are set to $\overline{\mathbf{0}}$.
2. $\mathbf{C_p}$, $\mathbf{C_x}$, $\mathbf{C_xp}$ and $\mathbf{C_px}$ are computed based on the estimated force and state obtained on the previous step and become parameters of the joint input-state estimation algorithm.

An example of input and output estimation for a 5% relative error on the Young modulus E, a 1% relative error on the volumic mass ρ and a 10% relative error on the damping ratios is presented

in figure 2.8.

MATLAB® script for Euler-Bernouilli beam model and joint input-state estimation algorithm is presented in Appendix C.

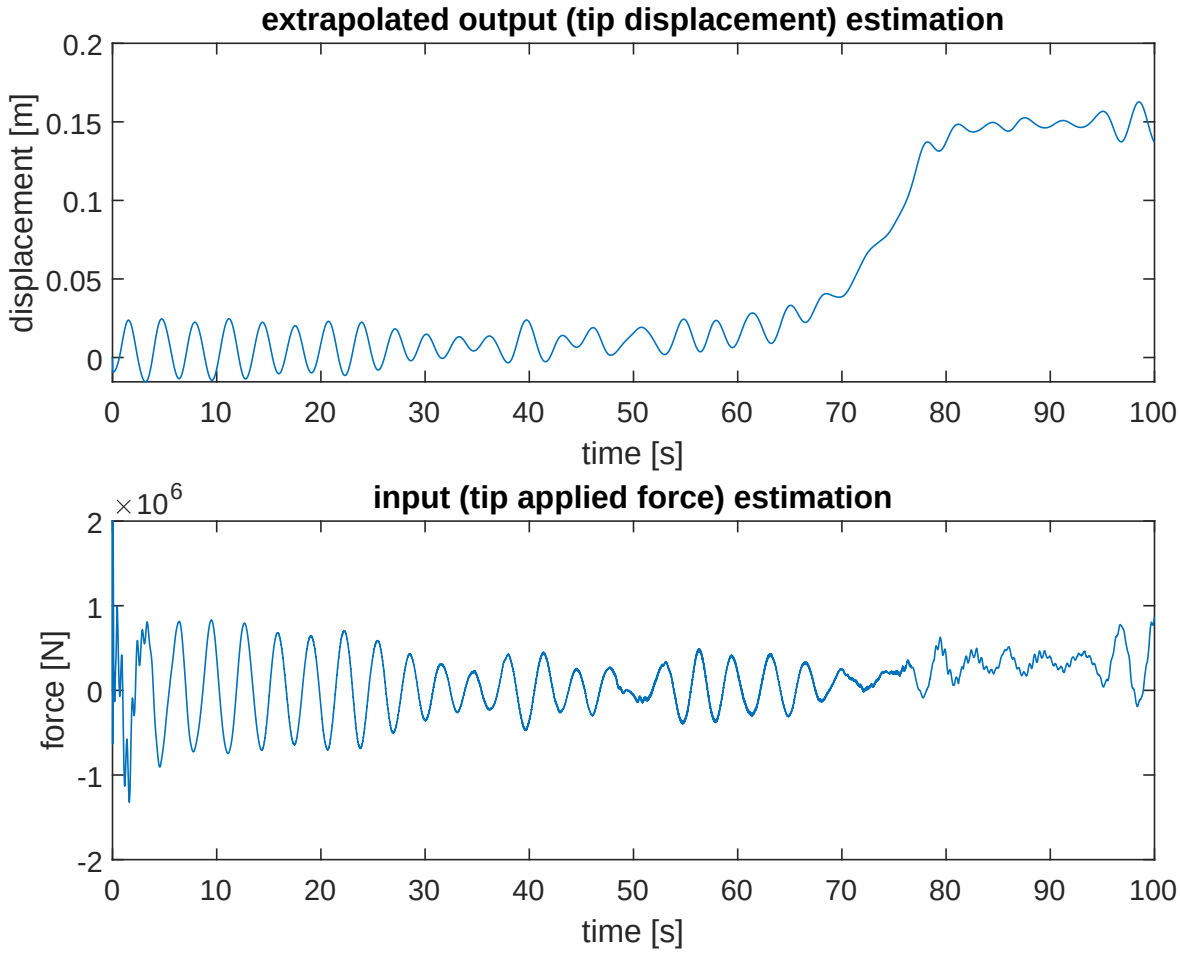


Fig. 2.8 : Input force and exptrapolated output estimation by the joint input-state estimation algorithm in presence of stochastic forces, measurement noise and modeling errors

The algorithm is run with the same parameters but without modeling errors. The results are presented in Appendix D .

From the comparison between the two results, one can see that the output estimation show a good behavior in presence of modeling errors. Concerning the input, as long as the stochastic forces remain dominant compared to the modeling error (cf. section 1.2.2, the estimation remains trusful which means that the filtering process is effective.

Sililarely to what is done in section 1.2.2, the impact of error on three parameters of the system are studied and compared next. The three parameters are :

- Young modulus E because during a windturbine lifetime, corrosion or other factors can occur and change the material properties of the monopile.

- Volumic mass ρ because a real turbine contains a lot of different equipement which add mass to the monopile. This is taken into account through uncertainety on the volumic mass.

- Damping matrix $\mathbf{\Gamma}$ because on a windturbine, damping depends both on windspeed and material properties. As no further concern is given to damping calculation in the algorithm, this phenomenon is simply replaced by uncertainty on the damping matrix.

First, error on damping matrix is set to 0. The dimensionless standart deviation of the error induced by modeling errors σ^*_{model}[3] on the extrapolated output for different values of error on Young modulus and on volumic mass is presented in table 2.1.

$\sigma^*_{model} \times 10^{-4}$	1%	5%	10%	20%	50%	80%
1%	1.82	5.25	6.09	6.44	6.77	7.09
5%	4.81	6.02	6.36	6.55	6.81	7.10
10%	6.11	6.45	6.57	6.66	6.85	7.12
20%	6.79	6.80	6.80	6.81	6.91	7.14
50%	7.14	7.10	7.08	7.05	7.04	7.19
80%	7.22	7.20	7.19	7.16	7.11	7.21

Table 2.1: Values of σ_{model} for different relative error on Young modulus (rows) and volumic mass (columns) without relative damping error.

For an error of 20% in damping matrix, the values of σ_{model} are significantly the same that without error on damping matrix. This observation confirms that damping is not the most important parameter in this algorithm.

Similarely to section 1.2.2, there is a limit on error introduced by modeling error on the output estimate. However, when the model error reach this limit error, it means that the filtering process is not efficient anymore and that the estimation relies almost completely on the measured quantities. The filtering inefficiency can be seen by looking at the standart deviation of the error induced by modeling errors $\sigma_{input_{mod}}$ on the input. Indeed, contrary to the output, there is not any input measurement that can prevent the input error to go to infinity.

Values of dimensionless $\sigma^*_{input_{mod}}$[4] for different values of Young modulus and volumic mass error and without error on damping matrix are displayed in table 2.2.

From table 2.2, one can see that the Young modulus is the more sensitive parameter, then comes the volumic mass ρ and finally the damping which does not have any real impact on the modeling error.

[3] σ_{model} is divided by the extrapolated output maximum value to obtain the dimensionless σ^*_{model}

[4] $\sigma_{input_{mod}}$ is divided by the input maximum value to obtain the dimensionless $\sigma^*_{input_{mod}}$

$\sigma^*_{inputmode}$ $[N]$	1%	5%	10%	20%	50%	80%
1%	0.088	0.255	0.302	0.360	0.556	1.040
5%	0.232	0.292	0.318	0.378	0.579	1.080
10%	0.295	0.314	0.332	0.399	0.609	1.131
20%	0.329	0.334	0.356	0.444	0.670	1.235
50%	0.345	0.362	0.425	0.592	0.860	1.549
80%	0.351	0.390	0.508	0.755	1.056	1.867

Table 2.2: Values of $\sigma^*_{inputmode}$ for different relative error on Young modulus (rows) and volumic mass (columns) for a relative damping error of 20%

Algorithm efficicency

As the algorithm is ment to be used in windturbine structural monitoring, it is important to be aware of its execution speed. Tipically, real-time monitoring is only possible if the algorithm execution time is lower than the time sample on which the algorithm is run.

