Titre: Title:	The Pipe Conveying Fluid as a Model Experiment to Develop a Digital Twin
Auteur: Author:	Morgan Demenois
Date:	2022
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Demenois, M. (2022). The Pipe Conveying Fluid as a Model Experiment to Develop a Digital Twin [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/10458/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/10458/
Directeurs de recherche: Advisors:	Frederick Gosselin
Programme: Program:	Génie mécanique

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

The Pipe Conveying Fluid as a Model Experiment to Develop a Digital Twin

MORGAN DEMENOIS
Département de génie mécanique

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie mécanique

Août 2022

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

The Pipe Conveying Fluid as a Model Experiment to Develop a Digital Twin

présenté par **Morgan DEMENOIS**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Cédric BÉGUIN, président
Frédérick GOSSELIN, membre et directeur de recherche
Olivier DOARÉ, membre

DEDICATION

À ma grand mère, Chantal, qui a su éveiller ma curiosité depuis mon plus jeune âge, qui m'a accompagné dans de nombreuses visites culturelles et scientifiques, qui m'a donné le goût d'apprendre et m'a toujours poussé à donner le meilleur de moi même dans mes études …

ACKNOWLEDGEMENTS

I would first like to thank Frédérick Gosselin, my research supervisor for giving me the opportunity to conduct this project but also for his guidance, his support and his continuous positivity during my first steps in scientific research.

I also want to express my gratitude to the technicians from Polytechnique Montréal, more specifically Bénédict Besner, Fabrice Danet and Philippe Massé for their technical advice and their help during the experimental setup assembly. Another thank for the worker of Polytechnique, in particular André Francoeur and Albert Gaumond who helped me with the electrical installation of the setup and the walls and floors fixations.

My sincere acknowledgements to Hydro Quebec and Maya HTT for their financial and technical support and specially to Arnaud Divialle, Christian Semler, Barry Turner and Rémi Duquette for their technical and theoretical help.

Moreover, I want to thank the other members of the project with whom I worked closely Mohammed Abda, Sima Rishmawi, Hong Yan Miao, Vladislav Sushitskii and Souheil Serroud but also Lucas Berthet and Gaetan Raynaud, who always took time to listen to my problems and try to help me despite not working on the same project. My final thank goes to Louis Stefanuto, a summer intern who helped me solve computer vision problems.

Lastly, I would like to thank my parents Catherine and Frédéric who supported me and pushed me to do my best throughout my studies and are largely responsible for my success. I also want to thank my girlfriend Clementine and my friends from Quebec and France with whom I shared many rewarding and enjoyable experiences.

RÉSUMÉ

En 2015, les accords de Paris ont été adoptés par 196 pays s'engageant à limiter la hausse des températures. Les énergies renouvelables non pilotables prennent une place de plus en plus grande dans le mix électrique. L'énergie hydroélectrique prend alors un rôle central pour lisser et adapter la production à la demande. Il est donc d'autant plus important de surveiller les groupes turbines-alternateurs, notamment car les cycles de démarrage répétés et les utilisations en sous régime augmentent le risque de ruptures et de dommages notamment par fatigue. Dans ce projet, on se propose de travailler sur le développement d'un jumeau numérique qui, à terme, permettrait d'améliorer la surveillance, la prédiction des maintenances et d'optimiser l'opération des turbines hydrauliques.

On souhaite commencer notre travail sur les jumeaux numériques en travaillant sur une expérience de tuyau de pompier. En effet, le tuyau de pompier est une expérience d'interaction fluide-structure qui présente plusieurs similarités avec les phénomènes observés dans une turbine hydraulique. L'amortissement du tuyau est influencé par la masse ajoutée de l'eau dans le tuyau, sa vibration est également excitée par l'écoulement de l'eau et le tuyau subit des phénomènes de flottement comme la turbine. De plus, il s'agit d'un système simple et peu coûteux à reproduire et à observer en laboratoire. C'est toutefois un système riche qui démontre une physique riche et variée avec des fortes non-linéarités mais aussi des phénomènes très différents lorsqu'on modifie légèrement l'expérience. Enfin, c'est un système qui a été étudié en profondeur dans les dernières décennies et on dispose donc de nombreux modèles théoriques utilisables pour construire le jumeau numérique.

Durant ce projet, nous avons conçu, fabriqué et assemblé un montage expérimental de tuyau encastré-libre. Le débit parcourant le tuyau peut être réglé avec le contrôleur d'une pompe centrifuge et est mesuré par un débitmètre volumique. Le tuyau est filmé par deux caméras hautes vitesses placées dans des plans orthogonaux. Le tuyau est isolé de l'image en utilisant le contraste sur les images et sa position est ensuite approximée à l'aide d'un polynôme qui permet de stocker les informations. Le montage expérimental permet d'utiliser divers tuyaux de longueurs et diamètres intérieurs différents. Les tuyaux en caoutchouc siliconé sont moulés dans un moule usiné en aluminium et leurs caractéristiques sont déterminées en observant la vibration du mode fondamental du tuyau.

Nous avons mené une première campagne d'essais sur sept tuyaux différents avec trois diamètres différents. Nous avons étudié l'amortissement du tuyau avant la vitesse critique et avons pu la comparer au modèle linéaire du tuyau. Nous avons également déterminé et mesuré les vitesses critiques auxquelles le tuyau devient instable. Au dessus de la vitesse critique nous avons observé la relation entre l'amplitude de vibration ainsi que la fréquence de vibration avec le débit d'eau. Enfin nous avons également observé les différents phénomènes et modes vibratoires comme le mode chaotique à haut débit qui remplace les vibration planaires.

Dans une seconde partie du projet, nous avons expérimenté avec une nouvelle méthode pour construire le jumeau numérique du tuyau: des réseaux de neurones artificiels combinés au modèle physique. Cette technique utilise des réseaux de neurones pour approximer le mouvement du tuyau. Pour entraîner le jumeau numérique on utilise une fonction objectif à minimiser qui est constituée des données mesurées sur le montage expérimental (ou générées numériquement) mais aussi de l'équation linéaire du tuyau pénalisée à un certain nombre de points. Cette méthode permet d'utiliser les données de capteurs tout en régularisant la solution entre les données, en réduisant la quantité de données nécessaires et en permettant l'extrapolation. Dans ce mémoire on se limite à l'équation linéaire du tuyau et donc au comportement avant le débit critique. On utilise la formulation classique mais aussi une formulation avec décomposition modale pour une meilleure efficacité. On teste également l'implémention de la formulation faible des résidus de l'équation linéaire couplée avec une décomposition en modes propres de poutre. Enfin on montre que cette méthode permet de résoudre des problèmes inverses en définissant des paramètres du modèle linéaire comme des variables qui doivent être optimisées pendant l'entraînement du réseau de neurones.

Nous avons démontré la capacité des réseaux de neurones à résoudre un problème inverse en déterminant la valeur du débit coulant dans le tuyau à partir des données expérimentales et du modèle linéaire. Nous avons également montré que l'ajout du modèle linéaire permet de régulariser la solution entre les points de données mais aussi d'extrapoler.

Ce mémoire est la première étape du projet visant à développer un jumeau numérique d'un système de tuyau de pompier. Le montage expérimental servira pour la suite du projet pour entraîner, tester et valider les prochaines méthodes numériques et les prochains jumeaux numériques. Le montage et notamment les conditions aux limites et le tuyau seront amenés à évoluer pour étudier des phénomènes et des configurations différentes. Les modèles de jumeaux numériques devront également être améliorés en prenant en compte le modèle non-linéaire du tuyau, en utilisant une architecture de réseau de neurones différente ou encore une formulation différente des résidus. Enfin, un travail sur la convergence de l'optimisation du réseau de neurones devra être conduit.

ABSTRACT

Wind energy and solar energy production increased by 240% and 1200% between 2010 and 2018 in Canada. While these energies release 99% less CO_2 per gigawatt-hour than gas plants, they are prone to strong production irregularities. Hydraulic infrastructures smooth energy supply intermittency and serve as production reserves. For this new role, turbines are used in off-design conditions and subjected to more start and stop cycles which cause more risks of failures notably by fatigue while leaving less margin for maintenance. The turbines vibration frequencies, influenced by the water added damping are key to model fatigue evolution and are hard to compute numerically. Sensors measure the frequency, the vibrations, the strain or the pressure during the exploitation of hydraulic turbines. However, classic use of these data does not allow to predict maintenance accurately or to detect and predict some phenomena such as fatigue evolution. Digital twins are being developed for the monitoring, maintenance planning and operation optimization of all kinds of industrial systems such as jet engines, airframes, pumps... We propose to exploit sensors data with machine learning methods combining these data with physical prior knowledge to build a Digital Twin of the hydraulic infrastructures. The proof of concept of Digital Twins is based on the dynamic of a Pipe Conveying Fluid. Building a pipe conveying fluid experiment is cheap and easy. Yet this system exhibits linear, non-linear and chaotic dynamics. We designed and built a cantilevered pipe experimental setup and tested this setup by observing the behaviour of the pipe in different conditions and by measuring interesting characteristics of the pipe dynamics. A flowmeter measures the flow velocity while two orthogonal high speed cameras film the pipe to obtain training and validation data for the digital twin. We investigate Physics Informed Neural Networks (PINNs) to build the digital twin by combining sensor data and physical models. The data from the setup are used during training along with prior knowledge coming from a partial differential equation to optimize the weights and biases of a deep neural network. Data from the experimental setup can modify and adapt the model to fit the actual pipe and thus solve inverse problems. The formulation of the residuals from the physical model is a key element for the PINNs convergence and we investigate different methods such as the strong formulation, modal decomposition or the weak formulation. We demonstrate that PINNs can determine the water added mass damping on a pipe conveying fluid. This experimental setup and the framework we established will also serve as a testbench for testing and validating reduced-order models and other digital twin techniques.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

Acronyms

NN	Neural Network
DNN	Deep Neural Network
CNN	Convolution Neural Network
PINN	Physics Informed Neural Network
PDE	Partial Differential Equation
FSI	Fluid Structure Interactions
IPCC	Intergovernmental Panel on Climate Change
VIV	Vortex Induced Vibrations
MAC	Maximum Assurance Criterion
FFT	Fast Fourier Transform
RAM	Ramdom Access Memory
SSD	Solid State Drive
HP	Horse Power
PLA	Polyactic Plastic

Dimensional Pipe

E^*	The Kevin-Voigt damping parameter of the pipe
E	The Young modulus of the pipe material
I	The pipe quadrature moment
d	The pipe inner diameter
D	The pipe outer diameter
L	The pipe lenght
U	The flowrate
g	The gravity parameter
z	The vertical coordinate of the pipe
s	The Lagrangian coordinate along the pipe
x	The horizontal deflection of the pipe on the x axis
y	The horizontal deflection of the pipe on the y axis
M	The linear mass of the water in the pipe
m	The linear mass of the pipe
t	The time coordinate

δ_i	Logarithmic decrement of the i^{th} mode
f_i	Frequency of the pipe $i^{th} mode$

Non-dimensional Pipe

α	Non-dimensional damping parameter
u	Non-dimensional flowrate
β	Non-dimensional mass parameter
γ	Non-dimensional gravity parameter
η	Non-dimensional deflection
ξ	Non-dimensional vertical coordinate
τ	Non-dimensional time

Beam

λ_i	i^{th} beam eigenfrequency
ϕ_i	i^{th} beam mode shape
σ_i	Function used to construct the i^{th} beam mode shape

Pipe linear model solution

$\boldsymbol{M}$	Mass matrix of the system
$\boldsymbol{S}$	Damping matrix of the system
$\boldsymbol{K}$	Stiffness matrix of the system
$\boldsymbol{E}$	Shaker forcing matrix
$\boldsymbol{q}$	Time function of the decomposition
$\boldsymbol{Q}$	Time function vector
$\boldsymbol{F}$ and Δ	Eigenvalues diagonal matrices
$\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{D}$	Boundary conditions matrices
b_{ij}, c_{ij} and d_{ij}	Coefficients of the B, C and D matrices
$\boldsymbol{U}$ and $\boldsymbol{V}$	Reduced order matrices
$\boldsymbol{A}$ and λ	Eigenvectors and eigenvalues of the pipe
ω	Pipe complex frequency
ω_s	Shaker forcing frequency
η_s	Shaker forcing amplitude
θ	Phase angle of the shaker solution

PINN

Ω	Input domain of the NN
$\partial\Omega$	Boundary of the input domain
f	Function to approximate
f_{NN}	Approximation by the NN
f_{bc}	Boundary condition forcing function
$\boldsymbol{z}$	Input
$\boldsymbol{x}$	Output of the function
$\boldsymbol{x}_{NN}$	Output of the NN
σ	Activation function
$\mathscr{L}_d$	Data part of the loss
$\mathscr{L}_b$	Boundary part of the loss
$\mathscr{L}_s$	Strong form residuals part of the loss
$\mathscr{L}_w$	Weak form residuals part of the loss
$\mathscr{O}^p$	Linear differential operator of parameter p
h	Forcing term of the PDE
g	Boundary condition
N_{data}	Number of data points
N_s	Number of penalization points of the PDE
N_b	Number of boundary penalization points
N_k	Number of test functions for the weak form
$\boldsymbol{W}$	Weights of the NN
$\boldsymbol{b}$	Biases of the NN
v	Test function for the weak formulation
α_s	Weights coefficient from the loss
L_r	Learning rate

Image treatment

z_{tip_x}	Apparent z position of the tip on the front camera
z_{tip_z}	Apparent z position of the tip on the side camera
z_{x_a}	Apparent z position on the front camera
z_{x_c}	Corrected z position on the front camera
z_{y_a}	Apparent z position on the front camera
z_{y_c}	Corrected z position on the front camera
$z_{y_{camera}}$	Height distance between the camera and the top of the pipe

D_x	Distance between the calibration plan and the front camera
D_y	Distance between the calibration plan and the side camera
x_a	Apparent deflection on the x axis
x_c	Corrected deflection on the x axis
y_a	Apparent deflection on the y axis
y_c	Corrected deflection on the y axis
$f_x(z)$	Polynomial function to fit the x deflection
$f_y(z)$	Polynomial function to fit the y deflection

Mathematics

δ_{ij}	Kronecker symbol
$\|.\|$	Norm
$\mathbb{R}$	Real domain
j	Imaginary unit

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

In general aviation, planes are submitted to different types of maintenance every 50, 100 and 200 flight hours. Every 1000 flight hours an overhaul check is performed during which the whole plane is stripped down and every components are examined thoroughly. However, this rule is generic and does not take into consideration the plane history, the type of flights performed, the average take off weight, the type of runaway used... The decision to perform the maintenance is not based on observations, noise and vibrations measurements or calculations and physics-based simulations. Imagine the *Digital Twin* of a plane in a computer simulation that considers the physics that govern its behaviour and with sensors that feed it data in real-time. Such a digital twin could provide better maintenance planning, could help to optimize the operation of the plane and could also detect failures. All kinds of machines and systems could one day benefit from a digital twin, most notably large industrial equipments like airplanes, hydraulic turbines and nuclear power plants or large structures such as bridges and oil rigs.

We start the development of a digital twin with a simple yet rich experiment: the pipe conveying fluid. It is a flexible pipe fixed at one end with fluid flowing through it and is similar to a garden hose. This experiment can be installed in our university lab and allows us to have access to experimental data to develop and test different digital twin methods. The interesting and rich physics of the pipe conveying fluid makes it the perfect model experiment to work on different problems with the digital twins.

In the continuation of this introduction, we present the motivations to develop a Digital Twin and to use the pipe conveying fluid experiment as a model experiment. Lastly, we expose the organization of this master thesis.

1.1 Motivations

1.1.1 The hydraulic energy key role in the renewable energy development

In 2022, the Intergovernmental Panel on Climate Change (IPCC) published its last report, and claimed that emissions of CO_2 and other greenhouse effect gas must be reduced dramatically by 2030 to respect the goals from the 2015 Paris agreements: restrain global warming to a maximum of 2°C and take every actions to limit it to 1.5°C. The global warming has already reached 1.1°C since the 1850-1900 era and is caused without doubts entirely by human activity. Since electricity and heat production are responsible for 25% of greenhouse gas emissions, actions should be taken to reduce their emissions.

Renewable energies such as wind and solar energies could take over the production of electricity originally produced by fossil fuelled and coal power plants as they release 99% less CO_2 per gigawatt. In Canada, wind and solar energy growth is already well underway with respective increase in production capacity of 240% and 1200% from 2010 to 2018. However, these renewable energies are highly intermittent [2] and might endanger the grid stability [3]. The seasonal variations, the diurnal variations and the variations coming from passing weather systems of wind and solar energies have to be balanced with production reserves or with larger interconnected networks. Hydroelectric infrastructures also play a key role in the energy transformation as they represent 15.6% of the worldwide electricity production and 70% of the renewable electricity production. Not only producing electricity on a regular basis, hydroelectric infrastructures also act as production reserves to smooth the electricity production and balance renewable energies intermittence.

This new demand requires more performance from hydraulic turbines which are thinner and more lightweight. Alternator-turbines groups go through more start-stop cycles and load variations and are used in off-design conditions to compensate for renewable energy intermittence when it is needed. During these transients periods, the turbines are subjected to unsteady pressure loadings and asymmetric and cyclic forces that reduce the turbine lifespan [4, 5]. One of the main turbines failure mode is fatigue when a microscopic crack propagates at each loading cycle which leads to a large failure of the blades [6, 7]. This phenomenon is taken into account by evaluating the loading cycles and the response frequencies [8–10]. The water added mass and the flow-added damping influence these response frequencies and are difficult to compute numerically, thus causing the difficulties of the fatigue prediction [11,12]. Other damages on the turbine such as erosion or cavitation are hard to assess without visual inspections.

1.1.2 The emergence of the big data and of machine learning

It is of paramount importance to monitor hydraulic turbines closely to avoid unforeseen accidents. Most hydraulic turbines are equipped with many different sensors such as accelerometers, pressure sensors, strain gauges or proximity probes to detect hydraulic phenomena [13]. These sensors data are currently used to monitor the condition of the infrastructures in real time with classical methods [14]. However, it is hard to predict the evolution of the turbine state, to access its hidden characteristics or to detect some anomalies such as microscopic cracks on the blades.

For half a century, computational methods for solving physical problems have made considerable progress. Different methods such as finite elements, finite volumes, finite differences, mesh-free methods or spectral methods have been used to solve most linear and non-linear partial differential equations governing the physical phenomena. However, it is difficult to solve inverse problems to infer hidden characteristics of the system such as the material properties and the state of the turbines. Lastly, these methods fail to assimilate the data measured on most current industrial applications. Machine learning methods have generated a lot of attention during the last decade with the increase of data availability and with the presence of sensors on every physical systems. In the industrial field however, classical machine learning techniques are not yet commonplace for two main reasons. Firstly, on the contrary to big data applications, the data are not always available and are often expensive and complicated to obtain. Moreover, available data are frequently noisy and scattered. Another problem with machine learning in industrial applications is that these methods cannot infer the relations behind a physical phenomena.