A quick look at the estimation process shows the algorithm order: if n is the number of time steps, the algorithm is in $\mathcal{O}(n)$. Hence, if real-time monitoring is possible for one time sample size, then it is possible for any size of time sample.

For a 100 s time sample with a sampling frequency of $1250Hz$, when the model is reduced to its three first modes, the estimation process achieved by Matlab® takes 9.8 s. Then the real-time monitoring is possible with this kind of filtering algorithm.

2.3.3 Optimization of sensor layout

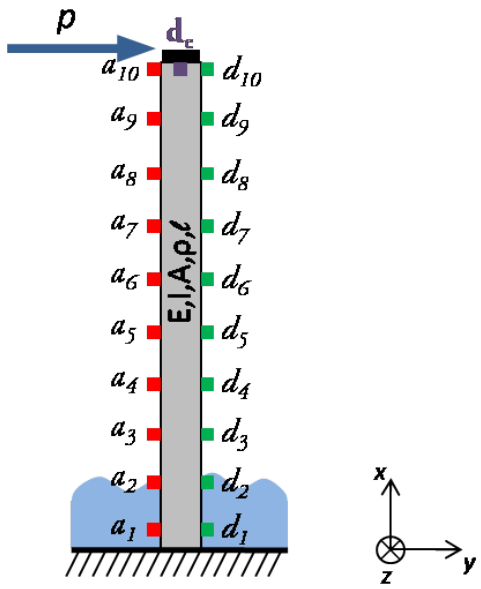
In this section, the error on the extrapolated output estimation $\mathbf{d_e}$ is studied and compared for different positions of the sensors: one accelerometer and one displacement sensor.

The error on the extrapolated output estimation - in case of zero mean and white stochastic processes assumption - correspond to the trace of the extrapolated output error covariance matrix $\mathbf{P_{de}}$ computed within the algorithm in equation 1.57.

In the following analysis, the measured output vector $\mathbf{d}$ includes a subset of ten horizontal acceleration, $a_1 - a_{10}$, and ten horizontal displacement, $d_1 - d_{10}$. The input $\mathbf{p}$ to be estimated is the force applied at the free end of the monopile. The output $\mathbf{d_e}$ to be estimated consists of the tip displacement of the monopile. The locations of the specified inputs and outputs are shown in figure 2.9.

For each sensor, the data are computed by FedEm® for a 100 s time sample with a sampling frequency of 1250 Hz.

The optimization process results are presented in figure 2.10.



sensor	height
a_1,d_1	9 m
a_2,d_2	20 m
a_3,d_3	31 m
a_4,d_4	42 m
a_5,d_5	53 m
a_6,d_6	64 m
a_7,d_7	75 m
a_8,d_8	86 m
a_9,d_9	97 m
a_{10},d_{10}	108 m

Fig. 2.9 : Force and sensor configuration (p: force, a_i accelerometer i, d_i displacement sensor i and $\mathbf{d_e}$ the output to be estimated).

Two contrary phenomena impact the error on the extrapolated output estimate:

- When the measurement is taken far from the extrapolated output location, the extrapolation process add a lot of error because of the modeling uncertainties. In the example of this report, the extrapolated output is the tip displacement, hence the lower the measurement will be taken, the bigger the error induced by modeling uncertainties will be.

- On an other hand, stochastic forces at a low level have less impact that stochastic forces applied close to the tip of the monopile. Hence when the measurements are taken high on the monopile, the error induced by stochastic forces is high.

The best sensor configuration is when the displacement measure is taken $97m$ above the mudline and the accelerometer is placed $64m$ above the mudline. Note that this is the best configuration between the configurations shown in figure 2.9, and that this optimizition process can be refined around this area by applying the same process on a smaller length. The optimization process writen in Matlab is given in Appendix E .

The transfer function analysis for each sensor position reveal that only sensors number 9 and 10 (displacement and acceleration) does not induce any unstable zeros. When the accelerometer does not induce any unstable zero, the joint input-state estimation algorithm works for any position of displacement sensor, even if some critical positions induce a very large error variance (position 1: $z = 9\ m$, 7: $z = 75\ m$ and 10: $z = 107.6\ m$).

However, for most of the displacement sensor positions, the error variance hardly depends on

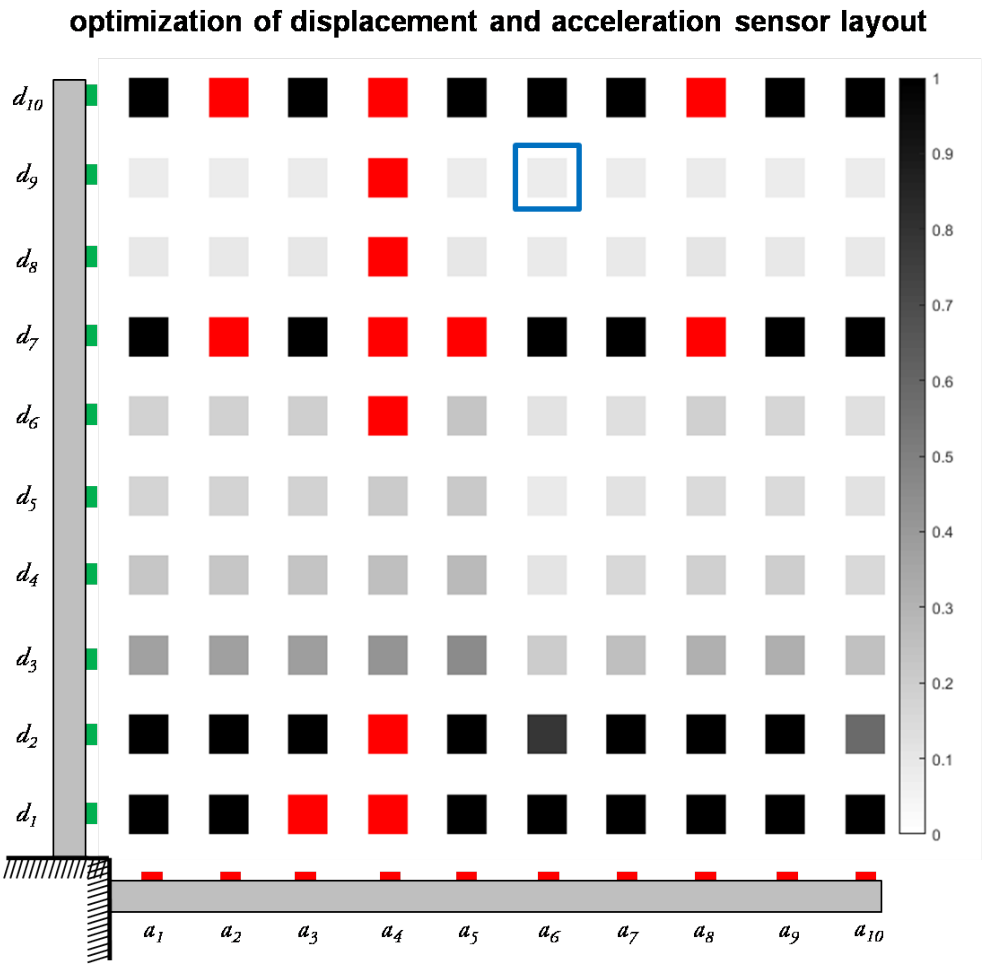


Fig. 2.10 : Variance of the error on the estimated output $\mathbf{d_e}$ in m for different sensor configurations. The configuration for which the smallest error variance is obtained is indicated by a blue square. Configurations for which the error is so high that the algorithm is unable to compute the output estimation are indicated in red.

the accelerometer position. This shows that the estimation mainly relies on the displacement data. Indeed, for each frequency, the joint input-state estimation algorithm weights the displacement and acceleration data in the estimation of system state. The weighting is based on the process noise and measurement noise covariance matrices, $\mathbf{Q}$, $\mathbf{S}$ and $\mathbf{R}$ (cf. section 1.2.2). At low frequencies, the information is mostly contained in the displacement data but at higher frequencies, the measurement noise disrupt the displacement information[5]. Accelerometers, especially piezoelectric ones are often more efficient at high frequencies than at very low frequencies. In the example studied in this report, the three first eigenfrequencies are 0.25 Hz, 2.4 Hz and 7.3 Hz, which are very low frequencies compared to the sampling frequency of 1250 Hz. Hence, for the studied windturbine, most of the information is contained in the displacement measure.

[5]A minimum of 4 points are necessary to describe a sinusoïdal wave. Hence the measurement can disrupt measures at frequencies higher than 1/4 of the sampling frenquency, in our case $1250/4 = 312Hz$

Algorithm response in absence of displacement measurement with only accelerometer

As it is said in section 2.2, the displacement sensors are not easy to install. Thus, if displacement measure can be avoided, it would save a large amount of money on the building process.

The algorithm is run for 3 acceleration measurements corresponding to a_4, a_9 and a_{10}. The extrapolated output data correspond to the acceleration taken at the tip of the monopile. Stochastic forces and modeling uncertainties are still the same that in section 2.3.2. Measurement noise is assumed to be $10^{-3} m/s^2$ for each accelerometer. Output estimation and force measurement are shown in figure 2.11.

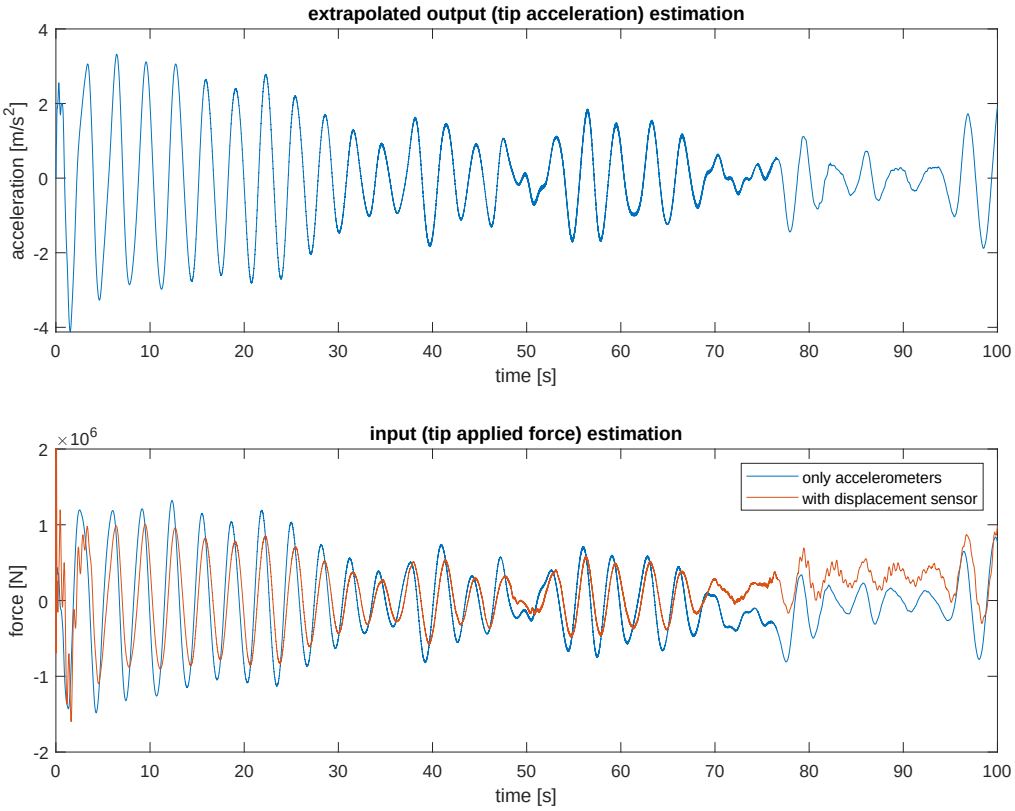


Fig. 2.11 : Extrapolated output and input force estimation with only acceleration measurement, in presence of modeling errors, stochastic forces and measurement noise. The input force estimation with displacement sensor is also plot in red for comparison

In the input force estimation, the small time shift between estimation with and without displacement sensor is due to the fact that the initial displacement is not exactly zero. When a displacement measure is given, the initial state can be known exactly but with only acceleration measurement, the initial state is assumed to be $\overline{0}$ which already induce a small error.

Moreover, one can see that around $t = 70s$, the force become globaly larger. This information can be seen with displacement sensor, as the monopile mean displacement will shift from zero to

a non-null value, but without displacement sensor, this overall increase information is totally lost. In practice, the wind is turbulent and can also show big time variation. In absence of displacement sensor, the strain induced by a wind speed increase or drop would not be seen by the joint input-state estimation algorithm. Variation on wind speed can induce charge/discharge cycles at very low frequencies compared to the structural eigenfrequencies which are taken into account in fatigue calculation.

The analysis of the joint-input estimation algorithm done so far does not enable to compute accurate results without any displacement sensor. Nevertheless, this research axis stays very interesting. The problem of overall displacement could for example be solved by adding the average wind speed as another algorithm input. In absence of displacement sensor, the stability conditions are violated but if the system does not include unstable zeros, *i.e.* if the accelerometers are located in the right spot, the system inversion can still be marginally stable [16].

2.4 Perspectives

This thesis fall within a large process of research on the Kalman Filter and on structural monitoring, this part is ment to put forward the possible improvements of the research done during this project.

First, in the windturbine model, the undamped natural frequencies and their corresponding mode shapes are computed under the assumption of proportional damping. The three first eigen frequencies (0.25 Hz, 2.4 Hz and 7.3 Hz) are quite high compared to eigen frequencies measured on a real windturbine, *e.g.* in [12] (0.360 Hz, 1.560 Hz and 3.910 Hz). This shows that the model of cantaliver beam with a tip mass under the assumption of proportional damping may be to simple and restrictive for an operating windturbine analysis. Instead of computing modal properties analytically, it could be done through finite element methods which would enable more complexity in the model.

Then, due to model reduction used in section 2.1.3, the algorithm keeps only the three first modes and does not take any information from the higher ones. However, higher modes which are responding in a quasistatic manner can be useful to measure the structure dynamic parameters such as stiffness and damping. Hence, this higher modes could be used to feed the algorithm with accurate and real-time model parameter and thus drasticly reduce the model uncertainties.

Thirdly, as it is said in section 2.3.3, deepening the research on the joint input-state estimation algorithm with only acceleration measurement is a very interesting research area. Solutions for periodical motion due to wind changes has to be found and then, an optimization process similar to what is done in section 2.3.3 can establish the optimal accelerometer distribution along the monopile.

Finally, this master thesis focuses on windturbine monitoring, but the main reason why monitoring is done is set aside. Indeed, calculation of fatigue damage is the final goal of the strain estimation process. Hence, this work could be carried further by analysing the value brought by the filtering process to fatigue analysis, and determine *e.g.* if filtering the signal with a such an algorithm can lead to a significent increase of lifetime.

Conclusion

In this master thesis, the joint input-state estimation algorithm, which can be seen as an extension of Kalman Filter is analysed on different systems in order to understand the impact of the different parameters which feed the algorithm.

A stability analysis of the algorithm revealed some conditions on the number and type of sensor. In case of a windturbine monopile analyis,only one main force which correspond to the load applied to the monopile tip through the action of wind on the rotor is to be estimated, then at least one acceleration sensor is necessary to satisfy the condition of system identifiability. Furthermore, in order to avoid the unstable transmission zeros defined in section 1.3.2 and to compute a unique input-state estimation, at least one displacement sensor has to be set on the monopile.

As this last condition has a heavy impact on sensoring cost, some analysis has been made to try to avoid the need of a displacement sensor. Unfortunately the results obtained in this thesis are not convincing and the filtering process becomes inefficient without displacement sensor. Nevertheless, this problematic stays really up to date as its resolution would lower the building and operation costs.