To tackle these limitations, we work in this project, on the development of a digital twin. This digital twin shall provide information on the evolution of the system in time but also be able to infer its hidden characteristics and detect different phenomena. This new approach to sensor data allows the operator to better predict maintenance, spot defaults and avoid any detrimental operations. We decided to build the digital twin using a new machine learning method: Physics Informed Neural Networks (PINNs) [15, 16]. This new method combines sensor data with physical prior knowledge which regularize the solution between scatter data and help the model to extrapolate while keeping physical consistency. The basic idea is to use a Neural Network (NN) to approximate the physical phenomena of interest. Physical prior knowledge is softly encoded by penalizing the residuals of the phenomena partial differential equation (PDE) and the initial and boundary conditions in the loss function. The loss is also composed of the experimental data and is minimized during training to best fit the data as well as the physical model. The physical part of PINNs reduce the need for data, limit the impact of over-fitting, allows for better prediction and generalization. This method can also solve inverse problem by optimizing parameters in the PDE and works well with ill posed problems with unknown boundary or initial conditions for example. This method is of great interest for hydraulic infrastructures as it could work with in situ sensor data, would use the well known turbines physical models, could predict the behaviour of the system and also infer its properties and state by solving inverse problems.

1.1.3 The pipe conveying fluid as a model experiment

To achieve our goal, we first experiment on a classical fluid structure interaction (FSI) problem: The pipe conveying fluid. The pipe conveying fluid is a "paradigm" of linear and non-linear dynamics making it the perfect system to develop a new computational method [17]. The pipe conveying fluid is an experiment that can be easily reproduced in a university lab at relatively low costs. However, it is a very rich system that exhibits a very diverse range of behaviours, from linear planar movements to cyclic three dimensional movements and to chaotic three dimensional behaviours. With modern computer vision algorithms, it is now possible to reconstruct the pipe three dimensional shape by filming it with two cameras. This system is therefore easy to observe and to fully characterize with only two cameras and a flowmeter. This possibility to observe the pipe easily allows us to collect data to test digital twins methods with sets of training data and of validation data. The pipe behaviour is heavily influenced by boundary conditions and it is therefore possible to study dramatically different phenomena with minor modifications to the experimental setup. The pipe dynamic presents diverse interesting FSI phenomena such as water added mass, flutter, flow induced excitation or hydrodynamic damping that are observed in many real life physical applications including hydraulic turbines [18]. Moreover, this experiment has been thoroughly studied and many physical models that can be used in the PINNs models have been developed by physicists. This experiment also allows us to experiment with solving inverse problems to determine hidden parameters of the system from the observation and the sensors data. For example, we access the pipe added damping or the pipe parameters by solving the inverse problem with PINNs

1.2 Thesis goals and organization

Here we present the work accomplished to design, build and operate a pipe conveying fluid experimental setup. We explain the experimental process as well as the data treatment. We reproduce some interesting figures characterizing the behaviour of the pipe and we compare the experimental result below critical speed with the classical linear pipe model. Lastly, we introduce the use of PINNs to model the pipe behaviour in the linear mode with both numerical data and experimental data to access hidden parameters such as the flowrate. We also compare the effectiveness of modal and weak-form PINNs against classic PINNs while underlying the regularization effect of the PDE penalization.

Figure 1.1 presents an overview of the project philosophy. On the left side, the experimental setup is used to acquire data through the two cameras and the flowmeter. A computer is used to format the data and treat the data before feeding them in the digital twin model. At the bottom of the figure, the physical prior knowledge represented by the pipe linear PDE is used in the PINN model alongside experimental data to train the model. Once the PINN model is trained, it becomes the pipe digital twin on the left and can mimic its behaviour and estimate its hidden characteristics.

This master thesis is organized as follows: after this introduction we conduct a literature review in Chapter 2 on the PINNs and on the pipe conveying fluid experiment. Then, based on the literature review, we construct and present the objectives of this master thesis in Chapter 3. Afterwards, in Chapter 4, we present the methodology used to design, assemble and use the experimental setup and to acquire data. We continue by presenting the results obtained with the experimental setup in Chapter 5 and we explain the methodology used to build the PINNs models in Chapter 6. Finally, we present and discuss the PINNs results in Chapter 7 and we conclude in Chapter 8 by summarizing our work and discussing limitations and future work possibilities.

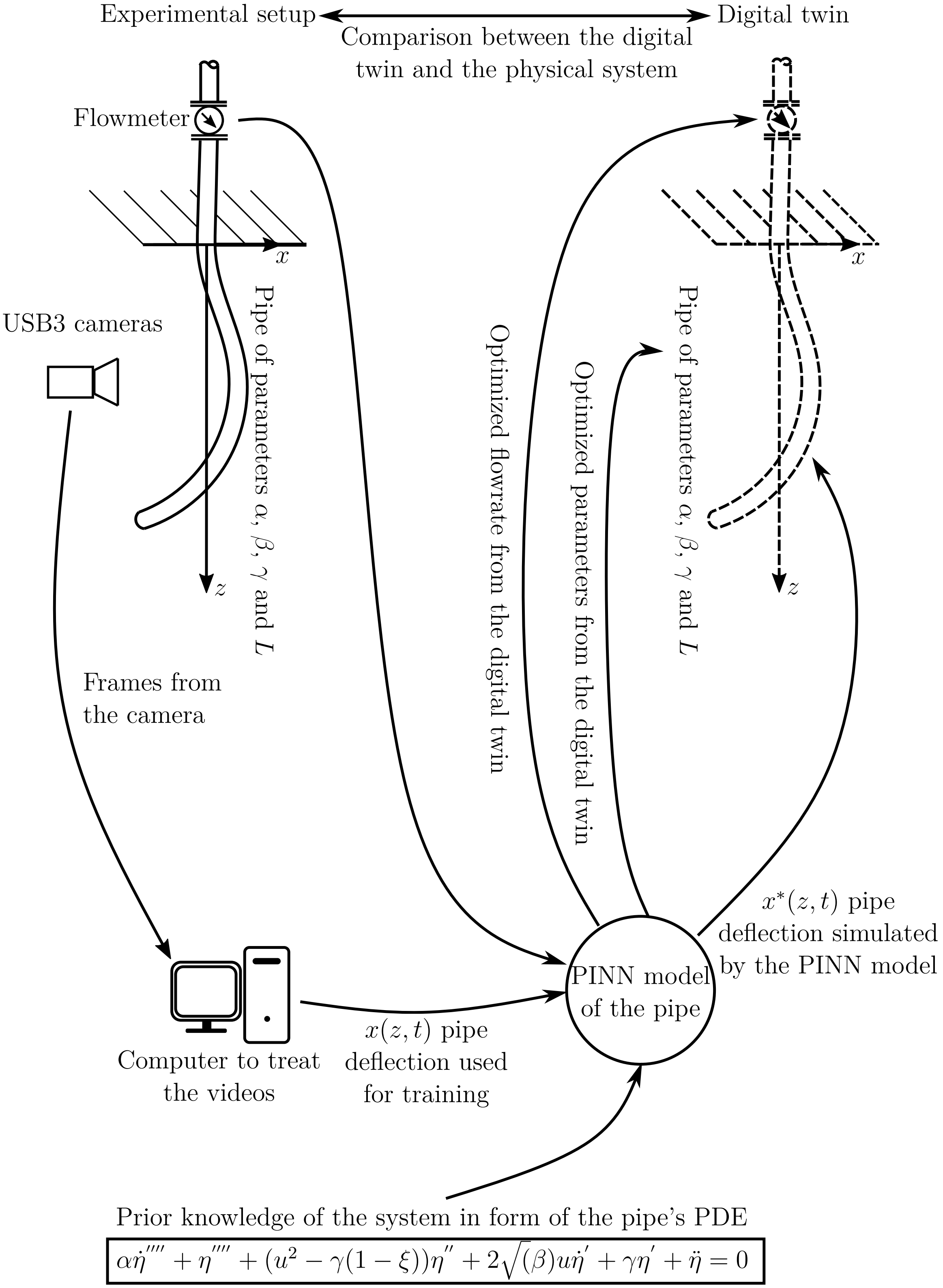

Experimental setup
Digital twin
Comparison between the digital twin and the physical system
Flowmeter
x
z
Pipe of parameters α, β, γ and L
USB3 cameras
Frames from the camera
Optimized flowrate from the digital twin
Optimized parameters from the digital twin
x
z
Pipe of parameters α, β, γ and L
$x^*(z,t)$ pipe deflection simulated by the PINN model
PINN model of the pipe
Computer to treat the videos
$x(z,t)$ pipe deflection used for training
Prior knowledge of the system in form of the pipe's PDE
$\alpha\dot{\eta}'''' + \eta'''' + (u^2 - \gamma(1-\xi))\eta'' + 2\sqrt{(\beta)}u\dot{\eta}' + \gamma\eta' + \ddot{\eta} = 0$

Figure 1.1 Schematic flowchart of a digital twin for a pipe conveying fluid system

CHAPTER 2 LITERATURE REVIEW

2.1 Physics informed neural networks

Machine learning methods and notably neural networks have made tremendous progress in diverse scientific fields such as image recognition [19], language processing [20] or autonomous driving [21]. However, neural networks perform well in large data conditions, data which might turn out to be expensive or complex to acquire in real world industrial applications. Moreover, the amount of data needed to keep the same data density over the domain increases exponentially with the problem dimension rendering impossible the use of neural networks on high dimensional problems. On the other hand, decades of scientific research have allowed physicists to develop models describing most physical phenomena through partial differential equations, intregro-differential equations, boundary conditions or physical laws such as the conservation principle. Using machine learning methods without considering these laws is a waste of knowledge.

Physics informed neural networks combine data with the prior knowledge cited before [22]. The prior knowledge helps regularize the solution between sparse data points, reduces the need for data and the over-fitting while allowing to solve inverse problem and to extrapolate beyond the available data while respecting physical laws [15, 16].

PINNs are already used with success in many different fields such as fluid mechanics to solve the Navier Stokes equations [23], in thermal to solve diffusion and heat transfer problems [24] or in structures to study fractures phenomena [25]. PINNs can also solve FSI problems such as vortex induced vibrations (VIV) on cylinders [26, 27].

2.1.1 Deep neural networks

A PINN based digital twin uses a neural network acting as a universal non-linear approximator that can approximate any function [28].

Figure 2.1 describes the architecture of a classic fully connected, feedforward, neural network. The NN is composed of one input layer of dimensions L (two in the figure), D hidden layers (depth)(two in the figure) composed of W neurons (width) and of one output layer of dimension R (one in the figure). For an input $\boldsymbol{z} \in \Omega \subset \mathbb{R}^D$, where Ω is the physical or input domain, we have the function representing the physical phenomenon to be approximated $f(\boldsymbol{z}) = \boldsymbol{x} : \Omega \to f(\Omega) = \mathbb{R}^R$ and we note $f_{NN}(\boldsymbol{z}) = \boldsymbol{x}_{NN}$ the approximation from the NN with respect to the chosen $\|.\|$ norm. As illustrated in Figure 2.1, each neuron of the hidden layers of a fully connected NN is connected with all neurons of the previous layer (as input) and each layer of the following layer (as output). The output of each neuron is computed from the input, the weight and the biases with a non-linear activation function such as ReLU, sigmoid or tanh. Figure 2.2 illustrates how each neuron acts on the input with the following relation between the output and the inputs of the neurons:

$$x = \sigma(W_1 z_1 + W_2 z_2 + W_3 z_3 + b), \tag{2.1}$$

where σ is the chosen activation function. Each neurons perform this operation to compute the value of the output $\boldsymbol{x}$ from the input $\boldsymbol{z}$.

We write the output of the neural network $\boldsymbol{x}_{NN}$ as $f_{NN}(z, \boldsymbol{W}, \boldsymbol{b})$, where $\boldsymbol{W}$ are the weights matrices and $\boldsymbol{b}$ are the biases vectors. The optimal approximation $f^*_{NN}(\boldsymbol{z})$ is obtained with the values of $\boldsymbol{W}^*$ and $\boldsymbol{b}^*$ which are the solutions of the following optimization problem:

$$(\boldsymbol{W}, \boldsymbol{b})^* = argmin_{\boldsymbol{W},\boldsymbol{b}}(\|f(\boldsymbol{z}) - f_{NN}(\boldsymbol{z}, \boldsymbol{W}, \boldsymbol{b})\|). \tag{2.2}$$

This optimization problem is solved during what is called the training phase of the neural network. Once the NN is trained and the optimal weights and biases are known, the NN computes the approximation of the function for any given input in the training range. To train the NN we select a set of N_{data} inputs $\boldsymbol{z}^{data} \in \Omega$ and the set of corresponding known output from the function to be approximated: $\boldsymbol{x}^{data}$. The optimization problem can be formulated:

$$(\boldsymbol{W}, \boldsymbol{b})^* = argmin_{\boldsymbol{W},\boldsymbol{b}} \sum_{i=1}^{N_{data}} (\|\boldsymbol{x}_i^{data} - f_{NN}(\boldsymbol{z}_i^{data}, \boldsymbol{W}, \boldsymbol{b})\|), \tag{2.3}$$

where the sum is the term to be minimized and is called the loss function $\mathscr{L}_d$. We use the subscript d on this loss term as it is corresponding to the data and is a part of a larger loss function as we will see later. With the L2 norm, this loss term has the following expression:

$$\mathscr{L}_d = \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} (\boldsymbol{x}_i^{data} - f_{NN}(\boldsymbol{z}_i^{data}, \boldsymbol{W}, \boldsymbol{b}))^2. \tag{2.4}$$

To solve this problem, we traditionally use gradient descent optimization algorithms such as Adam [29] or quasi Newton algorithms like L-BFGS-B [30]. These algorithms implement automatic differentiation [31] and the chain rule to compute the loss derivatives with respect

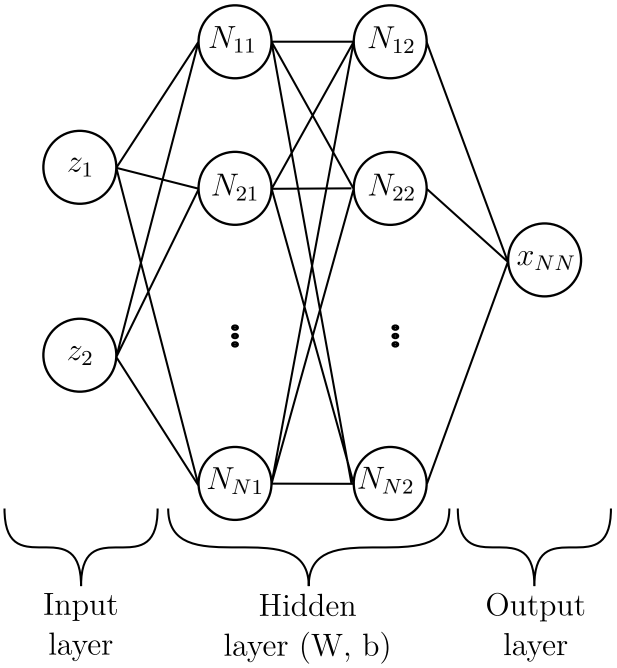


Figure 2.1 Global shape of a NN with two inputs, one output and with a N neurons width and a two layers length.

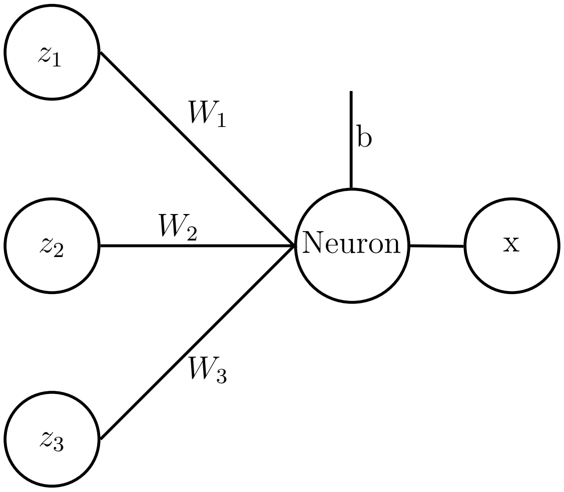


Figure 2.2 Focus on one neuron of the NN with three inputs and one output.

to the weights and biases parameters of the NN as follows:

$$\frac{\partial \mathscr{L}_d}{\partial \theta} = \frac{2}{N_{data}} \sum_{i=1}^{N_{data}} (\boldsymbol{x}_i^{data} - f_{NN}(\boldsymbol{z}_i^{data}, \boldsymbol{W}, \boldsymbol{b})) \times \frac{\partial f_{NN}(\boldsymbol{z}_i^{data}, \boldsymbol{W}, \boldsymbol{b})}{\partial \theta}, \tag{2.5}$$

where θ is the hyper parameter (weight or bias) with respect to which the loss function is derived. The value of:

$$\frac{\partial f_{NN}(\boldsymbol{z}_i^{data}, \boldsymbol{W}, \boldsymbol{b})}{\partial \theta}, \tag{2.6}$$

is computed with the activation function derivatives and with the chain rule by going through the neural network backward until the first layer is reached which is commonly called the back propagation step.

Some more advanced neural networks use adaptive activation functions to accelerate convergence or adaptive learning rate in the optimization algorithms to avoid getting stuck in local minima at the beginning and oscillations at the end of the training [32].

The "physics informed" part that we mentioned previously can be implemented by two different methods: soft implementation and hard implementation.

2.1.2 Soft implementation of the physics prior knowledge

Most physical problems are governed or at least constrained by a set of partial differential equations or integro-differential equations with boundary conditions. The following equation:

$$\mathscr{O}^{\boldsymbol{p}} f(\boldsymbol{z}) = h(\boldsymbol{z}), \tag{2.7}$$

represents the partial differential equation where $\boldsymbol{x} = f(\boldsymbol{z})$ is the vector representing the quantity of interest, $\boldsymbol{z} \in \Omega$ is the vector representing the physical variable over the physical domain, $h(\boldsymbol{z})$ is a forcing term and $\mathscr{O}$ is an integro-differential operator with scalar parameters $\boldsymbol{p}$.

The boundary condition is equally represented as follows:

$$f(\boldsymbol{z}) = g(\boldsymbol{z}), \tag{2.8}$$

where $\boldsymbol{z} \in \partial\Omega$ is the physical variable over the physical boundary.

This physical prior knowledge can be enforced in the NN through a soft penalty constraint during the optimization process. To do so we add two terms to the loss function $\mathscr{L} = \alpha_d \mathscr{L}_d + \alpha_b \mathscr{L}_b + \alpha_s \mathscr{L}_s$ where:

$$\mathscr{L}_s = \frac{1}{N_s} \sum_{i=1}^{N_s} (\mathscr{O}^{\boldsymbol{p}} f_{NN}(\boldsymbol{z}_i) - h(\boldsymbol{z}_i))^2, \tag{2.9}$$

is a loss based on the residuals of the partial differential equation and:

$$\mathscr{L}_b = \frac{1}{N_b} \sum_{i=1}^{N_b} (f_{NN}(\boldsymbol{z}_i) - g(\boldsymbol{z}_i))^2, \tag{2.10}$$

is based on the boundary conditions and where α_d, α_s and α_b are weights that can be applied to the different terms of the loss function.

The boundary conditions and the partial differential equation can therefore be penalized on large sets of points N_b and N_s. With this addition, the NN regularizes the solution between data points while respecting physical rules thus reducing the risk of over-fitting and the need for data during training. By including points outside the range of data in N_b and N_s, it is also possible for PINNs to extrapolate outside of the measurements. We use the subscript s for the loss term corresponding to the differential equation since we used a strong formulation for this term. The loss function also allows to solve inverse problem as the parameters $\boldsymbol{p}$ from the linear operator can be unknown and set as variables to be optimized during the training.