A special attention has been given to detection and quantification of the model uncertainties, almost always neglected in other studies for sake of simplicity. However, the model uncertainties should be a real concern for the analysis of a windturbine along its lifetime. Indeed, corrosion, cracks or other time factors can change the dynamical properties of the system. The main problem of model uncertainties is that, the error quantities computed within the joint input-state estimation algorithm are computed through system parameters hence, in presence of modeling errors, these quantities does not represent the true error anymore. One of the advantage of virtual simulation is that the input and output can be estimated with and without modeling errors and then, the error is simply computed by difference between the two estimations. In case of modeling errors, these are taken into accout in the algorithm in the covariance matrices.

Moreover, a process ment to detect the modeling errors based on Operational Modal Analysis (OMA) has been explained in section 1.2.2. Thus, the user is able to either correct the model or to include the uncertainty in the algorithm.

It is found that the filtering process is efficient only if the error induced by stochastic forces is predominant compared to the modeling errors (cf. 1.2.4). Indeed, with too high model uncertainties, the algorithm is unable to compute a proper input estimation, thus the algorithm does not

rely on the model anymore but only on the measurements.

Finally, an optimization process of sensor layout is made in this report. As the modeling error are assumed to have less impact than the stochastic forces and the measurement noise, the real error variance on the output estimate is assumed to be not too far from the error variance computed within the algorithm $\mathbf{P_{de}}$. The optimization process search the sensor layout for which the error variance is minimum, assuming that the sensor network is composed of one acceleration sensor and one displacement sensor. This optimization process gives an optimum solution presented in section 2.3.3 but shows also that most of the information is contained in the displacement sensor.

This work has been a great opportunity for me to deepen my knowledge and to learn a lot of new concepts in structural mechanics. As my background was more oriented around energy and fluid mechanics, a lot of concepts were totally new for me and they made my overall understanding of a windturbine increasing a lot. Besides, the Kalman Filter is widely used for a lot of application, even in a wider field that structural dynamics and it makes no doubt that my knowledge acquired thanks to this project on the Kalman Filter will be valuable in my future engineer life.

Bibliography

[1] R. Brincker, P. Andersen, and N. Møller. An indicator for separation of structural and harmonic modes in output-only modal testing. *International Modal Analysis Conference*, (IMAC XVIII):1649–1654, 2000.

[2] S. Gillijns and B. De Moor. Unbiased minimum-variance input and state estimation for linear discrete-time systems with direct feedthrough. *Automatica*, 43:934–937, 2007.

[3] G. H. James, T. G. Carne, and J. P. Lauffer. Complex mode indicition function and its application to spatial domain parameter estimation. *mssp*, 2(4):367–377, 1995.

[4] G. H. James, T. G. Carne, and J. P. Lauffer. The natural excitation technique (next) for modal parameter extraction from operating structures. *Journal of Analytical and Experimental Modal Analysis*, 10(4):260–277, 1995.

[5] Roy R. Craig Jr and Andrew J. Kurdila. *Fundamentals of structural dynamics*. John Wiley and Sons, 2006.

[6] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transaction of the ASME - Journal of Basic Engineering*, 82(1):35–45, 1960.

[7] S. Koskie. Discretization of continuous time state space systems, October 2005. Course from Perdue School of Engineering and Technology.

[8] P. A. A. Laura, J. L. Pombo, and E. A. Susemihl. A note on the vibrations of a clamped-free beam with a mass at the free end. *Journal of Sound and Vibration*, 37(2):161–168, 1974.

[9] E. Lourens and al. Joint input-response estimation for structural systems based on reduced-order models and vibration data from a limited number of sensors. *mssp*, 29:310–327, 2012.

[10] K. Maes and al. Design of sensor networks for instantaneous inversion of modally reduced order models in structural dynamics. *mssp*, 52-53:628–644, 2015.

[11] K. Maes and al. Joint input-state estimation in structural dynamics. *mssp*, 70:445–466, 2016.

[12] K. Maes and al. Kalman filter based strain estimation for fatigue assessment of an offshore monopile wind turbine. *mssp*, 76-77:592–611, 2016.

[13] J. Massey and M. Sain. Inverse of linear sequential circuits. *ieee*, C-17, 4:330–337, 1968.

[14] J. S. Mendat and A. G. Piersol. *Engineering applications of correlation and spectral analysis.* New York, Wiley-Interscience, 1980.

[15] Ö. Morgül. Dynamic boudary control of an euler-bernoulli beam. *ieee transactions on automatic control*, 37:639–642, 1992.

[16] P. Moylan. Stable inversion of linear systems. *ieee*, 22:74–78, 1977.

[17] A. Papoulis. *probability, Random Variables and Stochastic Processes - 3rd ed.* Mc Graw-Hill, Inc, 1991.

[18] A. Preumont. *Twelve Lectures on Structural Dynamics.* G.M.L. Gladwell, 2013.

[19] H. H. Rosenbrock. *State space and multivariable theory.* New York : Wiley Interscience Division, 1970.

[20] WindEurope. Wind in power 2017, annual combined onshore and offshore wind energy statistics, February 2018. Annual report.

[21] WindEurope. Wind energy in europe: senarios for 2030, September 2030.

Appendices

Appendix A

Steps of the Kalman Filter (with input estimation)

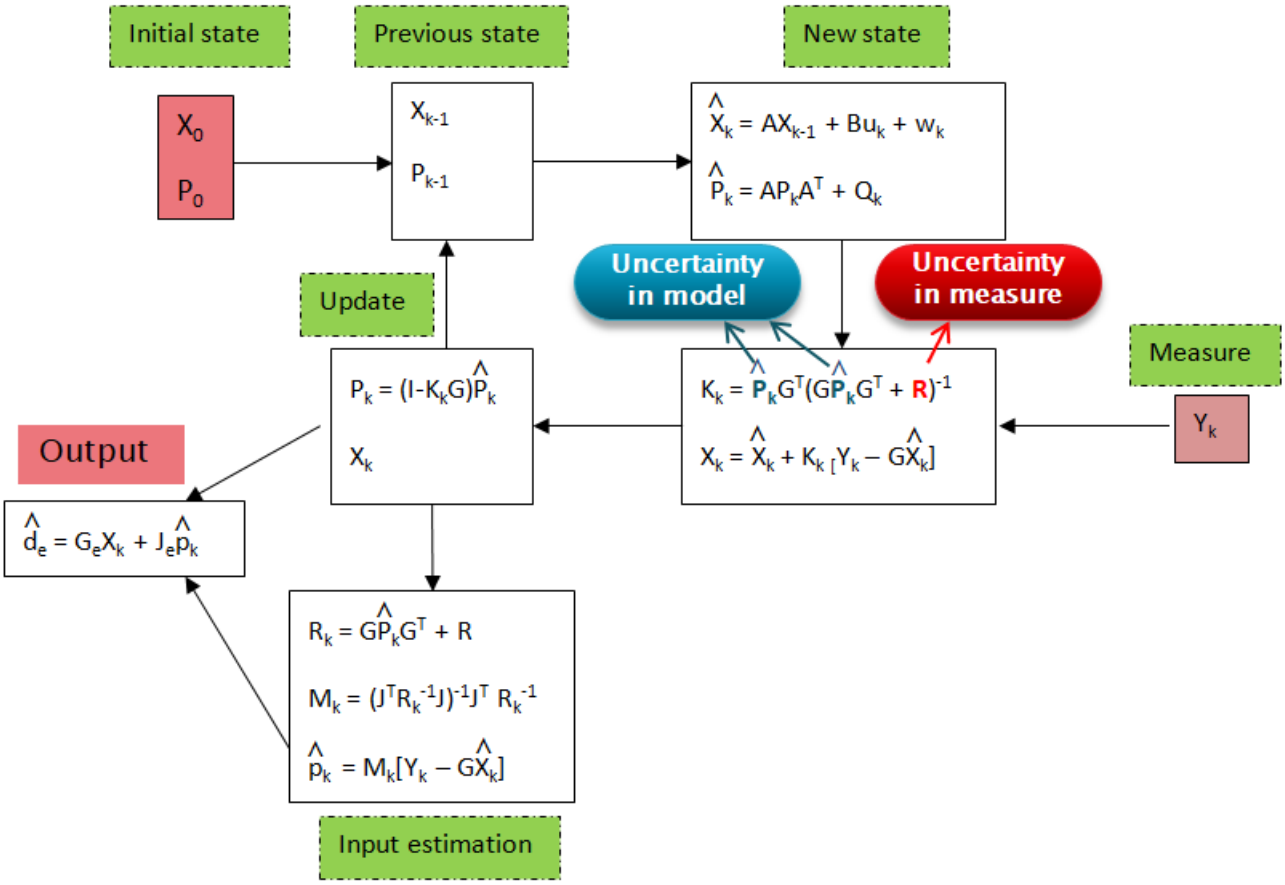


Fig. 12 : Sketch representing the different steps of the Kalman Filter Algorithm

Appendix B

Matlab® script for joint input state algorithm applied on a SDOF system. Files *measures.xlsx* contains measurement for displacement and velocity and file *mea_accel.xlsx* contains measurement for acceleration.