Once the training is completed, these parameters have the value of the actual corresponding physical parameters.

To improve the PINN performance it is possible to use the weak or variational formulation of the loss associated to the PDE [33, 34]. To do so, we multiply Equation (2.7) by a test function v_k that is defined on the Ω domain for $k = 1, 2, ..., N_k$ and we integrate on Ω thus reaching the following formulation:

$$\int_{\Omega} \mathscr{O}^{\boldsymbol{p}} f(\boldsymbol{z}) v_k(\boldsymbol{z}) d\Omega = \int_{\Omega} h(\boldsymbol{z}) v_k(\boldsymbol{z}) d\Omega. \tag{2.11}$$

The part of the loss function associated to the PDE can therefore be rewritten as:

$$\mathscr{L}_w = \frac{1}{N_k} \sum_{i=1}^{N_k} \left(\int_{\Omega} \mathscr{O}^{\boldsymbol{p}} f_{NN}(\boldsymbol{z}) v_k(\boldsymbol{z}) d\Omega - \int_{\Omega} h(\boldsymbol{z}) v_k(\boldsymbol{z}) d\Omega \right)^2. \tag{2.12}$$

Because of the neural network structure it is impossible to express analytically the integrals in this loss term if the NN is more than one layer deep. The integrals have to be computed numerically by using, for example, the quadrature rule.

This formulation has many advantages over the strong formulation. The order of the derivatives in the differential operator is reduced through integration by part, thus reducing the computational complexity and computational errors coming from high order derivatives combined with irregular solutions from the NN. Moreover, the quadrature rule used to estimate the integrals requires less penalization points compared to the strong form that has to be penalized evenly on the domain.

This approach can be combined with domain decomposition by dividing the domain Ω in sub domains $\Omega_1, \Omega_2, ..., \Omega_N$ and by using different test functions on all the domains [35]. This performs better for cases where strong local irregularities are present in the solution.

The loss function is a high dimension non-convex function, so it is complicated to minimize it during the training process to get the PINN hyper-parameters. In addition to adaptive learning rate as mentioned before, it is also possible to use adaptive weights on the loss function terms to improve the convergence of the minimization [36].

2.1.3 Hard implementation of the physics prior knowledge

Another possibility to incorporate prior knowledge into the NN is to work on the NN structure itself to force the solution to respect the physical model. This approach can and should be implemented in parallel with the precedent approach.

One of the first solution is to encode hard boundary conditions with prior dictionaries [37]. The idea is to multiply the output of the NN by a function that always respects the boundary conditions:

$$\boldsymbol{x}_{NN} = f_{NN}(\boldsymbol{z}, \boldsymbol{W}, \boldsymbol{b}) \times f_{bc}(\boldsymbol{z}), \tag{2.13}$$

where f_{bc} is the function forcing the boundary condition. For example, for a cantilevered beam, we use the $f_{bc}(\boldsymbol{z}) = \tanh(\boldsymbol{z})^2$ since this function and its first derivative are equal to zero for $\boldsymbol{z} = 0$ thus assuring that the deflection and the slope are equals to zero on the clamped end of the beam. This approach may however prove more complex for boundary condition on higher order derivatives.

Another approach is to work directly on the architecture of the NN to force it to respect some of the characteristics of the approximated function. A NN can be forced to respect reflection or rotation symmetries or translation and rotation invariances [15]. The most common NN architecture adapted to fit the application is the convolution NN for computer vision applications [38].

Finally, it is possible to use the prior knowledge on the shape of the solution. If a dynamic phenomenon is known to be following a sinusoidal law, we use modal PINNs [39]. The idea is to assume that the solution has the following form:

$$x = \sum_{k=1}^{N_{mode}} \phi_k(z) \exp(j\omega_k t), \tag{2.14}$$

where ω_k are the system eigenfrequencies and ϕ_i are the unknown spatial mode shapes. In this particular example, z is a scalar in the spatial domain and t is a scalar in the time domain, while x is also a scalar which represents the physical phenomena.

With this formulation, the NN takes only the space dimension as input and not the time dimension and has the N spatial modes as output:

$$f_{NN} : x \rightarrow (\phi_1^{NN}(z), \phi_2^{NN}(z), ..., \phi_N^{NN}(z)). \tag{2.15}$$

The penalization of the three terms of the loss function is still performed on the values of x after multiplication with the time functions and the summation. We can therefore write the three terms of the loss as:

$$\mathscr{L}_d = \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} \left(x_i^{data} - \sum_{k=1}^{N_{mode}} \phi_k^{NN}(z_i^{data}) \exp(j\omega_k t_i^{data}) \right)^2, \tag{2.16}$$

for the data part and:

$$\mathscr{L}_s = \frac{1}{N_s} \sum_{i=1}^{N_s} \left(\mathscr{O}^{\boldsymbol{p}} \sum_{k=1}^{N_{mode}} \phi_k^{NN}(z_i^s) \exp\left(j\omega_k t_i^s\right) - h(z_i^s) \right)^2 , \tag{2.17}$$

for the partial differential equation and:

$$\mathscr{L}_b = \frac{1}{N_b} \sum_{i=1}^{N_b} \left(\sum_{k=1}^{N_{mode}} \phi_k^{NN}(z_i^b) \exp\left(j\omega_k t_i^b\right) - g(z_i^b) \right)^2 , \tag{2.18}$$

for the boundary conditions.

2.2 The pipe conveying fluid

Flow induced vibrations in a pipe are observed in many physical and industrial applications such as risers in oil rigs or pipelines which motivated the study of the pipe conveying fluid dynamics over the years. The pipe conveying fluid is a system that has rapidly become a "paradigm of dynamics" and has been studied extensively since the middle of the 20th century. More than hundreds of articles presenting experimental or theoretical studies of the pipe have been published.

2.2.1 Overview

Early studies are mainly focused on linear models of the pipe that are mostly valid before the destabilization of the pipe movement and before the critical flow speed [40–42]. These studies mainly focus on two dimensional problems and are meant to determine the critical flowrate, to plot Argand diagrams presenting the eigenfrequencies along the flow velocity or also to study the pipe behaviour just after the critical speed. Some interesting phenomena are also observed such as the evolution of the pipe damping with the evolution of the flowrate because of the water added damping or the evolution of the critical flowrate with respect to the pipe dimensions [43].

More recent studies model the dynamics of the pipe beyond critical speed and dive into non-linear behaviour. The earliest of these studies constrain the pipe to a bi-dimensional movement [44–46] to facilitate measurements while more recent studies [47, 48] are, for most of them, studying the pipe dynamics in three dimensions.

The pipe was also studied in different configurations. The most studied case is the cantilevered pipe conveying fluid. In this case, the pipe is a non-conservative system that loses stability by flutter. At higher flowrate, the planar cyclic flutter becomes a three dimensions orbital or rotary flutter and the movement becomes chaotic at even higher flowrate. Particular pipes have also been studied in the literature such as long hanging pipes [49]. Pipe with an additional point mass were also studied and it was shown that this mass either destabilizes [50] the pipe or stabilizes [51] the pipe depending on its position and mass. Pipes with spring supports, either at the tip of the pipe or higher on the pipe were also considered [52, 53]. It was shown that depending on the springs stiffness and position, the pipe exhibits very rich and different dynamics.

Pipe clamped at both ends were also thoroughly examined and have a similar behaviour to pipes clamped at one end and supported at the other end [54, 55]. In these configurations, the pipe loses stability by buckling which leads to a static instability. It should be noted that the addition of supported boundary conditions has a tendency to destabilize the pipe. Subjected to a flowrate slightly under the critical flow speed, a cantilevered pipe buckles upon the addition of a support at its free end which is counter intuitive. With these boundary conditions, the pipe is a conservative problem as opposed to the cantilevered case.

The experimental setup conceived in this study shall allow the operator to easily study all the previous cases with only slight modifications. Many other interesting configurations are not studied in this project. We can, for example cite the cases of pipes conveying two-phase flows [56], horizontal pipes, pipes conveying pulsating flows [57] or pipes with external flows [58].

From this literature review, it is clear that the pipe conveying fluid is a perfect model experiment to develop a digital twin. Indeed, it is a simple system that is fairly easy to reproduce in a lab. However, despite its simplicity, it is a rich experiment that exhibits various behaviour on a large range of physical phenomena. On the first hand, the physics of the pipe can go from a linear behaviour with planar motions to three dimensions chaotic motions with strong non-linearities by just changing the flowrate in the pipe with no modifications to the setup. Then, slight modifications on the setup can also dramatically influence the movements of the pipe. For example, the non-linearities can be increased by adding an end mass on the pipe. One can also evaluate the impact of using different boundary conditions on the pipe physics as with springs for example. It is why, this experiment was chosen as it allows us to work on a large variety of phenomena with the digital twin to assess its effectiveness in different possibilities.

In addition, the pipe conveying fluid presents many different quantities of interest that are measured to train the digital twin. It is possible to measure the bending of pipe, its position, its acceleration as well as its elongation for example. This possibility to work with different types of data is of prime importance for our work as it enables us to assess which kind of data is the most effective to train the digital twin. It would then be an advantage to know which data to use on a real life system such as hydraulic turbines [13].

Finally, the pipe conveying fluid is an experiment with many hidden parameters that cannot be determined easily with classical methods. The pipe material such as its Young modulus and visco-elastic parameters, the fixation parameters that can be modelled with springs of a certain stiffness or even the water flowrate. These hidden parameters offer a great opportunity to test the ability of PINNs to evaluate these parameters by using the sensors data and the physical model on top of simulating the pipe motion.

2.2.2 Linear model

We have decided to develop the pipe digital twin with PINNs. As explained before, PINNs combine data from sensors and physical prior knowledge. As a first step toward the digital twin, we use the classic two-dimensional linear model from the cantilevered pipe conveying fluid. The associated PDE is derived and presented in reference [1]. This PDE was obtained through a Newton derivation which is detailed alongside the different simplifications in the book for interested readers. The material damping is modelled using the Kelvin-Voight visco-elastic model and we neglect friction from the fluid on the exterior of the pipe since it is air. Because the pipe is cantilevered and the pipe fixation is considered perfectly rigid, the tension, pressure and fixation flexibility terms are not considered. The dimensional equation can be written as follows:

$$\begin{aligned} E^{*}I\frac{\partial^5 x}{\partial t \partial z^4} + EI\frac{\partial^4 x}{\partial z^4} + \left\{MU^2 - g\left(M+m\right)\left(L-z\right)\right\}\frac{\partial^2 x}{\partial z^2} \\ + 2MU\frac{\partial^2 x}{\partial t \partial z} + g\left(M+m\right)\frac{\partial x}{\partial z} + \left(M+m\right)\frac{\partial^2 x}{\partial t^2} = 0, \end{aligned} \tag{2.19}$$

where the parameters are explained in Table 2.1.

In the previous pipe equation, we distinguish the following terms:

- $E^{*}I\frac{\partial^5 x}{\partial t \partial z^4}$ is the damping term that is modelled with a visco-elastic damping;
- $EI\frac{\partial^4 x}{\partial z^4}$ is the classic Euler-Bernoulli flexural rigidity term that we find in the cantilevered beam equation;
- $g(M+m)\frac{\partial x}{\partial z} - g(M+m)(L-z)\frac{\partial^2 x}{\partial z^2}$ is the term from the influence of gravity;
- $MU^2\frac{\partial^2 x}{\partial z^2}$ is the centrifugal force term;

Table 2.1 Dimensional parameters of the pipe equation of motion

Symbol	Signification
z	Vertical coordinate
t	Time
x	Pipe orthogonal deflection
I	Pipe quadratic moment
E	Pipe material Young modulus
E^*	Kelvin-Voight visco-elastic damping parameter
M	Linear mass of the fluid through the pipe
m	Linear mass of the pipe
L	Length of the pipe
U	Flow velocity

- $2MU\frac{\partial^2 x}{\partial t \partial z}$ is the Coriolis force term;
- $(M+m)\frac{\partial^2 x}{\partial t^2}$ is the classic inertial term that we find in the cantilevered beam equation.

This equation has the following non-dimensional form:

$$\alpha\dot{\eta}^{''''} + \eta^{''''} + \left(u^2 - \gamma\left(1-\xi\right)\right)\eta^{''} + 2\beta^{1/2}u\dot{\eta}^{'} + \gamma\eta^{'} + \ddot{\eta} = 0, \tag{2.20}$$

where the dimensionless parameters are defined in Table 2.2.

As with the dimensionnal equation we can identify the different terms:

- $\alpha\dot{\eta}^{''''}$ is the visco-elastic damping term;
- $\eta^{''''}$ is the flexural rigidity term;
- $u^2\eta^{''}$ is the centrifugal force term;
- $-\gamma\left(1-\xi\right)\eta^{''} + \gamma\eta^{'}$ is the term from the influence of gravity;
- $2\beta^{1/2}u\dot{\eta}^{'}$ is the Coriolis force term;
- $\ddot{\eta}$ is the inertial term.

Table 2.2 Parameters of the pipe adimensional PDE

Formulation	Signification
$\xi = \frac{z}{L}$	Adimensional vertical coordinates
$\eta = \frac{x}{L}$	Adimensional deflection of the pipe
$\tau = (\frac{EI}{M+m})^{\frac{1}{2}} \frac{t}{L^2}$	Adimensional time
$u = (\frac{M}{EI})^{\frac{1}{2}} LU$	Adimensional flowrate
$\beta = \frac{M}{M+m}$	Mass parameter
$\gamma = \frac{(M+m)L^3}{EI} g$	Gravity parameter
$\alpha = (\frac{I}{E(M+m)})^{\frac{1}{2}} \frac{E^*}{L^2}$	Damping parameter
$\dot{} = \frac{\partial}{\partial t}$	Time derivative
$' = \frac{\partial}{\partial z}$	Space derivative

This PDE is used as our model in the PINN. We have to solve this PDE in order to generate numerical data, to compare it with the experimental results and with the simulations from the PINN. We expect this PDE to be a valid model of the pipe for a flowrate from 0 to the critical speed. Above the critical speed, the behaviour of the pipe has to be described with the non-linear model in particular to account for the large deflection of the pipe.

To solve this PDE, we used the Galerkin method as detailed in reference [1]. The idea is to write an approximation of the solution as a summation of cantilevered beam eigenfunctions (ϕ_i) with time functions (q_i) as follows:

$$\eta(\xi, \tau) = \sum_{i=1}^{N} \phi_i(\xi) q_i(\tau). \tag{2.21}$$

The ϕ_i functions are well known and take the following shape:

$$\phi_i(\xi) = cosh(\lambda_i \xi) - cos(\lambda_i \xi) - \sigma_i(sinh(\lambda_i \xi) - sin(\lambda_i \xi)), \tag{2.22}$$

with λ_i the cantilevered beam eigenfrequencies verifying the following:

$$cos(\lambda_i) cosh(\lambda_i) + 1 = 0, \tag{2.23}$$

and with:

$$\sigma_i = \frac{sinh(\lambda_i) - sin(\lambda_i)}{cosh(\lambda_i) + cos(\lambda_i)}. \tag{2.24}$$

We inject the solution (2.21) into the Equation (2.20), multiply it by ϕ_j for $1 \leq j \leq N$ and integrate the resulting N equations between 0 and 1 on the ξ domain. We obtain the following equations for every $1 \leq j \leq N$:

$$\begin{aligned}\sum_{i=1}^{N} \Big\{ \alpha \int_0^1 \phi_i(\xi)^{''''} \phi_j(\xi) \dot{q}_i(\tau) d\xi + \int_0^1 \phi_i(\xi)^{''''} \phi_j(\xi) q_i(\tau) d\xi \\ + \Big(u^2 - \gamma\left(1-\xi\right)\Big) \int_0^1 \phi_i(\xi)^{''} \phi_j(\xi) q_i(\tau) d\xi + 2\beta^{1/2} u \int_0^1 \phi_i(\xi)^{'} \phi_j(\xi) \dot{q}_i(\tau) d\xi \\ + \gamma \int_0^1 \phi_i(\xi)^{'} \phi_j(\xi) q_i(\tau) d\xi + \int_0^1 \phi_i(\xi) \phi_j(\xi) \ddot{q}_i(\tau) d\xi \Big\} = 0.\end{aligned} \tag{2.25}$$

From the cantilevered beam equation, we obtain:

$$\phi_i(\xi)^{''''} = \lambda_i^4 \phi_i(\xi). \tag{2.26}$$

We also use the fact that the cantilevered beam eigenfunctions ϕ are orthonormal meaning:

$$\int_0^1 \phi_i(\xi) \phi_j(\xi) d\xi = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases}. \tag{2.27}$$

We therefore obtain the following equation:

$$\begin{aligned}\sum_{i=1}^{N} \Big\{ \delta_{ij} \ddot{q}_i + \Big[\Big(\alpha \lambda_i^4 + \sigma\Big) \delta_{ij} + 2\beta^{1/2} u \int_0^1 \phi_j \phi_i' \mathrm{d}\xi \Big] \dot{q}_i \\ + \Big[\Big(\lambda_i^4 + k\Big) \delta_{ij} + \Big(u^2 - \gamma\Big) \int_0^1 \phi_j \phi_i'' \mathrm{d}\xi \\ + \gamma \int_0^1 \phi_j \phi_i' \mathrm{d}\xi + \gamma \int_0^1 \phi_j \xi \phi_i'' \mathrm{d}\xi \Big] q_i \Big\} = 0, \quad s = 1, 2, \ldots N.\end{aligned} \tag{2.28}$$

With the boundary conditions of the cantilevered pipe, one can obtain a formula for the following terms which are presented in Table 2.3:

$$\int_0^1 \phi_i(\xi) \phi_j(\xi)^{'} d\xi = b_{ij}, \tag{2.29}$$

$$\int_0^1 \phi_i(\xi) \phi_j(\xi)^{''} d\xi = c_{ij}, \tag{2.30}$$

$$\int_0^1 \phi_i(\xi)\phi_j(\xi)''\xi d\xi = d_{ij}. \tag{2.31}$$

We therefore obtain the N following equations for $1 \leq j \leq N$:

$$\sum_{i=1}^{N} \Big\{ \delta_{ij}\ddot{q}_i + \Big[\alpha\lambda_i^4\delta_{ij} + 2\beta^{1/2}ub_{ij}\Big]\dot{q}_i + \Big[\lambda_i^4\delta_i j + (u^2-\gamma)c_{ij} + \gamma b_{ij} + \gamma d_{ij}\Big] q_i \Big\} = 0. \tag{2.32}$$

This equation can be written as a $(N \times N)$ matrix system:

$$\ddot{\boldsymbol{q}} + \Big[\boldsymbol{F} + 2\beta^{1/2}u\boldsymbol{B}\Big]\dot{\boldsymbol{q}} + \Big[\boldsymbol{\Delta} + \gamma\boldsymbol{B} + (u^2-\gamma)\boldsymbol{C} + \gamma\boldsymbol{D}\Big]\boldsymbol{q} = \boldsymbol{0}, \tag{2.33}$$

where the $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{D}$ matrices are $(N \times N)$ matrices respectively containing the values of b_{ij}, c_{ij} and d_{ij}. $\boldsymbol{F}$ and $\boldsymbol{\Delta}$ are diagonal matrices with their respective components being $\alpha\lambda_i^4$ and λ_i^4.