Matlab script

```
%Kalman filter algorithm tracking a single dof system and estimating
mea = [];
%Measurements%
measures_dis_vel = xlsread('measures.xlsx')';
mea(1,:) = measures_dis_vel(1,:);
mea(2,:) = xlsread('mea_accel.xlsx');
%Time%
t = linspace(0,10,length(mea));
dt = (t(end) - t(1))/(length(t)-1); %Time step%
% Noise in the measurements
sigma_dis = 10^(-3);
av_dis = 0;
sigma_acc = 10^(-2);
av_acc = 0;
Vm(1,:) = sigma_dis*randn(1,length(mea));
Vm(2,:) = sigma_acc*randn(1,length(mea));
mea = mea + Vm;

% System parameters %
m = 10; %[kg]%
k = 1000; %[N/m]%
c = 2.0; %[kg/s]%

%Define matrices
Rm = [(sigma_dis)^2 0; 0 (sigma_acc)^2]; % [(m/s^2)^2] measure
   uncertainty
Cp = (2)^2; %Covariance matrix (stochastic force with standart deviation
    of 2 N)
%State matrices
A = [1 dt; -dt*k/m 1-dt*c/m];
B = [0; dt/m];
J = [0; 1/m];
G = [1 0 ;-k/m -c/m];
size_B = size(B);
size_A = size(A);
```

```
34 np = size_B(2);
35 ns = size_A(2);
36 %Stochastic excitation matrix
37 Bs = [0; dt/m];
38 Js = [0; 1/m];
39 %Output quantity matrices
40 Ge = [1 0];
41 Je = 0;
42 Jes = 0;%Matrix relating extrapolated output vector to vector of
      sthocastique forces
43 % Noise processes covariance matrices
44 n = length(B);
45 l = length(J);
46 COV = [Bs;Js]*Cp*[Bs' Js'];
47 Q = COV(1:n,1:n);
48 S = COV(1:n,n+1:length(COV));
49 R = COV(n+1:length(COV),n+1:length(COV)) + Rm;
50 Re = Jes*Cp*Jes';
51 Rc = Jes*Cp*Js';
52
53 %Pre-definition of variables
54 x = zeros(n,length(mea));
55 x_e = zeros(n,length(mea));
56 Px = zeros(n); %size of B matrix
57 Pp = zeros(length(J(1,:))); %size of J matrix
58 f = zeros(1,length(mea));
59 de = zeros(1,length(mea));
60
61 %Initialization
62 y0 = 0; %Initial position
63 v0 = 0; %Initial velocity
64 Px0 = [0 0;0 0];
65 x(:,1) = [y0; v0];
66 Px = Px0;
67 Px_e = Px;
68
69 %Looping process
70 for i=2:length(mea)
71     %Input estimation
72     R_k = G*Px_e*G'+R;
73     M_k = (J'/R_k*J)\(J'/R_k);
74     f(i) = M_k*(mea(:,i)-G*x(:,i-1));
75     Pp = eye(size(Pp))/(J'/R_k*J);
```

```
76      %Measurement update
77      K = Px_e*G'/R_k;
78      x_e(:,i) = x(:,i-1)+K*(mea(:,i)-G*x(:,i-1)-J*f(i));
79      Px = Px_e-K*(R_k-J*Pp*J')*K';
80      Pxp = -K*J*Pp;
81      %Time update
82      x(:,i) = A*x_e(:,i)+B*f(i);
83      I=eye(size(J*M_k));
84      N_k = A*K*(I-J*M_k)+B*M_k;
85      Px_e = [A B]*[Px Pxp;Pxp' Pp]*[A';B']+Q+N_k*S'+S*N_k';
86      %Output vector
87      de(i) = Ge*x_e(:,i)+Je*f(i);
88      Z = Ge*K*(I-J*M_k)+Je*M_k;
89      Pde = Ge*Px*Ge'+Je*Pp*Je'+Ge*Pxp*Je'+Je*Pxp'*Ge'+Re-Z*Rc'-Rc*Z';
90  end
```

Appendix C

Matlab® script for joint input state algorithm applied on a euler-bernouilli cantaliver beam with a tip mass at the free end. File *position_X_97m_100sec.xlsx* contains position measurement for 100 sec at $z = 97\ m$ and *acceleration_X_64m_100sec.xlsx* contains acceleration measurement for 100 sec at $z = 64\ m$