We transform this matrix equation to a classic second order differential matricial equation as follows:

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{S}\dot{\boldsymbol{q}} + \boldsymbol{K}\boldsymbol{q} = \boldsymbol{0}. \tag{2.34}$$

We then reduce this system to a first order system as follows:

$$\boldsymbol{Q} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{q} \end{bmatrix}, \; \boldsymbol{U} = \begin{bmatrix} \boldsymbol{M} & \boldsymbol{S} \\ \boldsymbol{0} & \boldsymbol{1} \end{bmatrix} \text{ and } \boldsymbol{V} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{K} \\ \boldsymbol{-1} & \boldsymbol{0} \end{bmatrix}. \tag{2.35}$$

And the equation becomes:

$$\boldsymbol{U}\dot{\boldsymbol{Q}} + \boldsymbol{V}\boldsymbol{Q} = \boldsymbol{0}. \tag{2.36}$$

Table 2.3 Formula for the integral terms of the PDE

	Case $i = j$	Case $i \neq j$
b_{ij}	2	$\frac{4}{(\frac{\lambda_i}{\lambda_j})^2+(-1)^{i+j}}$
c_{ij}	$\lambda_i\sigma_i(2-\lambda_i\sigma_i)$	$\frac{4(\lambda_i\sigma_i-\lambda_j\sigma_j)}{(-1)^{i+j}-(\frac{\lambda_j}{\lambda_i})^2}$
d_{ij}	$\frac{1}{2}c_{ii}$	$\frac{4(\lambda_i\sigma_i-\lambda_j\sigma_j+2)}{1-(\frac{\lambda_j}{\lambda_i})^4}(-1)^{i+j} - \frac{3+(\frac{\lambda_j}{\lambda_i})^4}{1-(\frac{\lambda_j}{\lambda_i})^4}b_{ij}$

We suppose $\boldsymbol{Q} = \boldsymbol{A}e^{\lambda\tau}$, and inject this solution into Equation (2.36), to get:

$$(\lambda \boldsymbol{U} + \boldsymbol{V})\boldsymbol{A} = \boldsymbol{0} \leftrightarrow \lambda \boldsymbol{A} = (-\boldsymbol{U}^{-1}\boldsymbol{V})\boldsymbol{A}, \tag{2.37}$$

where λ is an eigenvalue of $-\boldsymbol{U}^{-1}\boldsymbol{V}$ and $\boldsymbol{A}$ is an eigenvector. This system is solved as an eigenvalues problem with built in functions in Python for example.

It is then possible to compute the eigenfrequencies and eigenvectors of $\boldsymbol{q}$ for different values of u, β or γ. However, if one wants to study the evolution of these eigenfrequencies with the values of u for example, then it is necessary to track the modes when the input values change. Indeed, the Python function used to compute the eigenvalues and the eigenvectors returns them in random order. To plot the evolution of an eigenfrequencies, it is necessary to plot them for the same modes. Therefore, we used the Maximum Assurance Criterion (MAC) to order the output of the Python function according to the correct mode [59].

We applied the following method to sort the results with the correct modes for the evolution of u:

- At $u = 0$, we sort the modes from smallest to biggest eigenvalue real parts;
- At each $u + \Delta u$, we compute the MAC matrix of every modes from u with every modes from $u + \Delta u$;
- We associate the modes from $u + \Delta u$ with the modes from u with which they have the largest MAC;
- If the MAC is less than 0.9, we go back to the second step with $\Delta u_{new} = \Delta u/2$;

For all the following computations we used $N = 10$ as the number of beam modes as it is recognized to be enough in the literature. All the codes to solve the linear model and other codes used in the next part of this master thesis are available at reference [60] with the data.

The plots in Figures 2.3 and 2.4 reproduce results presented in reference [1].

We ploted Argand diagrams in Figure 2.3 which represent the imaginary part of the i^{th} non-dimensional frequency ω_i as a function of its real part for a range of non-dimensional u from 0 to 20. The non-dimensional frequency ω is defined as $\omega = -i\lambda$. The real part represents the frequency of the motion of the pipe while the imaginary part represents the damping of the motion. The higher the imaginary part is, the more stable the system is. On the contrary, when the imaginary part gets smaller, the system is getting less stable and unstable when the imaginary part is negative. The points placed on the curves are placed for every integers values of u starting at $u = 0$.

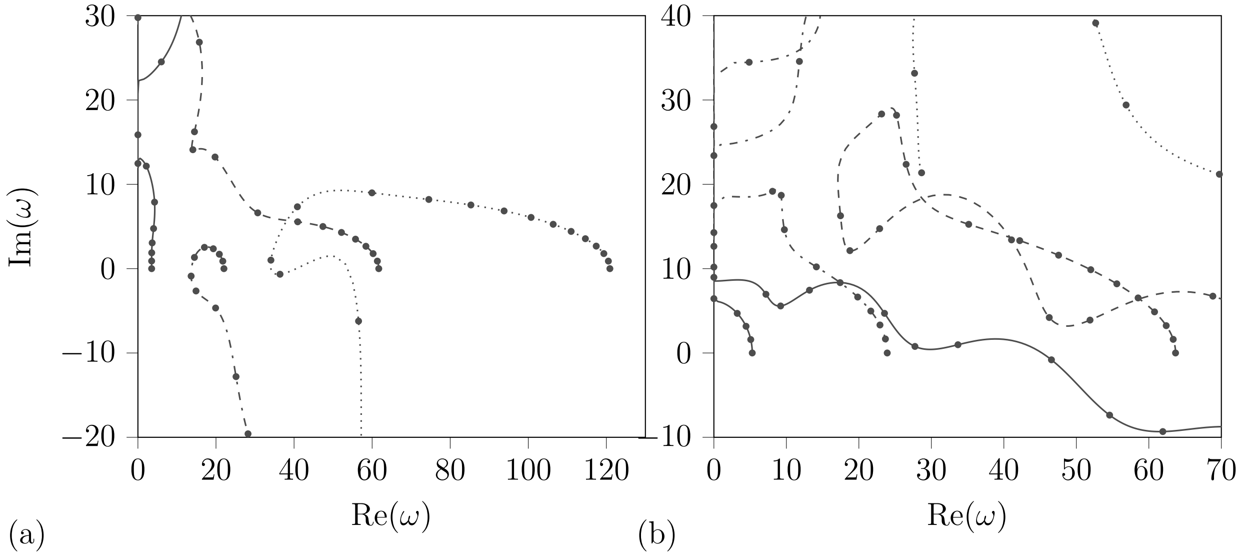


Figure 2.3 Argand diagrams reproduced with the linear model detailed in reference [1]: (a) for $\alpha = \gamma = 0$ and $\beta = 0.2$ and (b) for $\alpha = 0$, $\beta = 0.65$ and $\gamma = 10$. The points on the curves are placed at every integer values of u starting at $u = 0$. The first four modes are represented: (——) for first mode, (·-·-·) for second mode, (- - -) for third mode and (······) for fourth mode.

Figure 2.3(a) shows that the pipe destabilizes firstly through its second mode and then through its fourth mode. For the first and the third modes, we notice that the damping increases when the flowrate increases. For the second and fourth modes, the damping also increases before decreasing again and becoming negative with the flutter destabilization. We will observe experimentally this augmentation of the water added damping with the increase of the flowrate before the destabilization on the experimental setup.

Figure 2.3(b) presents the same phenomena: the pipe destabilizes first with the first mode and the damping also starts by increasing with the flow velocity.

With the same approach, we constructed a stability map in Figure 2.4(a) which shows the evolution of the critical flowrate at which the flutter appears on a range of mass parameters β from 0 to 1 and for four different values of gravity parameters γ.

To build this plot, we compute the first ten eigenvalues λ for a range of flowrate starting at $u = 0$. When at least one of the real part of the eigenvalues has a negative value, it means that the damping has became negative and that the pipe is unstable through flutter. The critical flowrate u is the lowest flowrate at which one of the eigenvalues real part becomes negative. For the gravity parameter $\gamma = -10$ we had to find the flow velocity at which the real part changes sign from positive to negative as some eigenvalues have a negative real value even at $u = 0$ which indicates a static instability and not flutter. Indeed, a negative gravity parameter represents an upside down pipe which would be unstable even without flow.

A larger gravity parameter induces a better stability with a higher critical speed. It is logical as a larger γ means that the gravity has a stronger effect on the pipe and acts to keep the pipe in its rest position. We will observe this phenomenon experimentally with the experimental setup with the pipes having different lengths. Furthermore, a larger mass parameter increases the stability of the pipe because it increases the importance of the Coriolis force in comparison to the centrifugal force. The Coriolis force absorbs energy and stabilizes the pipe while the centrifugal force destabilizes the pipe. We also observe this phenomenon on the experimental setup with the three pipes having different inner diameters. One last interesting thing to notice on this graph is the restabilization: at certain values of β the pipes destabilize at a first critical speed and then restabilize before destabilizing again at a second critical speed.

Lastly, as shown in Figure 2.4(b), we use the linear model to generate numerical data on the deflection of the pipe. For this purpose, we proceed as follows: We compute the $2N$ eigenvalues and eigenvectors for the needed u, β, γ and α parameters with the method explained before. We rewrite the deflection as:

$$\eta(\xi, \tau) = \sum_{i=1}^{N} \phi_i(\xi) \boldsymbol{Q}_{N+i}(\tau), \tag{2.38}$$

where the $\boldsymbol{Q}$ vector is of size $2N$ and defined as:

$$\boldsymbol{Q}_i(\tau) = \sum_{j=1}^{2N} c_j \boldsymbol{A}_{i,j} e^{\lambda_j \tau}, \tag{2.39}$$

with $\boldsymbol{A}_j$, λ_j, c_j and i respectively the j^{th} eigenvectors and eigenvalues determined before, a constant depending on the initial conditions and the line of the $\boldsymbol{Q}$ and $\boldsymbol{A}$ vectors.

Then we chose the initial conditions: here we assumed that the pipe initial position corresponds to the first beam mode shape and that the pipe initial speed is equal to zero, which means:

$$\eta(\xi, 0) = \phi_1(\xi) = \sum_{i=1}^{N} \left\{ \phi_i(\xi) \sum_{j=1}^{2N} c_j \boldsymbol{A}_{N+i,j} \right\}. \tag{2.40}$$

With this initial condition equation, we determine the values of the c_j constants. Once the coefficients are known, we compute the deflection as follows:

$$\eta(\xi, \tau) = \sum_{i=1}^{N} \left\{ \phi_i(\xi) \sum_{j=1}^{2N} c_j \boldsymbol{A}_{N+i,j} e^{\lambda_j \tau} \right\}. \tag{2.41}$$

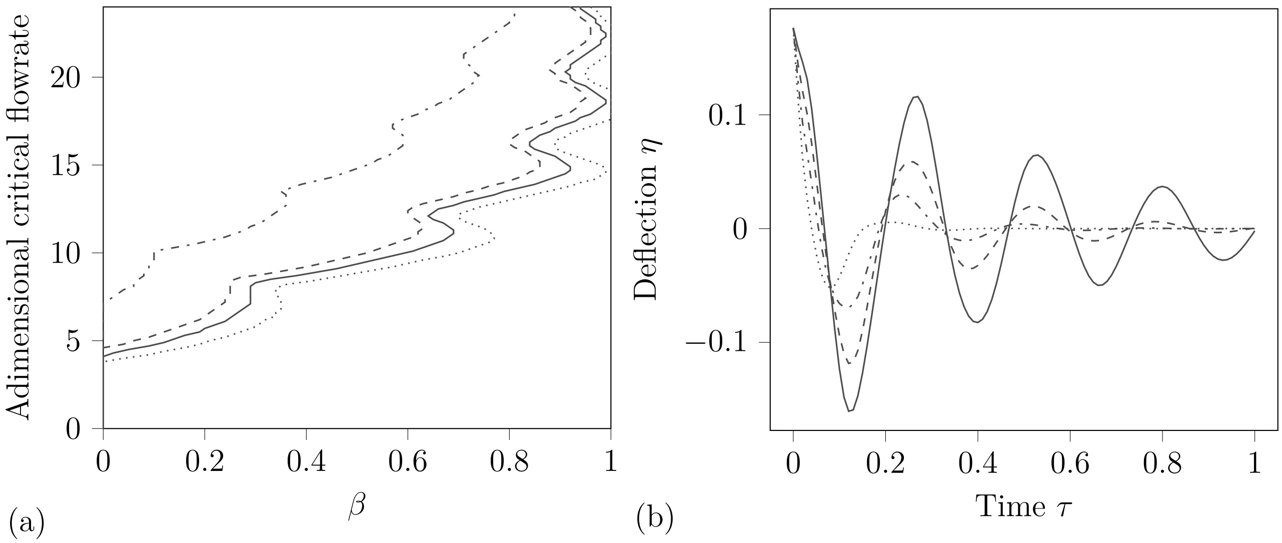


Figure 2.4 Graphics reproduced with the linear model detailed in reference [1]: (a) Flowrate of apparition of flutter instability for different gravity parameters: (——) $\gamma = 0$, (- - -) $\gamma = 10$, (· - · - ·) $\gamma = 100$ and (······) $\gamma = -10$ and (b) Data generated with the linear mode for the deflection of the tip of the pipe for different flowrates (——) $u = 3$, (- - -) $u = 6$, (· - · - ·) $u = 9$ and (······) $u = 12$.

Figure 2.4(b) represents the deflection of the tip of the pipe in time after being subjected to a perturbation corresponding to the precedents initial conditions. The flowrate influences the evolution of the damping which we will observe on the experimental results later.

CHAPTER 3 SYNTHESIS OF THE LITERATURE REVIEW AND RESEARCH OBJECTIVES

3.1 Synthesis of the literature review

Hydraulic power plants are under more stress than ever to adapt to the introduction of intermittent renewable energy. This new usage of hydraulic infrastructures creates the need for a closer monitoring of the turbines to follow their state, to predict maintenance and to avoid accidents and unexpected shutdowns. This motivates the global objectives of this project: develop methods and knowledge that could ultimately lead to the development of a hydraulic turbine digital twin. Because of the availability of lots of sensor data and the existence of many physical models describing the hydraulic turbines functioning, we use PINN methods to build a digital twin by combining sensors data and theoretical prior knowledge.

To develop the digital twin, it is not advisable to work on a real hydraulic turbine as it is a very complex system and it would be complicated to obtain the data needed to work on the development of the PINN methods. We therefore decided to work on a pipe conveying fluids for two main reasons. Firstly, it is a system that is fairly cheap and easy to install in a university lab to obtain data. Additionally, it is a very rich system that presents interesting physics phenomena on a wide range. It is also a system that can be modified easily to observe other phenomena with different boundary conditions or different pipes. The second reason we chose the pipe conveying fluid is that it is a system with fluid structure interactions analogous to some of those in hydraulic turbines. For example, water added mass and added damping play a fundamental role in the fatigue evolution for hydraulic turbines. This same water added damping also influences the behaviour of the pipe conveying fluid.

3.2 Research objectives

From what we have summarized in the literature review, we propose four objectives for this thesis that are also of interest for doctoral students working on the follow-up of the project.

- Design an experimental pipe conveying fluid setup that provides training data and validation data for the digital twin. The first setup is a cantilevered pipe but is flexible enough to install different other boundary conditions such as clamped tip or springs. It shall also be possible to use pipes of different shapes or with embedded ring masses. The pipe has to be observed both in linear conditions before the critical flowrate and also in non-linear instability conditions after the critical flowrate. The pump should deliver a flowrate which is high enough to reach the chaotic oscillations from the pipe conveying fluid.

- Set up the data acquisition and treatment system to provide the data to the digital twin. A new method with cameras is used to track the pipe position in three dimensions in all configurations and to measure the flow speed at the same time. This measurement method can also provide live data while the experiment is running to train the digital twin online in the future.

- Test the setup on the full range of the flowrate using pipes of different lengths and different diameters. Identify the mode shapes of the pipe and the influence of the flowrate on the motion, on the amplitude and on the vibration frequency of the pipe. We identify the different motion types such as planar, circular and chaotic, and other phenomena that could be studied in the future.

- Use PINNs with the pipe conveying fluid linear model before the critical speed. Use classic, weak-form and modal PINNs methods to simulate the pipe behaviour in the linear domain with numerically generated data to demonstrate that these methods are effective to characterize the pipe motion. Prove that the modal and weak formulations are more effective than the classic formulation and reduce the need for data and the risk of over-fitting. Then use classic PINNs with generated data and experimental data to simulate the pipe and to determine some hidden parameters of the system demonstrating that the PINNs not only simulate the motion of the pipe but also solve inverse problem.

CHAPTER 4 METHODOLOGY: EXPERIMENTAL SETUP OF A PIPE CONVEYING FLUID SYSTEM

4.1 Global architecture of the pipe conveying fluid setup

The architecture of the pipe conveying fluid setup is inspired by Paidoussis [50] and shown in Figure 4.1 with the principal components listed in Table 4.1. A two Horse Power (HP) centrifugal pump ② generates the flow in the circuit. The pump is powered by a tri-phase variator drive ④ to control the flowrate. The drive controls the current output frequency from 0 to 60 Hz with a 0.1 Hz frequency step. A flow control valve after the pump controls the flowrate more precisely if needed. The water then goes into a 15 USG steel tank ③ rated for a pressure up to 75 psi. In this tank, the free liquid surface between the water and the air on the top mitigates the potential fluctuations in the flow induced by the pump. The flexible plastic tubing in the circuit also adds damping to the system and makes the assembly of the setup easier. A magnetic Rosemount 8711 flowmeter with a 3/4" flowtube and a 8732 transmitter ⑤ measures the volumic flowrate in the circuit. The flow velocity is read directly on the flowmeter or is stored in a text file with a National Instrument NI9203 current acquisition card. The flowmeter delivers a 4-20 mA current signal that is scaled to the needed flowrate range. The section of the circuit then increases to a 2 inches diameter from a 3/4 inches diameter. A flow straightener ⑥ built with plastic straws taped together is fixed in the higher diameter section. This flow straightener with low diameters straws breaks large movements in the flow and also helps to get a laminar flow. After the flow straightener, the diameter of the circuit shrinks back to 3/4 inches to improve the flow quality. At the end of the circuit, the "pipe conveying fluid" ⑦ is fixed with a clamp and a hose fitting on which the pipe is embedded. The origin point of the pipe is considered to be the point where the hose fitting in the pipe ends. The pipe is then contained in a 4 × 4 × 2 ft acrylic and translucent tank ⑨. A white plastic liner covers up the tank and contain the splashes. The two sides of the tank opposite to the cameras are covered with white opaque plastic sheets to have a white unified background for the cameras and to facilitate image treatment. The water from the pipe flow goes to the main 35 USG plastic tank ① through a drain in the translucent tank before going back in the pump. Two relatively high speed FLIR BFS-U3-20S4C-C cameras with two 8 mm FL-CC0814-2M lenses ⑧ and ⑧’ are fixed with T-slotted rails. These fixations allow the operator to move the cameras in both directions. The lenses are as close as possible to the plastic plates of the tank to avoid the water drops on the tank. A water repellent spray applied on the tank walls increases the dripping of the water.