Matlab script

```
%Kalman filter algorithm tracking a beam and estimating excitation
close all;clc;

%measurements
alt_d = 97; %altitude of displacement sensor
alt_a = 64; %altitude of accelerometer
mea_dis = xlsread('position_X_97m_100sec.xlsx');
mea_acc = xlsread('acceleration_X_64m_100sec.xlsx');
mea = [mea_dis(:,2)' ; mea_acc(:,2)'];

t = linspace(0,100,length(mea));
dt = (t(end) - t(1))/(length(t)-1); %Time step%

%System parameters
%Geometry
geometry.L = 107.6; % beam length (m)
geometry.E = 2.1e11; % Young Modulus (Pa)
geometry.nu = 0.3; % Poisson ratio
geometry.rho = 8500; % density (kg/m^3)
geometry.t = 0.06 ; %thikness of the tube (m)

% case of a tube
D = 6 ; % beam diameter (m)
d = D - geometry.t ; % Inner diameter of the tube (m)
% beam is symmetrical around axes y and z
Iy = pi.*(D.^4 - d.^4)./64;
Ix = pi.*(D.^4 - d.^4)./64;
geometry.I = Iy; % affectation of quadratic moment

%Mass at the tip of the beam
geometry.Mtip = 400*10^(3) ; %(kg)

ndisc = 108 ; % number of discretisation points for the beam
geometry.y = linspace(0,geometry.L ,ndisc);
```

```
35 V = ((pi.*(D/2).^2)-(pi.*(d/2).^2))*geometry.L;
36 geometry.m = (geometry.rho.*V)./geometry.L;
37
38
39 % Number of modes
40 Nmodes =3; % number of mode wanted
41 [phi,wn] = eigenModes(geometry,Nmodes);
42 %Eigen frequencies
43 Omega = diag(wn);
44 %Damping ratio
45 Gamma = 2/100*diag([1.89 1.38 0.89]).*Omega;
46
47 %Excitation force location
48 np = 1; %Number of forces applying on the structure
49 Sp = zeros(ndisc, np);
50 Sp(ndisc,1) = 1; %1 force located on the tip of the beam
51 %Measurement transmission matrices
52 nd = 2; %number of measurement (disp, vel and acc)
53 Sdd = zeros(nd,ndisc);
54 Sdd(1,alt_d) = 1 ; %Displacement measurement taken at 20 m from the
      bottom
55 Sdv = zeros(nd,ndisc); %No velocity measurement
56 Sda = zeros(nd,ndisc);
57 Sda(2,alt_a) = 1 ; %Acceleration measurement taken at 20 m from the tip
58
59 %Modeling errors
60 %Geometry
61 geometry_err.L = 1*geometry.L; % beam length (m)
62 geometry_err.E = (1+0.05)*geometry.E; % Young Modulus (Pa)
63 geometry_err.nu = 1*geometry.nu; % Poisson ratio
64 geometry_err.rho = (1-0.01)*geometry.rho; % density (kg/m^3)
65 geometry_err.t = 1*geometry.t; % thikness (m)
66
67
68 % case of a tube
69 D_err = 6; % beam diameter (m)
70 d_err = D_err - geometry_err.t ; % Inner diameter of the tube (m)
71 % beam is symmetrical around axes y and z
72 Iy_err =  pi.*(D_err.^4 - d_err.^4)./64;
73 Ix_err =  pi.*(D_err.^4 - d_err.^4)./64;
74 geometry_err.I = Iy_err; % affectation of quadratic moment
75
76 geometry_err.y = linspace(0,geometry_err.L ,ndisc);
```

```
77 V_err = ((pi.*(D_err/2).^2)-(pi.*(d_err/2).^2))*geometry_err.L;
78 geometry_err.m = geometry_err.rho.*V_err./geometry_err.L;
79
80 % Number of modes
81 Nmodes =3; % number of mode wanted
82
83 [phi_err,wn_err] = eigenModes(geometry_err,Nmodes);
84
85 %Eigen frequencies
86 Omega_err = diag(wn_err);
87 %Damping ratio
88 Gamma_err = (1+0.1)*Gamma;
89
90 %State space model for model error
91 ns = 2*Nmodes;
92 A_err = exp([zeros(Nmodes) eye(Nmodes); -Omega_err.^2 -Gamma_err]*dt);
93 B_err = (A_err - eye(ns))/[zeros(Nmodes) eye(Nmodes); -Omega_err.^2 -
      Gamma_err]*[zeros(Nmodes,np); phi_err*Sp];
94 G_err = [Sdd*phi_err' - Sda*phi_err'*Omega_err^2 , Sdv*phi_err' - Sda*
      phi_err'*Gamma_err];
95 J_err = (Sda*(phi_err'*phi_err)*Sp);
96
97 %State space model
98 A = exp([zeros(Nmodes) eye(Nmodes); -Omega.^2 -Gamma]*dt);
99 B = (A - eye(ns))/[zeros(Nmodes) eye(Nmodes); -Omega.^2 -Gamma]*[zeros(
      Nmodes,np); phi*Sp];
100 G = [Sdd*phi' - Sda*phi'*Omega^2 , Sdv*phi' - Sda*phi'*Gamma];
101 J = (Sda*(phi'*phi)*Sp);
102
103 delta_A = abs(A_err - A);
104 delta_B = abs(B_err - B);
105 delta_G = abs(G_err - G);
106 delta_J = abs(J_err - J);
107
108 [x_est,f_est] = Joint_input_state_beam_func(mea,alt_d,alt_a);
109
110 Cp = var(f_est)*eye(np,np);
111 Cx =diag(var(x_est'));
112 Cxp = zeros(ns,np);
113 for i=1:ns
114     Cxp(i) = [1 0]*cov(x_est(i,:),f_est)*[0;1];
115 end
116 Cpx = Cxp';
```

```
117
118 %Stochastic forces
119 nps = ndisc; %Number of stochastic forces
120 ps = ones(nps,1); %Stochastic forces distributed all along the beam
121 ps(1:19) = 1000*ones(19,1);
122 ps(20) = 5000;
123 ps(21:nps) = 100*ones(nps-20,1);
124 Cps = diag(ps).^2; %Stochastic forces of 10 N standart deviation
125 Bs = zeros(ns,nps);
126 Js = zeros(nd,nps);
127 for i=1:nps
128     Sps = zeros(nps,1);
129     Sps(i) = ps(i);
130     Bs(:,i)=(A - eye(ns))/[zeros(Nmodes) eye(Nmodes); -Omega.^2 -Gamma
        ]*[zeros(Nmodes,np); phi*Sps];
131     Js(:,i)=(Sda*(phi'*phi)*Sps);
132 end
133
134 %error covariance matrices
135 Rm = zeros(nd);
136 Rm(1,1) = 10^(-1); %measurement error on first measure
137 Rm(2,2) = 10^(-1); %measurement error on second measure
138 %Rm(3,3) = ... %measurement error on third measure etc.
139 Q = Bs*Cps*Bs' + delta_A*Cx*delta_A' + delta_B*Cp*delta_B' + delta_A*Cxp
    *delta_B' + delta_B*Cpx*delta_A';
140 S = Bs*Cps*Js' + delta_A*Cx*delta_G' + delta_B*Cp*delta_J' + delta_A*Cxp
    *delta_J' + delta_B*Cpx*delta_G';
141 R = Js*Cps*Js' + Rm + delta_G*Cx*delta_G' + delta_J*Cp*delta_J' +
    delta_G*Cxp*delta_J' + delta_J*Cpx*delta_G' ;
142
143 %Output vector
144 nde = 1; %Number of output to be estimated
145 Sdde = zeros(nde,ndisc);
146 Sdde(1,100) = 1 ; %Displacement measurement taken at the tip of the beam
147 Sdve = zeros(nde,ndisc); %No velocity output
148 Sdae = zeros(nde,ndisc); %No acceleration output
149 Ge = [Sdde*phi' - Sdae*phi'*Omega^2 , Sdve*phi' - Sdae*phi'*Gamma];
150 Je = (Sdae*(phi'*phi)*Sp);
151 Jes = zeros(nde,nps); %Relating output vector to stochastic forces
152
153 %output covariance matrices
154 Re = Jes*Cps*Jes';
155 Rc = Jes*Cps*Js';
```

```
156
157 %Pre-definition of variables
158 x = zeros(ns,length(mea));
159 x_e = zeros(ns,length(mea));
160 Px = zeros(ns,ns);
161 Pp = zeros(np,np); %size of J matrix
162 f = zeros(np,length(mea));
163 de = zeros(nde,length(mea));
164
165 %Initialization
166 y0 = phi*mea(1,1)*ones(ndisc,1); %Initial position (modal coordinates)
    for each element
167 v0 = phi*mea(2,1)*ones(ndisc,1); %Initial velocity (modal coordinates)
    for each element
168 Px0 = zeros(size(Px));
169 x(:,1) = [y0; v0];
170 Px = Px0;
171 Px_e = Px;
172
173 %Looping process
174 tic
175 for i=2:length(mea)
176     %Input estimation
177     R_k = G*Px_e*G'+R;
178     M_k = (J'/R_k*J)\(J'/R_k);
179     f(i) = M_k*(mea(:,i)-G*x(:,i-1));
180     Pp = eye(size(Pp))/(J'/R_k*J);
181     %Measurement update
182     K = Px_e*G'/R_k;
183     x_e(:,i) = x(:,i-1)+K*(mea(:,i)-G*x(:,i-1)-J*f(i));
184     Px = Px_e-K*(R_k-J*Pp*J')*K';
185     Pxp = -K*J*Pp;
186     %Time update
187     x(:,i) = A*x_e(:,i)+B*f(i);
188     I=eye(size(J*M_k));
189     N_k = A*K*(I-J*M_k)+B*M_k;
190     Px_e = [A B]*[Px Pxp;Pxp' Pp]*[A';B']+Q+N_k*S'+S*N_k';
191
192     %Output vector
193     de(i) = Ge*x_e(:,i)+Je*f(i);
194     Z = Ge*K*(I-J*M_k)+Je*M_k;
195     Pde = Ge*Px*Ge'+Je*Pp*Je'+Ge*Pxp*Je'+Je*Pxp'*Ge'+Re-Z*Rc'-Rc*Z';
196 end
```

```
197 toc
```

eigenModes function

```
1  function [phi,wn,M] = eigenModes(geometry,Nmodes)
2
3  y= geometry.y;
4  Nz = numel(y); % number of discrete elements
5
6  % volume V and mass m of the beam
7  m = geometry.m;
8  L = geometry.L;
9
10 D = 6 ; % beam diameter (m)
11 d = D - geometry.t ; % Inner diameter of the tube (m)
12 Mtip = 700000 ;
13 %Mass matrix for each element
14 m_el = (1/2)*geometry.rho*((pi.*(D/2).^2)-(pi.*(d/2).^2))*geometry.L/Nz
       *[1 0;0 1];
15 %Mass matrix
16 M = zeros(Nz);
17 for i = 1:Nz-1
18     M(i:i+1,i:i+1) = M(i:i+1,i:i+1) + m_el ;
19 end
20 M(Nz,Nz)=M(Nz,Nz)+Mtip;
21
22 % get the non trivial solution of f
23
24 if Nmodes<10,
25     Ndummy=1:Nmodes^2; % the number is arbitrary fixed as the square of
          the number of Nmodes.
26 else
27     Ndummy=1:100; % the number is arbitrary fixed as the square of the
          number of Nmodes.
28 end
29 tolX = 1e-8;
30 tolFun = 1e-8;
31 options=optimset('TolX',tolX,'TolFun',tolFun,'Display','off');
32 h =fsolve(@modeShape,Ndummy,options);
33 % small values of h come from numerical errors
34 h(h<0.4)=[];
35 % Analytically, many solutions are identical to each other, but
       numerically, it is not the
36 % case. Therefore I need to limit the prceision of the solutions to 1e
```

```
    -4.
h = round(h*1e4).*1e-4;
% the uniques solution are called beta:
beta = unique(h);
beta = beta(1:min(numel(beta),Nmodes));

%modes shapes calculations
wn = beta.^2.*sqrt(geometry.E*geometry.I./(m.*geometry.L^4)); % in rad
phi = zeros(Nmodes,Nz);

    for ii=1:Nmodes
        phi(ii,:)=(cosh(beta(ii)*y./L)-cos(beta(ii)*y./L))+...
                ((cosh(beta(ii))+cos(beta(ii))).*(sin(beta(ii)*y./L)-
                   sinh(beta(ii)*y./L)))./...
                (sin(beta(ii))+sinh(beta(ii)));

        phi(ii,:)=phi(ii,:)/sqrt((phi(ii,:)*M*phi(ii,:)')); %Mass
           normalization
    end

end
```

modeShape function

```
function [f] = modeShape(y)
    Mtip = 400*10^3; %[kg]
    rho = 8500; %[kg/m^3]
    l = 107.6; %[m]
    D = 6; %[m]
    t = 0.06 ; %[m]
    d = D - t; %[m]
    M = ((pi.*(D/2).^2)-(pi.*(d/2).^2))*l*rho ; %[kg]
    f=cos(y).*cosh(y) + 1 +  Mtip/M.*y.*(cos(y).*sinh(y) - sin(y).*cosh(
       y)) ;
end
```

Joint input state beam func function

This function is the joint input state estimation algorithm without modeling error, used to have a first estimate of the input and output.

```
function[x,f] = Joint_input_state_beam_func(mea,alt_d,alt_a)

t = linspace(0,100,125001);
dt = (t(end) - t(1))/(length(t)-1); %Time step%
```

```
5
6  %System parameters
7  %Geometry
8  geometry.L = 107.6; % beam length (m)
9  geometry.E = 2.1e11; % Young Modulus (Pa)
10 geometry.nu = 0.3; % Poisson ratio
11 geometry.rho = 8500; % density (kg/m^3)
12 geometry.t = 0.06 ; %thikness of the tube (m)
13
14 % case of a tube
15 D = 6 ; % beam diameter (m)
16 d = D - geometry.t ; % Inner diameter of the tube (m)
17 % beam is symmetrical around axes y and z
18 Iy = pi.*(D.^4 - d.^4)./64;
19 Ix = pi.*(D.^4 - d.^4)./64;
20 geometry.I = Iy; % affectation of quadratic moment
21
22 %Mass at the tip of the beam
23 geometry.Mtip = 400*10^(3) ; %(kg)
24
25 ndisc = 108 ; % number of discretisation points for the beam
26 geometry.y = linspace(0,geometry.L ,ndisc);
27 V = ((pi.*(D/2).^2)-(pi.*(d/2).^2))*geometry.L;
28 geometry.m = (geometry.rho.*V)./geometry.L;
29
30
31 % Number of modes
32 Nmodes =3; % number of mode wanted
33 [phi,wn] = eigenModes(geometry,Nmodes);
34 %Eigen frequencies
35 Omega = diag(wn);
36 %Damping ratio
37 Gamma = 2/100*diag([1.89 1.38 0.89]).*Omega;
38
39 %Excitation force location
40 np = 1; %Number of forces applying on the structure
41 Sp = zeros(ndisc, np);
42 Sp(ndisc,1) = 1; %1 force located on the tip of the beam
43 %Measurement transmission matrices
44 nd = 2; %number of measurement (disp, vel and acc)
45 Sdd = zeros(nd,ndisc);
46 Sdd(1,alt_d) = 1 ; %Displacement measurement taken at 20 m from the
     bottom
```

```
47 Sdv = zeros(nd,ndisc); %No velocity measurement
48 Sda = zeros(nd,ndisc);
49 Sda(2,alt_a) = 1 ; %Acceleration measurement taken at 20 m from the tip
50
51 %State space model
52 ns = 2*Nmodes;
53 A = exp([zeros(Nmodes) eye(Nmodes); -Omega.^2 -Gamma]*dt);
54 B = (A - eye(ns))/[zeros(Nmodes) eye(Nmodes); -Omega.^2 -Gamma]*[zeros(
       Nmodes,np); phi*Sp];
55 G = [Sdd*phi' - Sda*phi'*Omega^2 , Sdv*phi' - Sda*phi'*Gamma];
56 J = (Sda*(phi'*phi)*Sp);
57
58 %Stochastic forces
59 nps = ndisc; %Number of stochastic forces
60 ps = ones(nps,1); %Stochastic forces distributed all along the beam
61 ps(1:19) = 1000*ones(19,1);
62 ps(20) = 5000;
63 ps(21:nps) = 100*ones(nps-20,1);
64 Cps = diag(ps).^2; %Stochastic forces of 10 N standart deviation
65 Bs = zeros(ns,nps);
66 Js = zeros(nd,nps);
67 for i=1:nps
68     Sps = zeros(nps,1);
69     Sps(i) = ps(i);
70     Bs(:,i)=(A - eye(ns))/[zeros(Nmodes) eye(Nmodes); -Omega.^2 -Gamma
          ]*[zeros(Nmodes,np); phi*Sps];
71     Js(:,i)=(Sda*(phi'*phi)*Sps);
72 end
73
74 %error covariance matrices
75 Rm = [10^(-1) 0; 0 10^(-1)]; %measurement error on displacement and
       acceleration
76 Q = Bs*Cps*Bs';
77 S = Bs*Cps*Js' ;
78 R = Js*Cps*Js' + Rm ;
79
80 %Output vector
81 nde = 1; %Number of output to be estimated
82 Sdde = zeros(nde,ndisc);
83 Sdde(1,100) = 1 ; %Displacement measurement taken at the tip of the beam
84 Sdve = zeros(nde,ndisc); %No velocity output
85 Sdae = zeros(nde,ndisc); %No acceleration output
86 Ge = [Sdde*phi' - Sdae*phi'*Omega^2 , Sdve*phi' - Sdae*phi'*Gamma];
```

```
87  Je = (Sdae*(phi'*phi)*Sp);
88  Jes = zeros(nde,nps); %Relating output vector to stochastic forces
89
90  %output covariance matrices
91  Re = Jes*Cps*Jes';
92  Rc = Jes*Cps*Js';
93
94  %Pre-definition of variables
95  x = zeros(ns,length(mea));
96  x_e = zeros(ns,length(mea));
97  Px = zeros(ns,ns);
98  Pp = zeros(np,np); %size of J matrix
99  f = zeros(np,length(mea));
100 de = zeros(nde,length(mea));
101
102 %Initialization
103 y0 = phi*mea(1,1)*ones(ndisc,1); %Initial position (modal coordinates)
    for each element
104 v0 = phi*mea(2,1)*ones(ndisc,1); %Initial velocity (modal coordinates)
    for each element
105 Px0 = zeros(size(Px));
106 x(:,1) = [y0; v0];
107 Px = Px0;
108 Px_e = Px;
109 Rk_test = zeros(2,2,length(mea));
110 Px_test = zeros(ns,ns,length(mea));
111 Pp_test = zeros(np,np,length(mea));
112 Pxp_test = zeros(ns,np,length(mea));
113
114 %Looping process
115 for i=2:length(mea)
116     %Input estimation
117     R_k = G*Px_e*G'+R;
118     Rk_test(:,:,i) = R_k;
119     M_k = (J'/R_k*J)\(J'/R_k);
120     f(i) = M_k*(mea(:,i)-G*x(:,i-1));
121     Pp = eye(size(Pp))/(J'/R_k*J);
122     Pp_test(:,:,i) = Pp;
123     %Measurement update
124     K = Px_e*G'/R_k;
125     x_e(:,i) = x(:,i-1)+K*(mea(:,i)-G*x(:,i-1)-J*f(i));
126     Px = Px_e-K*(R_k-J*Pp*J')*K';
127     Px_test(:,:,i) = Px;
```

```
128     Pxp = -K*J*Pp;
129     Pxp_test(:,:,i) = Pxp;
130     %Time update
131     x(:,i) = A*x_e(:,i)+B*f(i);
132     I=eye(size(J*M_k));
133     N_k = A*K*(I-J*M_k)+B*M_k;
134     Px_e = [A B]*[Px Pxp;Pxp' Pp]*[A';B']+Q+N_k*S'+S*N_k';
135
136     %Output vector
137     de(i) = Ge*x_e(:,i)+Je*f(i);
138     Z = Ge*K*(I-J*M_k)+Je*M_k;
139     Pde = Ge*Px*Ge'+Je*Pp*Je'+Ge*Pxp*Je'+Je*Pxp'*Ge'+Re-Z*Rc'-Rc*Z';
140 end
```