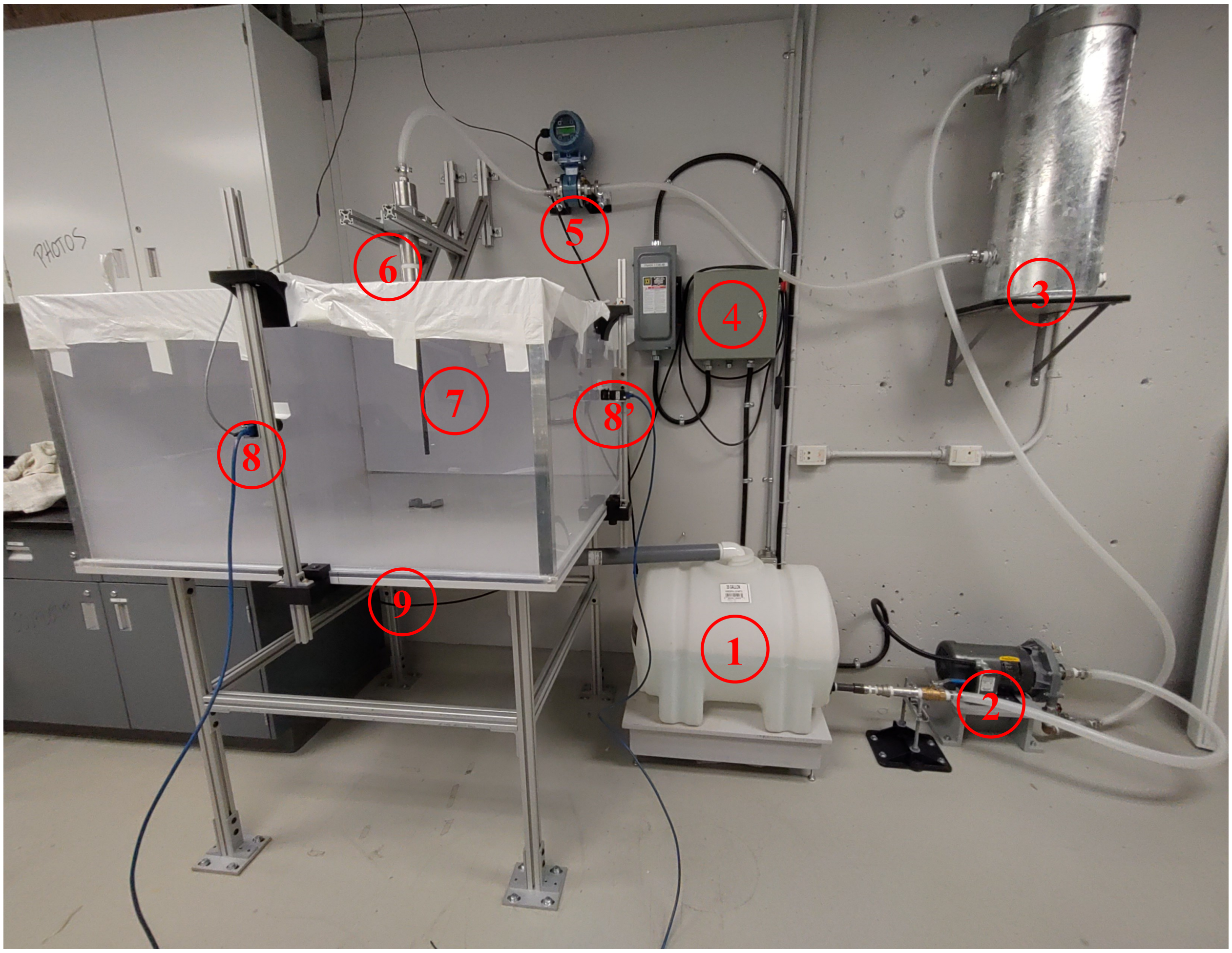


Figure 4.1 The pipe conveying fluid experimental setup

Table 4.1 Description of the components highlighted in Figures 4.1

No.	Name	Characteristics
1	Main storage tank	35USG plastic water tank with 3/4" drain
2	Pump	2HP centrifugal pump
3	Pressurized tank	15USG steel pressure rated water tank
4	Pump drive	Three phase pump drive from 0 to 60Hz
5	Flowmeter	Rosemount volumic flowmeter
6	Flow straightener	Large 2" pipe with straws
7	Pipe	Rubber-silicone pipe
8 and 8'	Two cameras	Flir USB3 cameras with 8 mm lenses
9	End container	$4 \times 4 \times 2$ ft transparent acrylic tank

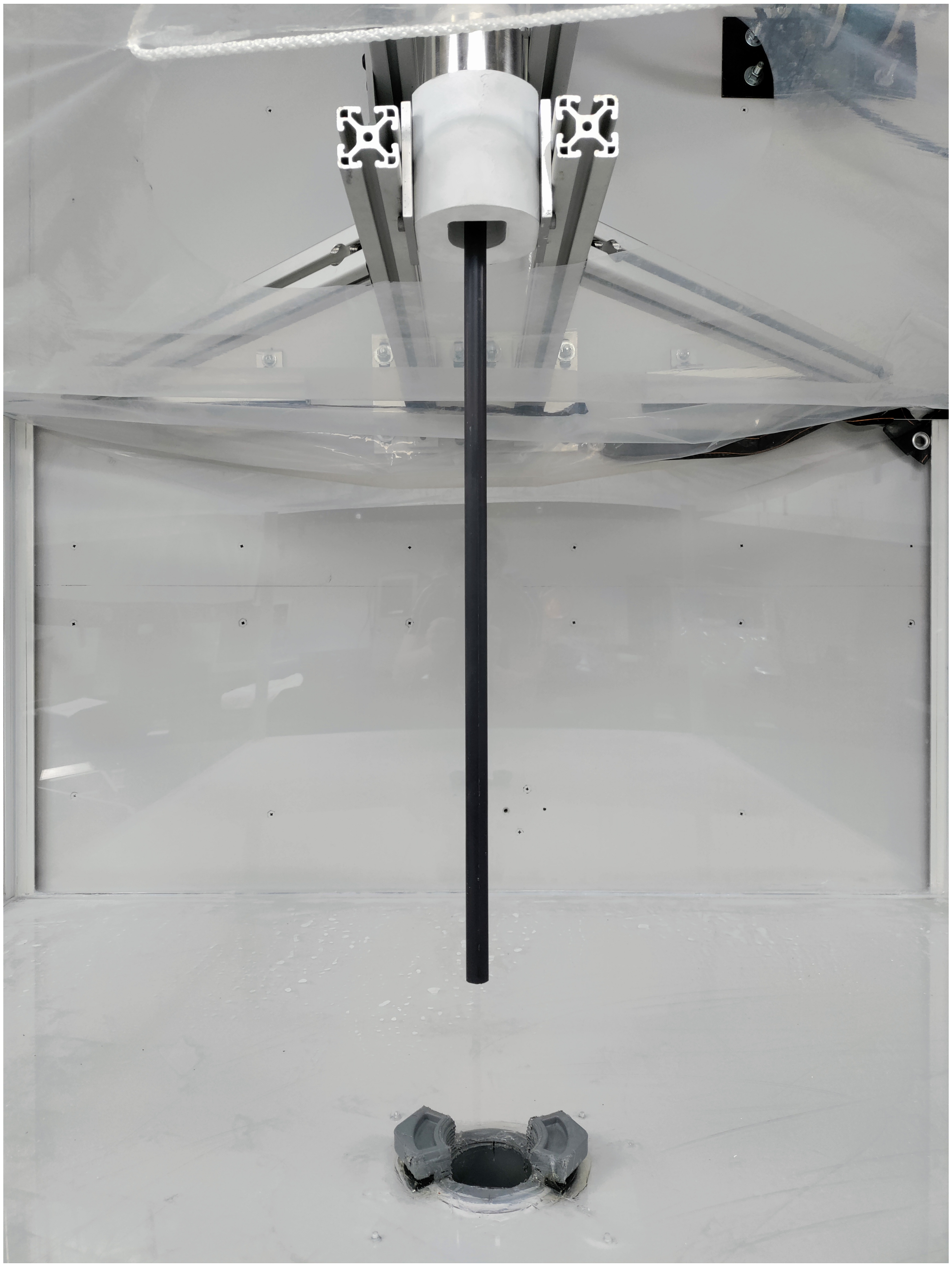


Figure 4.2 View of the pipe from the front camera position

4.2 Manufacturing and characterization of the pipe

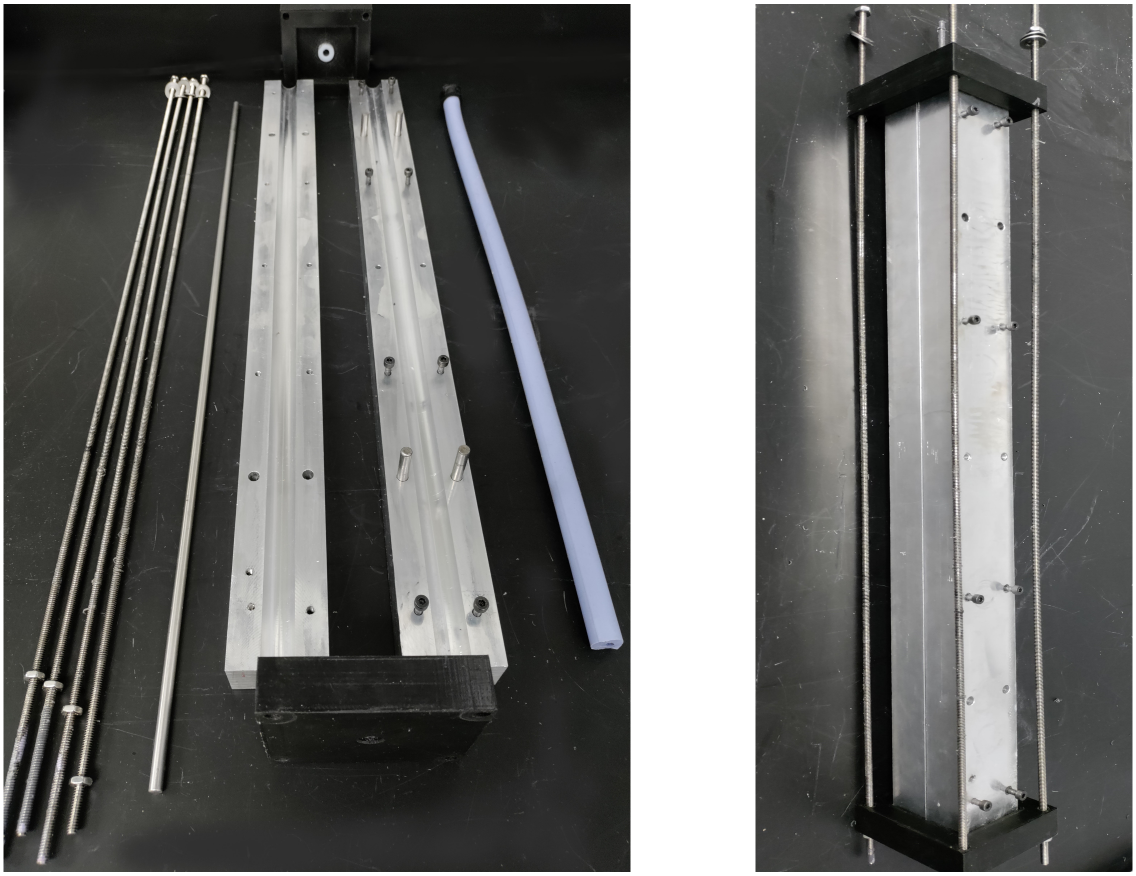
Commercial rubber-silicone pipes are widely available. However, we manufacture our own pipes since commercial pipe quality is not perfect. For example, they often have an initial curvature and their section is not perfectly circular because they are rolled during storage. These defects could be corrected by heating the pipes and maintaining them straight for a few hours, but we prefer to manufacture our own pipes shown in Figure 4.4 to have more flexibility on the choice of the pipe dimensions, parameters and also to have the possibility to embed metallic rings in the pipe. In the future, it might be interesting to use commercial pipes with their defects to assess the ability of the digital twin to handle the unknown defects.

4.2.1 The pipe manufacturing

To build our own pipe, we used a similar method as the one described in reference [17]. We built a mould described in Figure 4.3 composed of the following elements:

- Two aluminum parts that were machined with a three axes numerical milling machine and a 5/8" ball end mill. The internal diameter corresponds to the external diameter of the pipe and the length is the maximal length of the pipe. Four centring holes with centring pins and ten threaded holes were drilled in both parts and are used to tighten the two aluminum parts together and to avoid any leaks. It is possible to modify these parts of the mould by machining them to make different pipes or to mould rings in the pipe.
- A central aluminum rod to make the central hole in the pipe. Its diameter corresponds to the internal diameter of the pipe. These rods were bought from a local metal supplier.
- Two 3D printed parts made from Polyactic Plastic (PLA) maintaining the rod centred as well as the two aluminum parts. One of these parts is shaped so that we can inject the rubber silicone with a syringe. These two PLA parts are maintained together using four threaded rods longer than the mould.

Smooth On OOMOO30, a room temperature vulcanizing (RTV) rubber-silicon was selected to manufacture the pipes. The choice of the rubber-silicon is important to obtain the desired properties for the pipe. It is also important to make sure that the working time is long enough for the moulding process. One of the tricky point is to obtain a pipe rigid enough with a silicon that is fluid enough. Rubber silicon with low viscosity is easier to mould into a pipe. In order to obtain more rigid pipe, rubber silicon with larger viscosity should be used.



(a) (b)

Figure 4.3 (a): Different components of the mould with, from left to right: the threaded rods, the centring rod, the aluminum and PLA parts and the pipe. (b): Assembled mould.

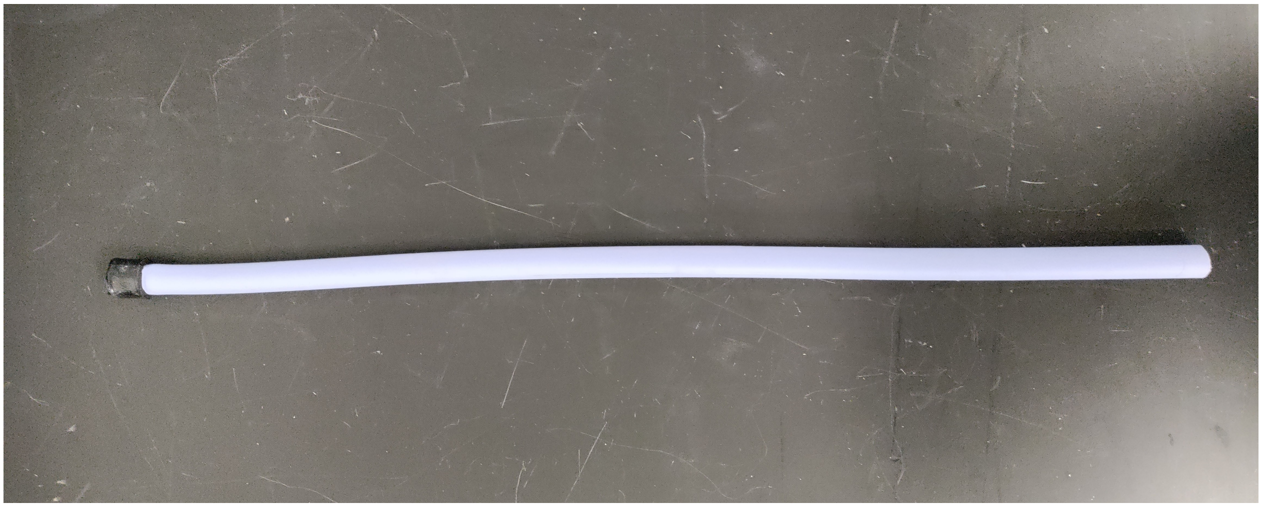


Figure 4.4 Pipe without the black pigments

However, it will be hard to inject high viscosity rubber silicon in the mould and it could produce bubbles or other defects in the pipe. The chosen rubber-silicon has lavanda colour so we mix black pigments with the rubber-silicone to blacken the pipe to facilitate image treatment.

The detailed procedure to cast a pipe is as follows:

- Perfectly clean all the parts of the mould and spray them with easy unmoulding chemical to facilitate the removal of the pipe from the mould after cure.
- Close the two external parts of the mould together and adjust their positions by using the centring pins. Then put the two PLA parts on each side of the mould and tighten at the same time the two aluminum parts and the two PLA parts with the screws and the threaded rods.
- Position the mould vertically on a pierced wood plank.
- Mix the two components of the rubber-silicon in a large cup according to the supplier instructions. The black pigments are added at this step at a maximum of 3% of the mass of the mix. After a thorough mixing of the two components, the cup is placed in a vacuum chamber for a few minutes to remove air bubbles. Remove the cup from the chamber when no more bubbles are visible.
- Suck the rubber-silicon with a syringe and pour it into the mould from the bottom hole until the mix is spilling out from the top hole. Then, insert the centring rod and block the bottom hole of the mould once the rod is in place.
- After the twelves hours cure time, unmould the pipe by removing the two PLA parts and then the two aluminum parts. The easiest way to remove the centring rods is with a high pressure air gun. Put the tip of the gun between the centring rod and the rubber silicone, inject the air, and then pull the pipe from the rod while continuing to inject the air.
- Cut the pipe to the desired length.

4.2.2 Characterization of the pipe by free vibration

To use the linear PDE of the pipe to properly train the digital twin, it is important to determine the non-dimensional parameters of the pipe in Equation (2.19) or (2.20). We cannot use the data given by the rubber-silicone manufacturer as they are not accurate and are subjected to changes in different lot numbers. Therefore, we tried two different methods to characterize the pipe: free vibration and vibration by shaker.

The first method is inspired from the method of Paidoussis [61]. We record the pipe in cantilevered configuration with a high speed camera in planar free motion after a planar perturbation of 0.2 m without flow going through the pipe. To observe this movement, we hold the pipe in a perpendicular plane to the camera plane and then release it. An image processing program, presented later, retrieves the amplitude decay and the pipe free motion frequency from the videos. We then use these data to determine the parameters as follows.

Figure 4.5 shows the evolution of the position of the tip of pipe No.1 as listed in Table 4.2 along time after perturbation. The grey solid curve shows the position of the tip at each frame and the two black dashed lines follow the decay of the amplitude of the pipe with time. To obtain the evolution of the amplitude, we used the function find_peaks from Scipy in Python to find the peaks of the position of the tip. Figure 4.5 confirms that the position follows a sinusoidal law multiplied by an exponential amplitude term.

We then fitted a linear function to the logarithm of the amplitude from the black dashed lines and performed a FFT transform on the position of the tip of the pipe in Figure 4.6. Figure 4.6(a) compares the linear regression results in dashed line and the actual measured amplitude in solid line. The overall agreement is good but the logarithmic amplitude is not fully linear which could be explained by the large amplitudes at the beginning of the video. It would be possible to have a better agreement on a smaller range of amplitude at the expense of having to use two coefficients for the full amplitude range. The amplitude logarithmic coefficient is the first term of the linear regression. We performed the operation on three videos and computed the coefficient as an average of the three videos: $\delta = -0.0167$ where δ is the logarithmic amplitude decrement corresponding to the first regression coefficient. Figure 4.6(b) shows the result from the FFT of the position of the pipe. This FFT gives us the natural frequency of the pipe without water. As for the logarithmic decay, we filmed three videos and computed the average frequency $f = 0.915$ Hz.