Appendix D

Estimation results without modeling errors:

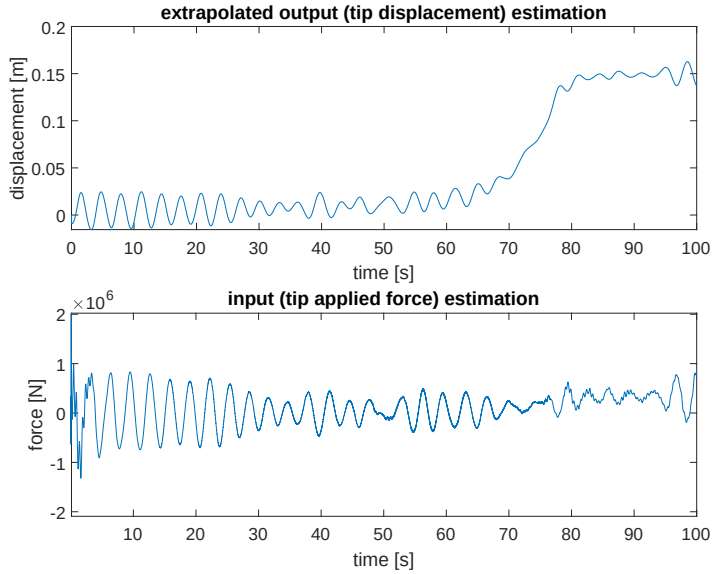


Fig. 13 : Joint input state test estimations without modeling error

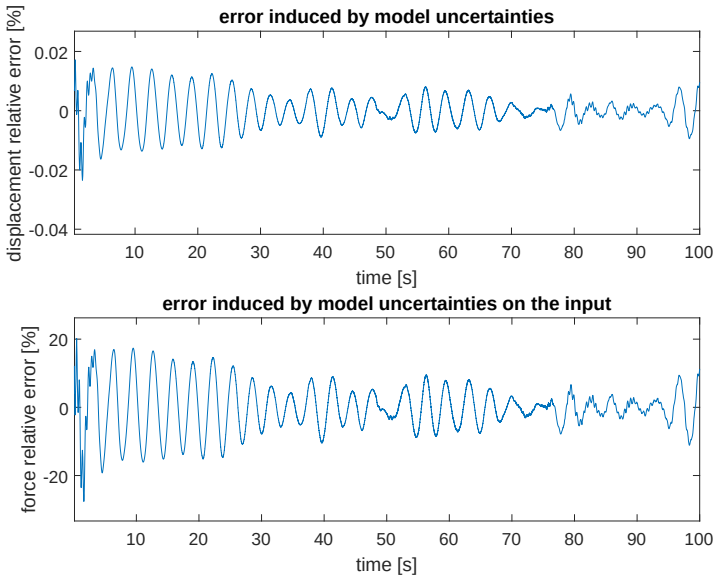


Fig. 14 : Error iduced by modeling uncertainties - difference between figure 13 and figure 2.8

Appendix E

Optimization of the sensor layout process:

Matlab script

```
%measurements
mea = zeros(2,125001,10);
mea_dis_1 = xlsread('position_X_9m_100sec.xlsx');
mea_acc_1 = xlsread('acceleration_X_9m_100sec.xlsx');
mea(:,:,1) = [mea_dis_1(:,2)' ; mea_acc_1(:,2)'];
mea_dis_2 = xlsread('position_X_20m_100sec.xlsx');
mea_acc_2 = xlsread('acceleration_X_20m_100sec.xlsx');
mea(:,:,2) = [mea_dis_2(:,2)' ; mea_acc_2(:,2)'];
mea_dis_3 = xlsread('position_X_31m_100sec.xlsx');
mea_acc_3 = xlsread('acceleration_X_31m_100sec.xlsx');
mea(:,:,3) = [mea_dis_3(:,2)' ; mea_acc_3(:,2)'];
mea_dis_4 = xlsread('position_X_42m_100sec.xlsx');
mea_acc_4 = xlsread('acceleration_X_42m_100sec.xlsx');
mea(:,:,4) = [mea_dis_4(:,2)' ; mea_acc_4(:,2)'];
mea_dis_5 = xlsread('position_X_53m_100sec.xlsx');
mea_acc_5 = xlsread('acceleration_X_53m_100sec.xlsx');
mea(:,:,5) = [mea_dis_5(:,2)' ; mea_acc_5(:,2)'];
mea_dis_6 = xlsread('position_X_64m_100sec.xlsx');
mea_acc_6 = xlsread('acceleration_X_64m_100sec.xlsx');
mea(:,:,6) = [mea_dis_6(:,2)' ; mea_acc_6(:,2)'];
mea_dis_7 = xlsread('position_X_75m_100sec.xlsx');
mea_acc_7 = xlsread('acceleration_X_75m_100sec.xlsx');
mea(:,:,7) = [mea_dis_7(:,2)' ; mea_acc_7(:,2)'];
mea_dis_8 = xlsread('position_X_86m_100sec.xlsx');
mea_acc_8 = xlsread('acceleration_X_86m_100sec.xlsx');
mea(:,:,8) = [mea_dis_8(:,2)' ; mea_acc_8(:,2)'];
mea_dis_9 = xlsread('position_X_97m_100sec.xlsx');
mea_acc_9 = xlsread('acceleration_X_97m_100sec.xlsx');
mea(:,:,9) = [mea_dis_9(:,2)' ; mea_acc_9(:,2)'];
mea_dis_10 = xlsread('position_X_107m_100sec.xlsx');
mea_acc_10 = xlsread('acceleration_X_107m_100sec.xlsx');
mea(:,:,10) = [mea_dis_10(:,2)' ; mea_acc_10(:,2)'];

alt_d = [9 20 31 42 53 64 75 86 97 108];
alt_a = [9 20 31 42 53 64 75 86 97 108];

errE = 0.05; %errpr on young modulus
```

```
38 errrho = 0.01; %error on volumic mass
39 errc = 0.1: %error on damping
40
41 opt_mat = zeros(length(alt_d),length(alt_a));
42
43 for i=1:length(alt_d)
44     for j=1:length(alt_a)
45         measure = [mea(1,:,i+9) ; mea(2,:,j+2)];
46         opt_mat(i,j) = Joint_input_state_beam_err(errE,errrho,errc,
               measure,alt_d(i),alt_a(j));
47     end
48 end
```

The different excel files contains position and acceleration data for different sensor heigts. And function `Joint_input_state_beam_err` is the main algorithm presented in Appendix B written in a function with $\mathbf{P_{de}}$ for output.

For some sensor configuration, numerical error are too high so the final optimization matrix is computed part by part.