Then we compute the evolution of the non-dimensional fundamental frequency with respect to the γ parameter $Re(\omega_1)(\gamma)$, using the same method used to plot the Argand diagram in Figure 2.3 in the literature review for 501 values of γ in the $[0, 500]$ interval and for $u = \beta = 0$. We notice that $Re(\omega_1)$ does not depend on α, so we can generate these points with any values of α. We use $\alpha = 0.02$. We fit $Re(\omega_1)(\gamma)$ with a polynomial function of order 20 to obtain a formulation for γ values in the range of the values used to generate the data. Finally, the relation between the dimensional and the non-dimensional parameters links the measured fundamental frequency of the pipe and the non-dimensional fundamental

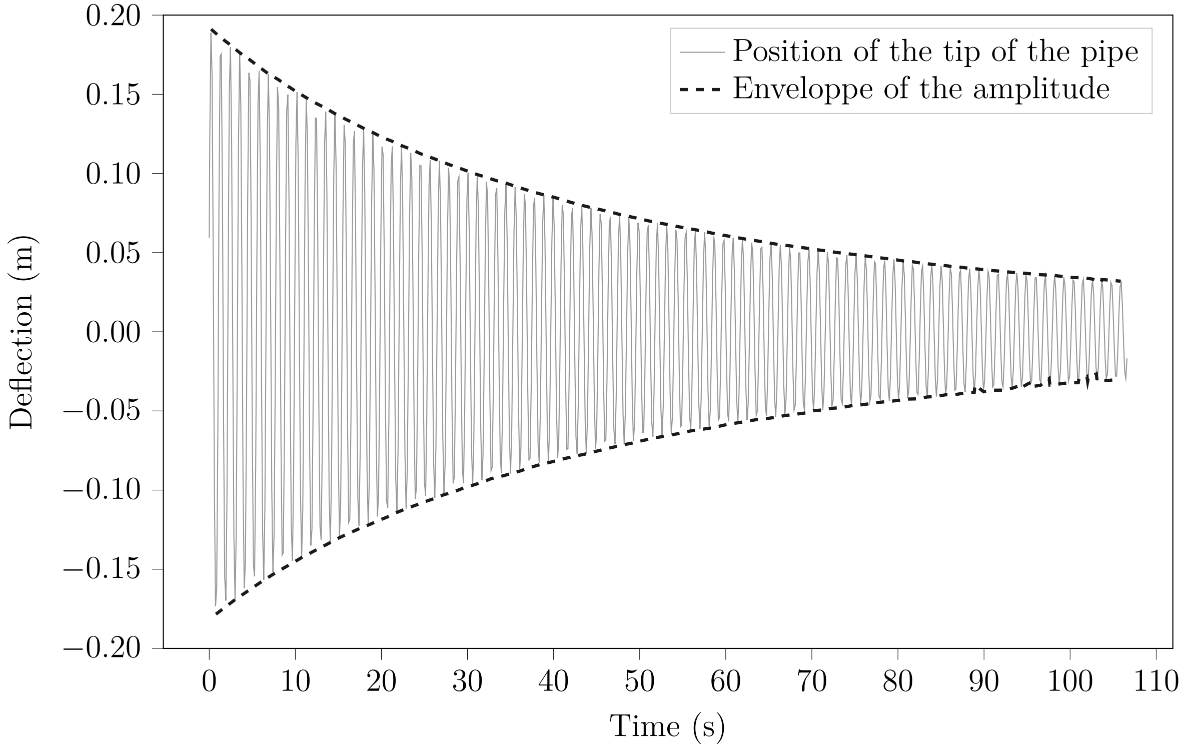


Figure 4.5 Position evolution of the tip of the pipe No.1 from Table 4.2 after a perturbation of 0.2 m and without flow.

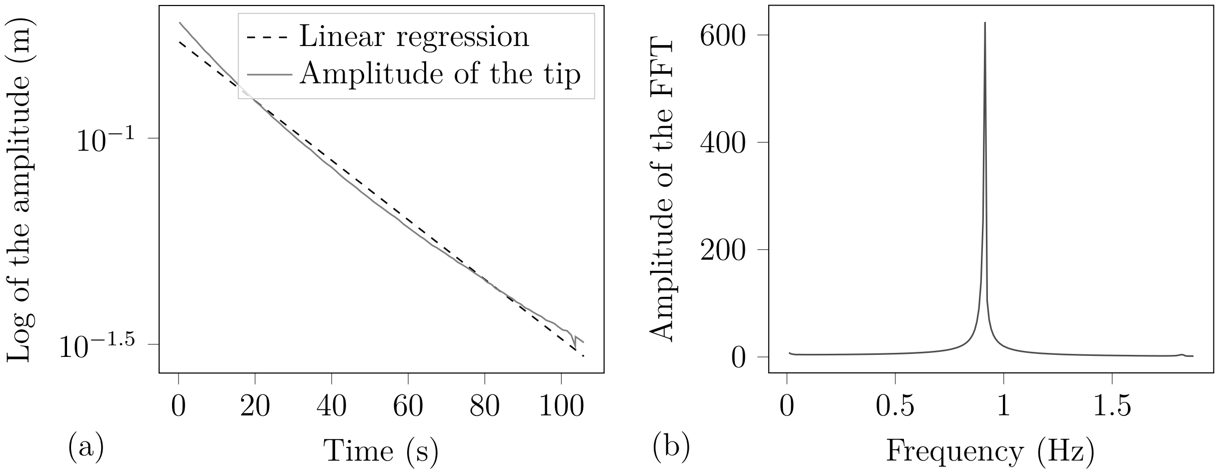


Figure 4.6 (a): Comparison of the logarithmic amplitude of the tip and linear regression associated and (b) result of the FFT transform of the position of the tip.

frequency as follows:

$$\tau = \left(\frac{EI}{M+m}\right)^{\frac{1}{2}} \frac{t}{L^2}, \tag{4.1}$$

and

$$\gamma = \frac{(M+m)L^3}{EI} g. \tag{4.2}$$

We rewrite Equation (4.1) as:

$$\frac{1}{Re(\omega_1)} = \left(\frac{EI}{M+m}\right)^{\frac{1}{2}} \frac{1}{2\pi f_1 L^2}, \tag{4.3}$$

where f_1 is the measured fundamental frequency of the pipe.

We then combine Equation (4.2) with (4.3) to get the following relation:

$$\frac{\gamma}{Re(\omega_1)^2} = \frac{g}{4\pi^2 f_1^2 L}. \tag{4.4}$$

With the polynomial fit giving $Re(\omega_1)(\gamma) = P(\gamma)$, we can write the following equation:

$$\frac{\gamma}{P(\gamma)^2} - \frac{g}{4\pi^2 f_1^2 L} = 0. \tag{4.5}$$

A Newton solver finds the root of Equation (4.5) to obtain the correct value of γ. We then get the value of E, the pipe Young modulus. Figure 4.7 shows how to determine the γ parameter of the pipe from the measurement of the pipe natural frequency f_1. Figure 4.7(a) corresponds to the polynomial fitting the evolution of the real part of the eigenfrequency with respect to gamma. Figure 4.7(b) shows the solution of the Newton solver to find the value of γ from Equation (4.5).

We follow a similar method to find the damping parameter α. We use 501 values of α in the $[0, 0.05]$ interval to compute the evolution of the imaginary part of the non-dimensional fundamental frequency with respect to the α parameter $Im(\omega_1)(\gamma)$ and then we fit an order 20 polynomial to get a relation.

We then use the following relation:

$$\delta_1 = \frac{2\pi Im(\omega_1)}{Re(\omega_1)}, \tag{4.6}$$

where δ_1 is the logarithmic amplitude decrement of the pipe and with the value of $Re(\omega_1)$ from the previous step.

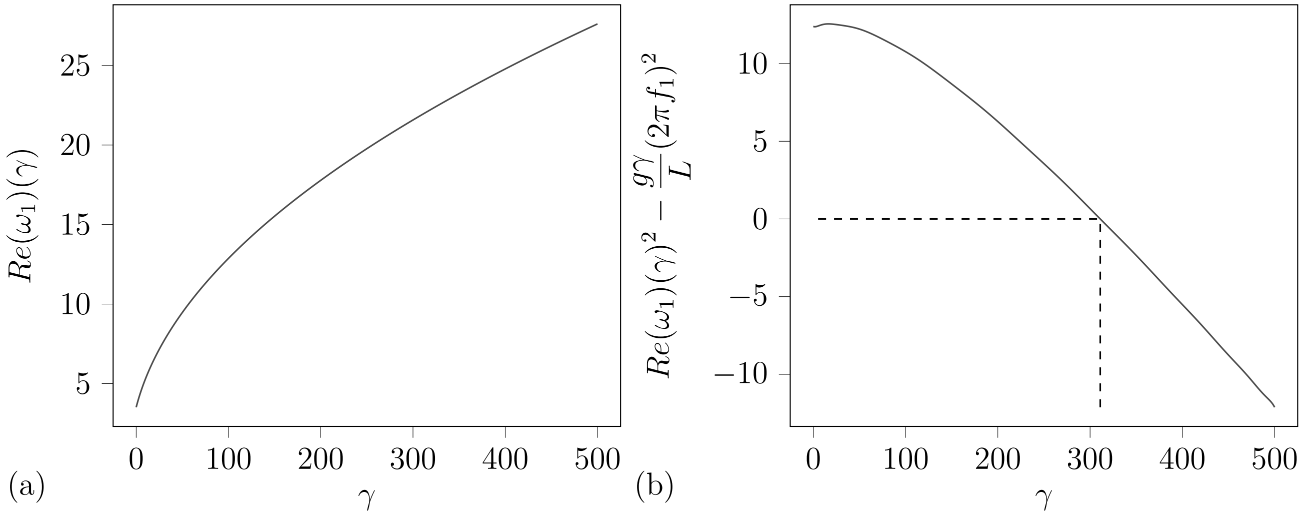


Figure 4.7 Determination of the γ parameter: (a) Real part of the first eigenvalue as a function of γ and (b) determination of the value of gamma by finding the root of the function constructed with the measured frequency.

We have to find the root of the following equation to find the value of α:

$$\delta - \frac{2\pi P(\alpha)}{Re(\omega_1)} = 0, \tag{4.7}$$

where δ is the measured amplitude decay and $Re(\omega_1)$ is known from the step before. This way we determine the value of α and then the value of the damping parameter E^*.

Figure 4.8 represents the imaginary part of the pipe first eigenvalue with respect to alpha on the left. On the right the function to determine alpha with the Newton algorithm is shown.

For this project, seven different pipes were casted. They are all listed in Table 4.2. They were made using the same rubber-silicone lot and thus have the same young modulus $E = 225$ kPa and the same damping parameter $E^* = 5334$. The pipe No.1 was used to determine the parameters E and E^* with the previous method. The seven pipes all have the same external diameter $D = 15.875$ mm.

We try another characterization method with a shaker to find the pipe parameters for higher modes. Unfortunately, this method did not lead to exploitable results because of two main problems: The shaker did not offer enough flexibility to subject the pipe to the appropriate forcing and the linear model is not appropriate to study forced vibrations of high amplitudes. The study of the pipe subjected to the shaker vibrations and the constructed characterization method is presented in Appendix A. This method could be of interest in the future with a better shaker and by applying the non linear model.

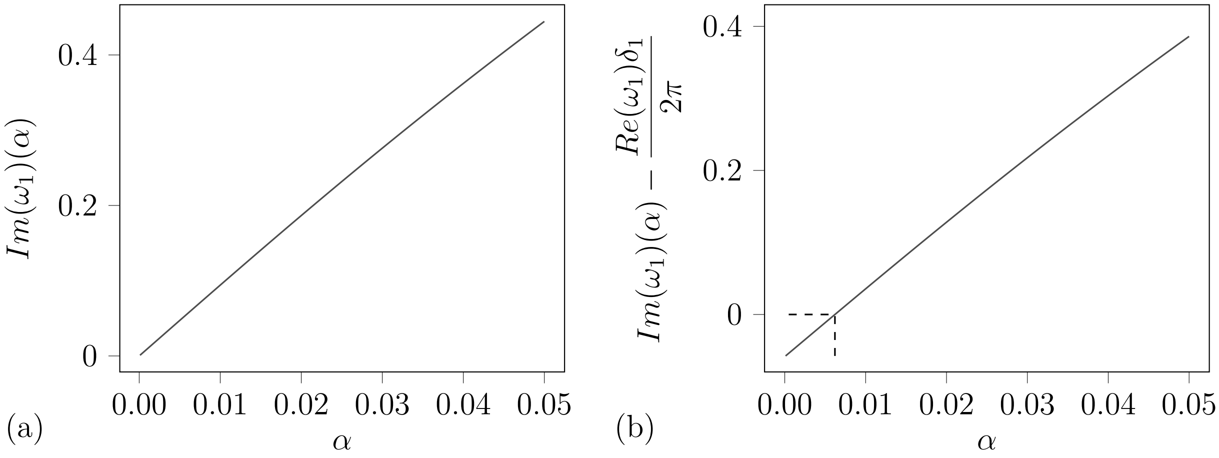


Figure 4.8 Determination of the damping parameter α: (a) Imaginary part of the first eigenvalue as a function of α and (b) determination of the value of α by finding the root of the function constructed with the measured frequency.

Table 4.2 Different pipes used in this project

No.	$L(m)$	$d(mm)$	α	γ	β	$m(kg/m)$	$M(kg/m)$	$I(10^{-9}m^4)$
1	0.46	6.35	0.005806	355.04	0.1240	0.2228	0.03154	3.038
2	0.41	6.35	0.007308	251.39	0.1240	0.2228	0.03154	3.038
3	0.36	6.35	0.009479	170.18	0.1240	0.2228	0.03154	3.038
4	0.46	7.9375	0.005765	360.12	0.1986	0.1989	0.04929	2.923
5	0.39	7.9375	0.008020	219.47	0.1986	0.1989	0.04929	2.923
6	0.46	9.525	0.005640	376.18	0.2948	0.1697	0.07097	2.714
7	0.40	9.525	0.007459	247.34	0.2948	0.1697	0.07097	2.714

4.3 Data acquisition

The pipe is light and very flexible, therefore it is almost impossible to put traditional sensors like strain gauges, pressure sensors or accelerometers on it. We therefore use contactless measurements. In most work from the literature, laser distance measuring sensors or points tracking system track the pipe [51,52,62]. In this project, we want to access data on the whole length of the pipe without limiting ourselves to planar movements. In some configurations, the pipe movement is three-dimensional and it does not make sense to constrain the pipe to a two dimensional movement. We therefore use two high speed cameras to film the pipe from two sides.

4.3.1 The cameras

To synchronize the two cameras, we use a 0-10 V square signal generated with a National Instrument NI9401 card to trigger the acquisition of the two cameras and of the flowmeter to be sure that each camera records a frame simultaneously and that we record the corresponding flowrate. It is important to verify with the cameras software that no images are lost during acquisition to make sure that both cameras stay synchronized.

Image treatment

To extract the position of the pipe from the images, we use the contrast between the pipe and the background since the pipe is a dark grey while the background and the liner are plain white. We follow the subsequent procedure to treat the image:

- We convert the image in grey scale using the OpenCV library in Python.
- We convert the image to a binary image which is basically an array where each pixel is either 1 or 0 depending if it is considered black or white. We use the threshold function from OpenCV, depending on the situation and the lighting we use a hard threshold value for the whole image or an adaptive threshold. The values of the thresholds are determined by trial and errors.
- We remove islands of pixel in the image i.e. each detached group of pixel containing a low number of pixels (white or black) is removed to have a better image. This function removes black points corresponding to water splashes and white points on the pipe corresponding to the light reflection.

Once the images are treated as explained above, we proceed to extract the position of the pipe.

Camera calibration

The two cameras are fixed on the two sides of the translucent tank and the sensing planes are perfectly parallel to the central plane of the tank. It is therefore not necessary to correct the tangential error. Also, after looking thoroughly at the images, we discovered that the radial distortion is very subtle and we decided to ignore it in the current study. We only have to calibrate the camera to get the origin point of the pipe on the vertical axis, the origin point on the horizontal axis and the scale linking the length in pixels on the image to the length in millimetre on the physical system.

Firstly, to determine the scale of the camera, we printed a black square on a rigid plate and placed it on the planes parallel to each camera at the rest position of the pipe. Knowing the actual size of the square, we find the size in mm of a pixel with the following formula: scale (mm/pixel) $= \frac{\text{square width (mm)}}{\text{square width (pixels)}}$.

Then, to find the origin of the coordinate system, we take a picture of the pipe at rest with no water. We compute the position of the middle line of the pipe which is the origin on the horizontal axis. Then, we find the coordinates of the tip of the pipe on the photo and knowing the length of the pipe, we find the coordinates of the top of the pipe which is the vertical origin point.

For every image in the video, we then have to subtract the coordinates of the origin on the vertical and horizontal axes and multiply by the scale to get the coordinates in mm in the pipe coordinates system.

Extraction of the data from the video

Once the images are treated and calibrated, we determine the (x, y, z, t) coordinates of the points on the pipe to train or validate the digital twin. Each image corresponds to a certain time t that is computed with the cameras frame rate. From each image, we extract the coordinates (x, z) from the front and (y, z) from the side of each pixel that is considered black i.e. on the pipe. We fit a polynomial function on the (x, z) and (y, z) to find $x = P(z)$ and $y = P(z)$. This polynomial functions goes through the centreline of the pipe with one condition: the tip of the pipe must be its lowest point. If it is not, there would be two values of y and x for one value of z and the polynomial would not work. We decided not to consider this problem as this configuration rarely appears and only at the highest pump speeds. We then store the coefficients values of the polynomial as well as the position of the tip of the pipe on each camera in a text file.

We then correct one last error on these data: the parallax. The pipe, moving in the tank is getting closer or further than the central plane in which the camera scale was computed. The deflection is therefore overestimated when the pipe is closer to the camera or underestimated when the pipe is further from the camera. We treat this problem by using trigonometry as follows.

We start by creating two lists of N points evenly distributed between $z = 0$ and $z = z_{tip}$ for both cameras. We then compute the lists of the associated deflection x and y with the polynomial functions determined previously. The points with the same index in the lists do not have the same z coordinates because z_{tip} might be higher for one of the camera if the

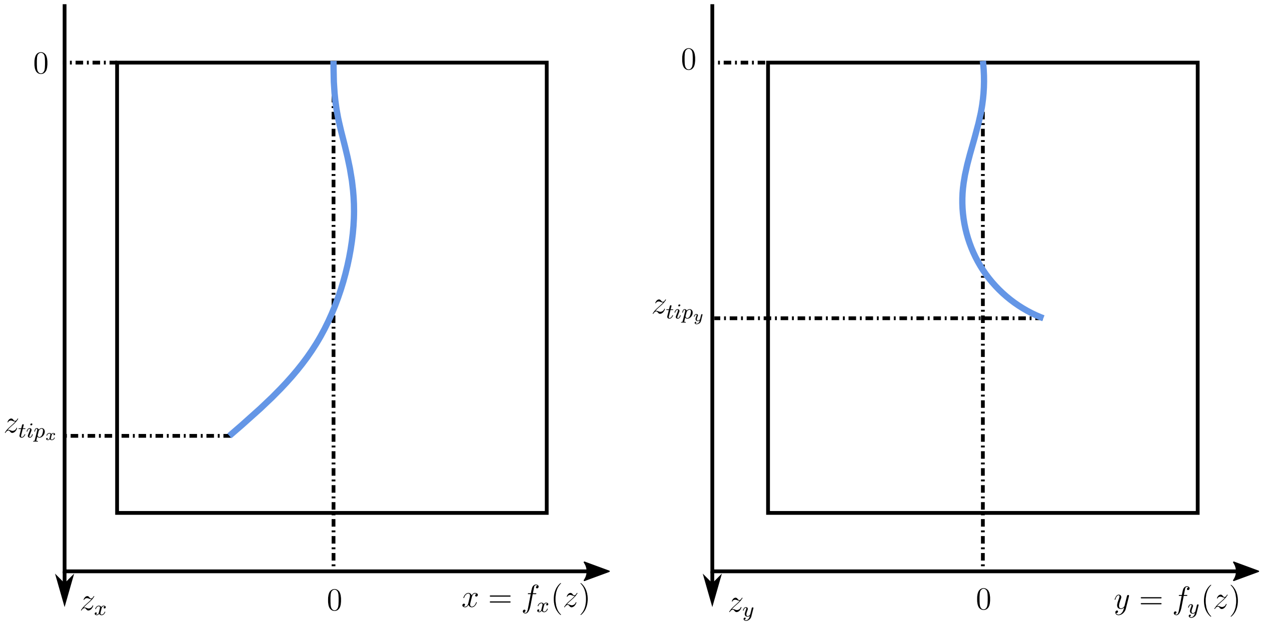


Figure 4.9 First step of parallax treatment: Creation of lists of points for both frontal and side images.

pipe is closer as illustrated on Figure 4.9. However, since the points are evenly distributed and since both lists have the same length, we assume that points with the same index in both lists correspond to the same physical point on the pipe.

Then, we correct the y and x coordinate of each points in the lists. We use the Thales theorem as shown in Figure 4.10 to compute the actual coordinates x and y from the coordinates observed by the camera in the central plane. We get the following relationships:

$$\frac{y_a}{y_c} = \frac{D_x}{D_x - x_c}, \tag{4.8}$$

and:

$$\frac{x_a}{x_c} = \frac{D_y}{D_y + y_c}. \tag{4.9}$$

By combining the previous equations we obtain the following formula for the corrected deflections y_c and x_c:

$$x_c = \frac{x_a D_x (D_y + y_a)}{D_y D_x + y_a x_a}, \tag{4.10}$$

and:

$$y_c = \frac{y_a D_y (D_x - x_a)}{D_y D_x + y_a x_a}. \tag{4.11}$$

With the same Thales theorem, we correct the value of z for each points in the list as shown

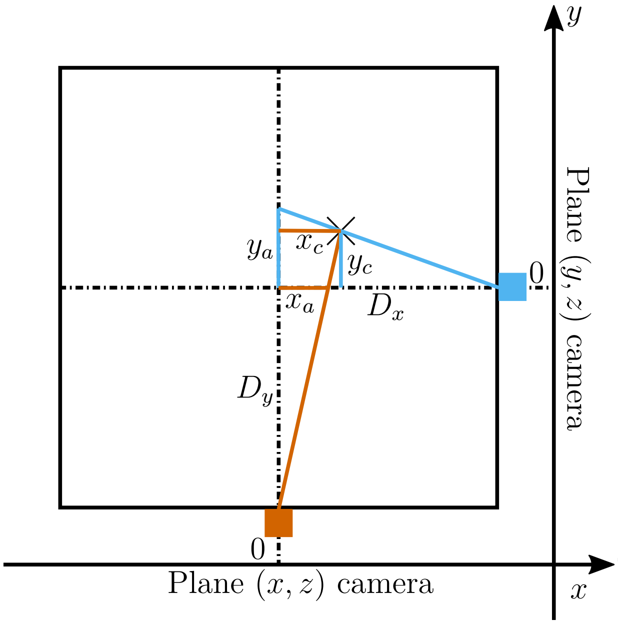


Figure 4.10 Second step of parallax treatment: correction of the x and y deflections.

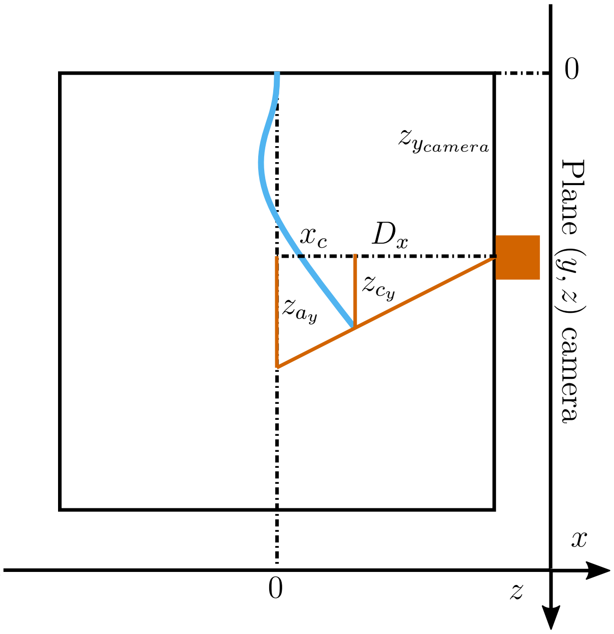


Figure 4.11 Third step of parallax treatment: correction of the z height for each points.

in Figure 4.11. We obtain the following formula for the correction of z for both cameras:

$$z_{x_c} = z_{x_a} \frac{D_y + y_c}{D_y}, \tag{4.12}$$

and,

$$z_{y_c} = z_{y_a} \frac{D_x - x_c}{D_x}. \tag{4.13}$$

The corrected value of z on both coordinates should now coincide. The small difference observed is caused by defaults in the image and we merge the two lists of z coordinates in one by taking the average of both.

We then repeat the polynomial fitting step on the z, x and y list after correction to obtain the coefficients of the polynomial valid with corrections.

During image acquisition, some frames are sometimes dropped. This is caused by speed limit of the computer Solid State Drive (SSD), thus letting the computer Random Access Memory (RAM) fill faster than it can be transferred in the SSD memory. This is a big problem when only one camera drops frames, since the frames from both cameras are not synchronized anymore. The software from the camera manufacturer does not allow us to detect reliably when a frame is dropped, so we had to develop our own program to detect it. To do so, we used the fact that the pipe tip height after correction should be the same on both cameras. If the error is too high it means one of two things. Either one of the image is of bad quality, probably because of a splash of water on the glass in front of the camera and the entire pipe cannot be detected or a few frames have been dropped on one camera and the images are not synchronized anymore. We easily identify which case it is by plotting the height error against time. If the error is only spiking for a few frames, then it corresponds to the first case caused by a splash of water. However, when the error is significantly increasing for all the frames, then frames have been dropped on one of the cameras. By looking at these graphics we can easily decide which videos to reshoot.

4.3.2 Flowrate

A magnetic Rosemount 8711 flowmeter with a 3/4" tube and a 8732 transmitter measures the volumic flowrate in the circuit. The 4-20 mA signal is scaled to correspond to a flowrate range from 0 to 50 liters per minute, 50 lpm corresponding to the maximum flowrate that is reached with pipe No.7. The flow speed is acquired five times per seconds with a National Instrument current acquisition card. The flowrate is computed as the mean of the flow velocity on the whole acquisition period to reduce noise influence. During long operations of the setup, the temperature of the water increases by up to 10°C. From 20 to 30°C, the density of water varies from 0.9982 kg/l to 0.9956 kg/l, which represents a 0.26% difference considered negligible in our experiments.

4.4 Acquisition method

The flowrate is firstly set to the desired value by changing the pump drive frequency and if needed, by adjusting the flow regulation valve. It is important to wait for up to one minute after the adjustment to let the flow velocity stabilize because of the flow and pump inertia. All the videos are filmed at 150 frames per second.

4.4.1 Below critical speed

In the case of measurements below critical speed, only one camera is needed. Indeed, the pipe is stable and the movement of the pipe is filmed after perturbation. We perturbed the pipe in a plane parallel to one of the camera to use only this camera. The movement usually stays in the same plane, thus justifying the use of only one camera.

The perturbation of the pipe has an effect on the flowrate in the pipe, especially at low flow. We therefore measure the flow velocity before the perturbation and consider this is the value of the flowrate in the pipe during the acquisition.

For our measurements, we started at the lowest drive frequency at which the water was running in the circuit. We follow the following steps until the critical speed is reached:

- Set the flowrate and wait for it to stabilize.
- Measure the flowrate during a period of 30 seconds and take the mean value.
- Start the video acquisition.
- Manually perturb the pipe.
- Once the movement has stopped, stop the video and increase the drive frequency by 0.5 Hz and get back to the first step.

Using the cameras as explained previously, we extract the position of any point on the pipe using a polynomial. We do not have to apply the correction steps for only one camera because the pipe stays at the same distance of the camera.

4.4.2 Above critical speed

When measuring the pipe movements at flowrates higher than the critical speed, we need the two cameras because the movement is three dimensional and even when the movement is planar there is no way to control the movement plane to make it parallel to one of the camera. We do not have to touch the pipe to induce any perturbation so we just have to set the flowrate and wait for it to stabilize and then start the video and the flowrate acquisition with the synchronizing signal.

We took 40 seconds videos every 0.5 Hz of the drive frequency between the critical speed and the maximum frequency of the drive.

CHAPTER 5 EXPERIMENTAL RESULTS ON THE PIPE BEHAVIOUR AND DISCUSSION

In this chapter, we present the results from the first experimental campaign conducted with the pipe conveying fluid setup on seven pipes of three different diameters and from low to maximum flowrate.

5.1 Below critical speed

In this section, we study the behaviour of the pipe before the critical speed at which flutter appears.

5.1.1 Response to initial perturbations

For low flowrates below the critical speed, we have to subject the pipe to a perturbation by holding it before releasing it. Figure 5.1 shows the transient period before the pipe returns to its rest position. We measure the position of the tip of the pipe during this transient period for different values of the flowrate below the critical speed. This figure shows the amplitude for the pipe No.1 from Table 4.2 of inner diameter $d = 6.35$ mm and of length $L = 0.46$ m. The tip of the pipe follows a sinusoidal law for the four non-dimensional flowrates: $u = 3.25$, $u = 8.34$, $u = 12.12$ and $u = 15.49$. The damping is higher for the Figure 5.1(b) and Figure 5.1(c) for intermediate flowrates. It can be seen that, the second mode appears in Figure 5.1(c) and Figure 5.1(d) for the highest flowrates. We observe the effect of added damping as the damping increases in figures (b) and (c) before decreasing when the flowrate gets closer to the critical speed in (d).

5.1.2 Evolution of the damping

To get a better understanding of the evolution of the damping with the flowrate, we computed the logarithmic decrement of the amplitude in Figures 5.2 for all the tested flowrates and for six different pipes with three different diameters and three different lengths. The dotted lines represent the experimental logarithmic decrement as a function of the reduced flow velocity u. The continuous lines represent the theoretical decrement obtained from the linear model presented in the literature review. To extract the decrement value we detect the peaks from Figure 5.1, we compute their logarithmic values and then fit them with a linear function. The first coefficient of this value is the decrement coefficient. For the theoretical part we proceed

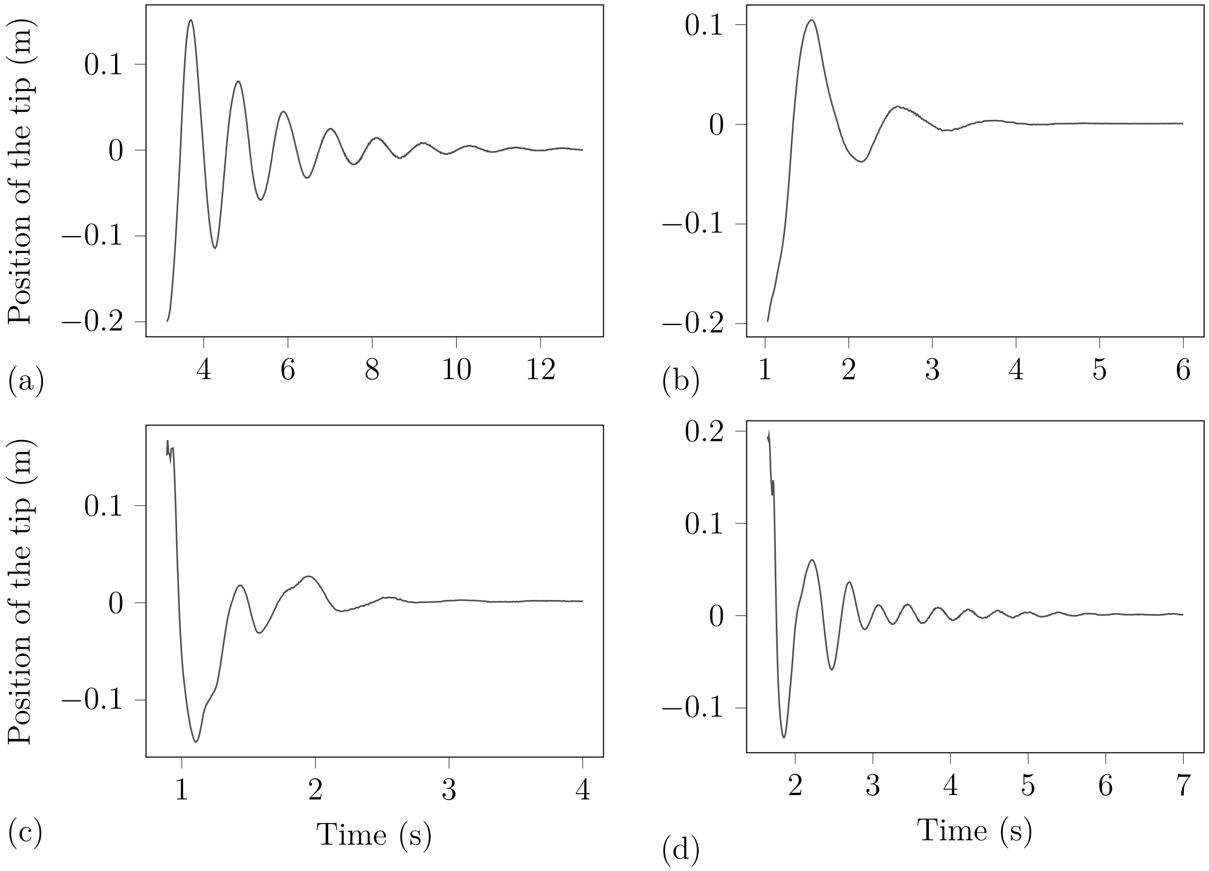


Figure 5.1 Evolution of the position of tip of the pipe in meters after a perturbation for the pipe No.1 and for four different non-dimensional flowrates: (a) $u = 3.25$, (b) $u = 8.34$, (c) $u = 12.12$ and (d) $u = 15.49$.

with the same method after generating the amplitude plots with the method presented in the literature review.

In all of the six cases, and for the theoretical and experimental cases, the decrement firstly increases with the flowrate to a maximum and then decreases sharply until the critical speed is reached. The maximum decrement is higher and is reached at higher flowrate for larger inner diameters pipes. This maximum decrement is higher and reached at higher flow velocity for longer pipes. For the pipe in Figure 5.2(c), the theoretical result shows a significant difference since the maximum decrement is not sharp but more of a plateau. This can be explained by the method used to compute the theoretical damping. Indeed, fitting a linear regression on the peaks of the logarithmic amplitude can prove to be an unreliable way to determine the logarithmic decrement when two modes are present as in Figure 5.1(c). This problem is also responsible for the irregularities at the maximum damping for the other plots. Despite this problem, the experimental results show the same tendency as the theoretical results even if the maximum decrement is sharper in the theoretical model and the experimental results are slightly shifted towards larger flowrates.

The water flow in the pipe is responsible for the added damping which explains why the decrement increases with the flowrate at first. Indeed, the Coriolis force term increases faster than the centrifugal term at low flowrates and therefore stabilizes the pipe. When the flow speed increases further, the centrifugal term increases faster than the Coriolis term and destabilize the pipe as the flowrate gets closer to the critical speed at which the motion becomes fully unstable. This explains why the decrement decreases after reaching a maximum at an intermediate flowrate.

For a higher inner diameter, the mass parameter β and the gravity parameter γ are higher as we can see in Table 4.2. The water added damping is hence higher since the Coriolis force is proportional to the β mass parameter and creates a stronger damping. It explains why the decrement maximum is higher when the inner diameter is larger but also why it is reached at higher flow velocity as the critical speed is also higher since the pipe is more stable with larger β and γ parameters as seen in Figure 2.4(a).

When the pipe is longer, the maximum damping is more important and is reached at higher flow velocity as we can see by comparing Figures 5.2(a), (c) and (e). Table 4.2 shows that longer pipes have a higher gravity parameter γ and damping parameter α and are thus more stable since the gravity restoring force is stronger. This explains why the damping is stronger and also why the curves in Figure 5.2(a) are shifted toward higher flowrate for a longer pipe compared to Figure 5.2(c) for a shorter pipe.

The experimental results in Figure 5.2 are shifted to higher flowrate compared to numerical data which concurs with the higher experimental critical flowrate compared to the theory as the next section demonstrates. This is explained by additional damping not taken into account in the linear model that would delay the destabilization of the pipe. Indeed, we did not consider air friction, damping from the structure and pipe fixation and non-linear damping terms that can increase the stability of the pipe [63]. Moreover, it is to be noted that the flowrate is not constant during the experiment. When the pipe is perturbed the flowrate drops before increasing back to the pre-perturbation value when the pipe has regained its rest position. The flow speed acquired is the flowrate measured before the perturbation and is actually higher than the flowrate going through the pipe during the vibration which explains why the experimental curves are shifted to the higher flowrates.

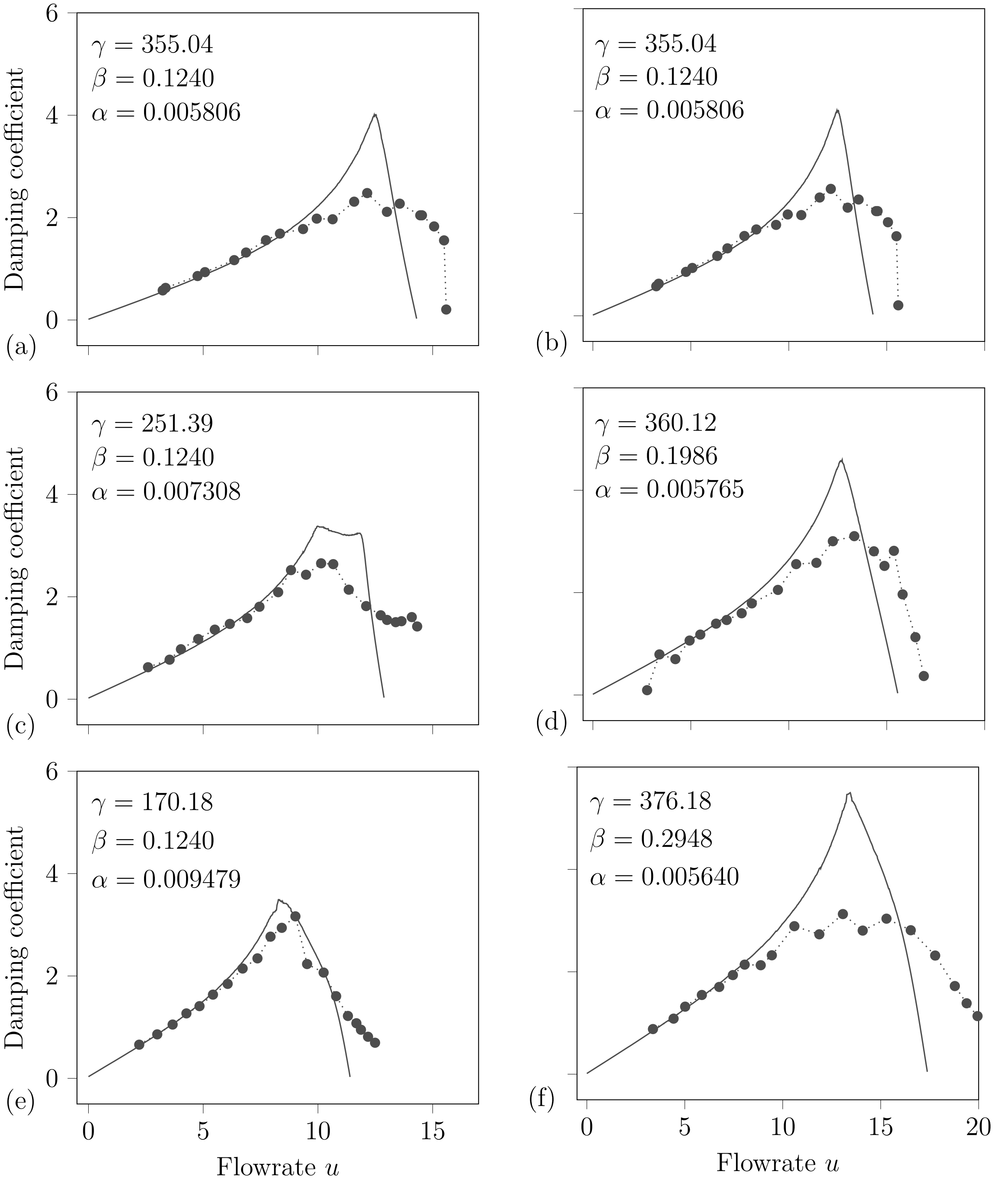


Figure 5.2 Damping: evolution of the tip amplitude logarithmic decrement with the non-dimensional flowrate below critical speed from experimental data (··· ● ···) and from the linear model (—) for different pipes. (a), (c) and (e) represent the evolution with the same inner diameter $d = 6.35$ *mm* and for three lengths $L = 46$ *cm*, $L = 41$ *cm*, and $L = 36$ *cm* and (b), (d) and (f) represent the evolution with the same length $L = 46$ *cm* and three inner diameter $d = 6.35$ *mm*, $d = 7,9375$ *mm* and $d = 9,271$ *mm*.

5.2 Determination of the critical speed

When the flowrate increases further, the pipe destabilizes through flutter instability. Figure 5.3 shows the experimental critical speed at which flutter instability appears for the seven pipes of three different inner diameters. We compare the experimental critical flowrate with theoretical results for the same three inner diameters and on a large range of pipe lengths. These results are obtained by finding the lowest flow velocity at which the real part of one of the eigenvalues obtained is negative as in the literature review. We ploted the critical flowrates with respect to the gravity parameters γ and damping parameter α. Indeed, these parameters are both increasing function of the pipe length.

The critical flowrate is higher when the inner diameter is larger and when the gravity and damping parameters are larger (i.e. when the pipe is longer), which is coherent with the results from Figure 5.2. A longer pipe with a larger inner diameter is more stable. The

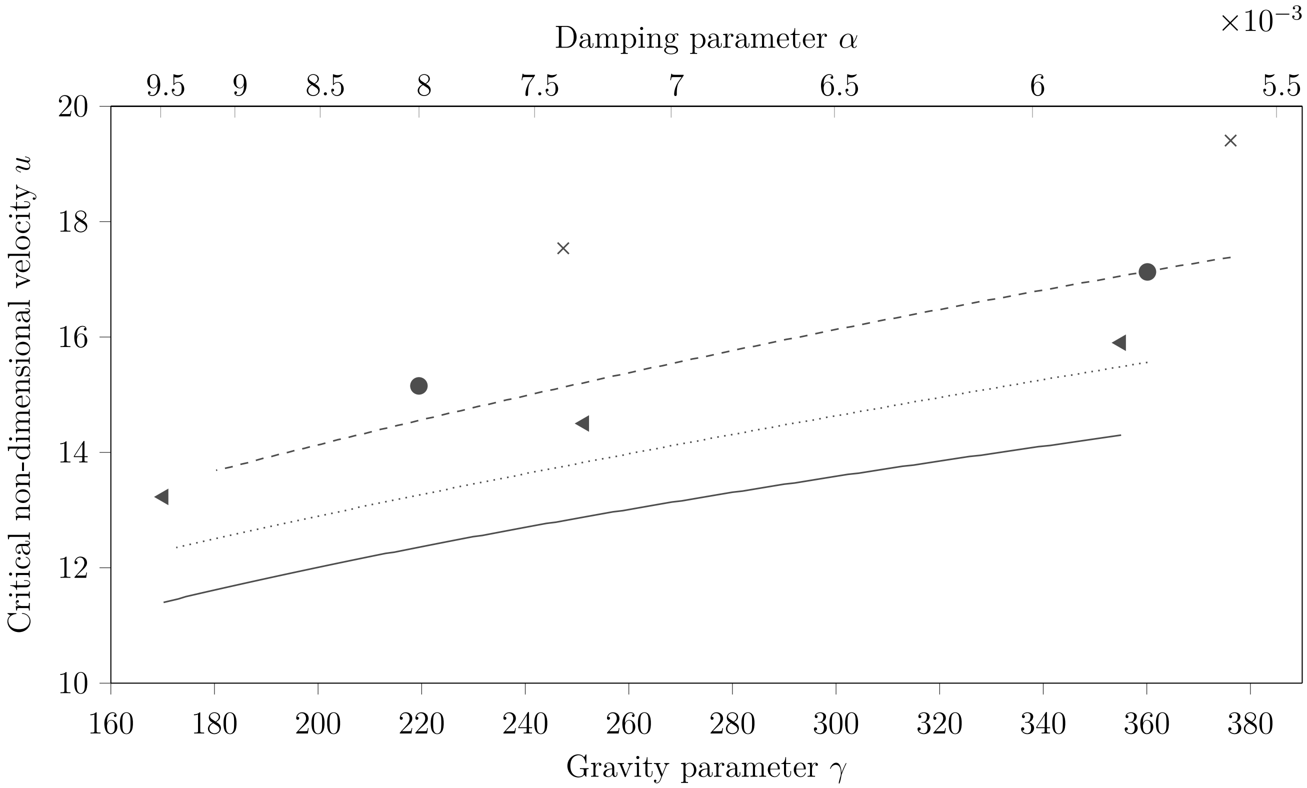


Figure 5.3 Critical flowrate of pipes with variable lengths and different inner diameters from experimental results and the linear model: (◀), (●) and (×) experimental for $d = 6.35\ mm$, $d = 7.9375\ mm$ and $d = 9.525\ mm$ and (—), (······) and (- - -) theoretical for $d = 6.35\ mm$, $d = 7.9375\ mm$ and $d = 9.525\ mm$

theoretical critical speeds are always smaller than the experimental values by a relative difference between 9 and 13% depending on the cases.

This difference is explained by the same reasons that were cited for the damping i.e. some additional damping terms not taken into account in the linear model might delay the destabilization of the pipe. Moreover, the operator has to decide when to consider the pipe unstable, which represents in itself an uncertainty.

As for the damping, the critical speed is higher with a larger inner diameter because the β mass number and the γ gravity parameter are larger which helps stabilize the pipe. Hence, the flow velocity has to be higher to destabilize the pipe. Likewise, longer pipes are more stable and have a higher critical speed since their β and α parameters are higher which is coherent with previous studies [43].

5.3 After the critical speed

After reaching the critical speed, the pipe loses stability by flutter. Both cameras are then needed to follow the pipe.

5.3.1 Qualitative considerations

When we increase the flowrate, we observe different phenomena and the pipe follows different types of motion.

Trace of the tip of the pipe

Figure 5.4 represents the trace of the tip of the pipe in the horizontal (x, y) plane for different pipes and flowrates with interesting phenomena. Figure 5.4(a) shows the trace for a low flowrate above the critical speed. At this flowrate the movement of the pipe is planar and stays in the same plane which is at 45° between the two cameras. It is the preferred vibration plane of the pipe which is caused by the imperfections of the pipe itself and of the fixation. This movement type is called the 2D limit cycle motion. When the flowrate increases further as in Figure 5.4(b), the movement stays plannar but the plane of motion rotates with time. Then, when the flowrate is increased almost to the maximum capability of the system, the pipe motion becomes erratic and completely three dimensional as shown in Figure 5.4(c). For other pipes, Figure 5.4(f) shows a circular three dimensional movement producing an ellipse in a 45° plane. This movement is called the 3D limit cycle motion or the orbital motion. Figure 5.4(d) and (e). also show this cycle motion in a rotating plane as in Figure 5.4(b).

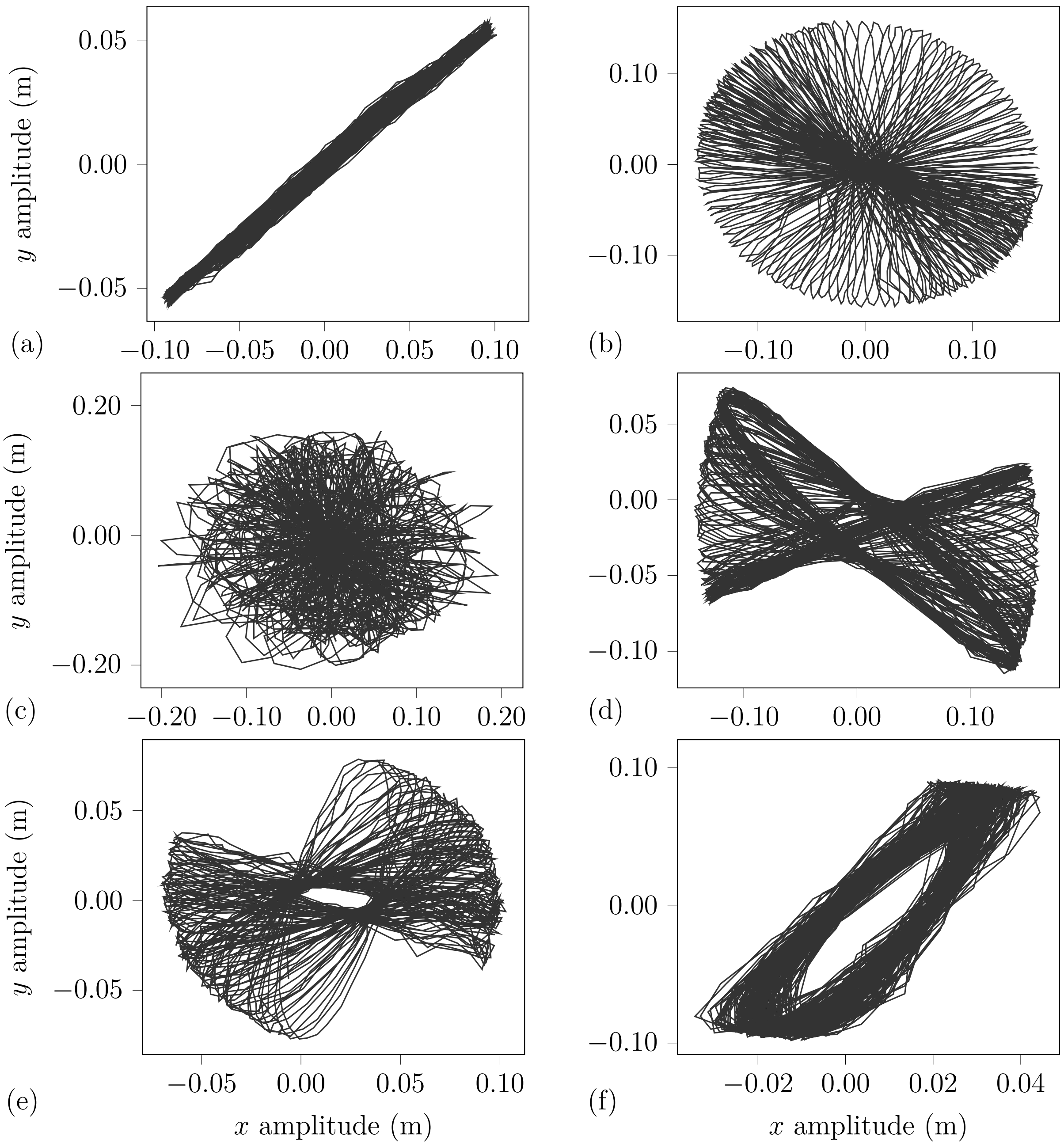


Figure 5.4 Trace of the tip of the pipe in the (x, y) plane for different pipes and reduced flowrates: (a) d = 6.35 mm, L = 46 cm and u = 17.49, (b) d = 6.35 mm, L = 46 cm and u = 22.70, (c) d = 6.35 mm, L = 46 cm and u = 30.21, (d) d = 6.35 mm, L = 41 cm and u = 23.25, (e) d = 7.9375 mm, L = 46 cm, u = 28.70 and (f) d = 9.525 mm, L = 46 cm and u = 63.95.

Snapshots

Another interesting qualitative observation was the deformed shapes of the pipe experiencing flutter. Figure 5.5 shows snapshots of the No.1 pipe in its flutter plane for different flowrates. We used the minimum inner diameter because the other two pipes could not withstand the strain from the highest flowrates for long enough and would break. These images are taken every 0.02 seconds and are obtained after treating the images from one camera. We used the front camera and treated the images to isolate the pipe by creating a binary image with a grey-scale threshold.

Figures 5.5(a) and (b) show classic first mode flutter which is observed at relatively low flowrate and corresponds to the 2D cycle motion. On the first figure we can observe the first mode of the pipe and a combination of the two first modes on the second figure. The amplitude of the flutter vibration is also higher in Figure 5.5(b) compared to the Figure (a). Figures 5.5(c), (d), (e) and (f) show the shapes at the highest flowrate where the 3D cycle motion is observed. The amplitude is higher than for the previous ones and we observe the third pipe mode especially in Figure 5.5(e). Figures 5.5(c) and (d) show unconventional mode shapes as the tip of the pipe goes over the bottom position of the pipe. This phenomena cannot be captured with the polynomial fitting method described in the methodology as there are two points with the same z coordinate.

5.3.2 Quantitative considerations

Amplitude evolution

To evaluate further the impact of the flowrate on the motion of the pipe, we studied the evolution of the amplitude of the tip of the pipe. We firstly plot the peaks amplitude for two different pipes in Figure 5.6. To obtain this figure we used a Python function to retrieve the amplitude peaks of the tip of the pipe from the experimental amplitude data of a 30 seconds long video.

Figure 5.6 represents the upper half part of a bifurcation diagram. Indeed, since the motion is three-dimensional, we plot the peaks of the absolute value of the amplitude defined as $\sqrt{(x^2 + y^2)}$. We therefore only have access to the positive part of the amplitude.

At the lowest flowrate, we observe the first bifurcation when the pipe loses stability by flutter and the amplitude of the pipe deflection increases. The peaks amplitude then stabilizes when the flowrate increases before reaching the second bifurcation. At the second bifurcation, we observe the period doubling phenomenon as the amplitude peaks have a range of different values for the same flowrate. The pipe from Figure 5.6(a) is longer than the pipe from Figure

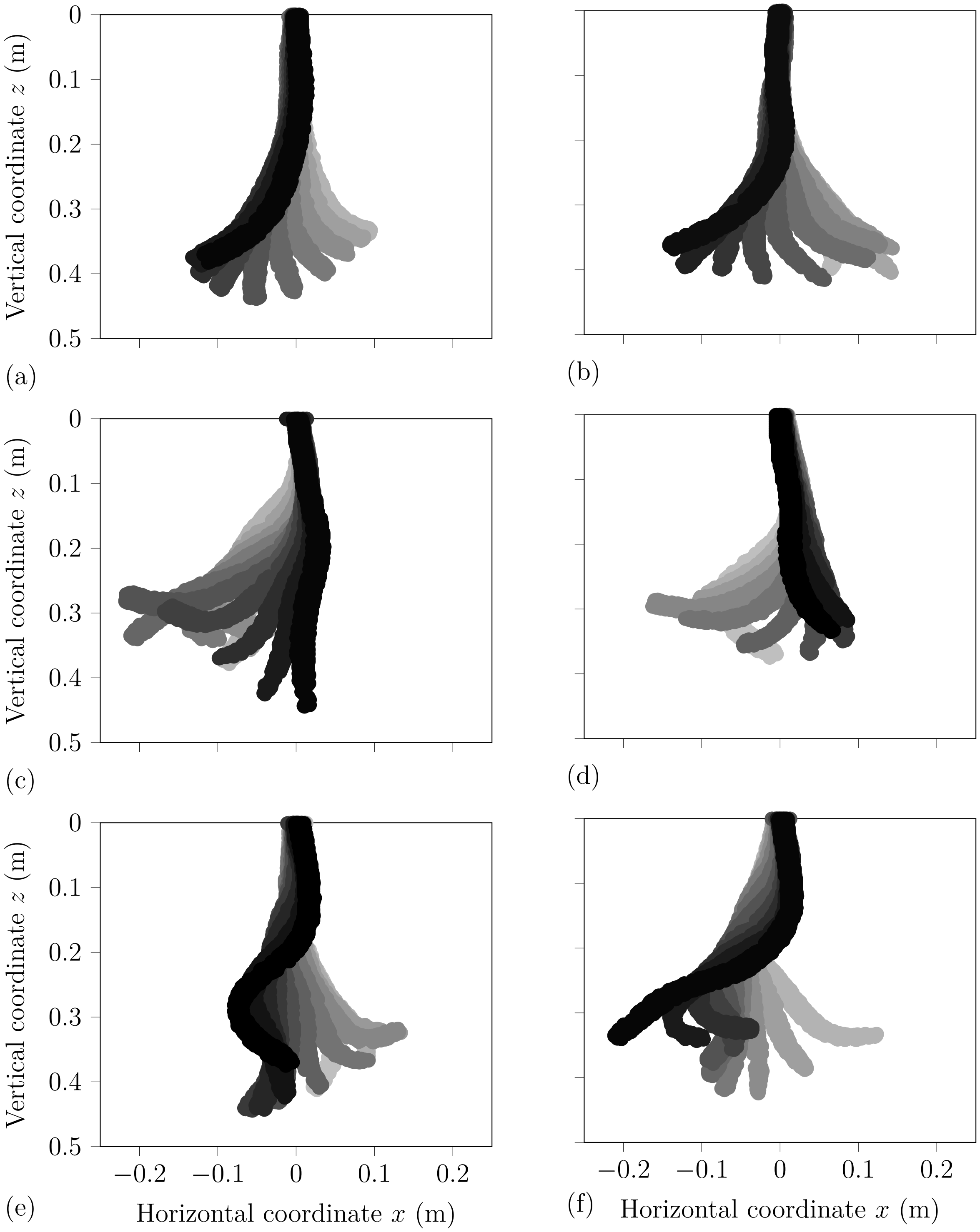


Figure 5.5 Snapchots of the pipe No.1 taken every 0.02 seconds for different non-dimensional flowrates (a): $u = 21.98$, (b): $u = 26.07$ and (c), (d), (e) and (f): $u = 30.20$.

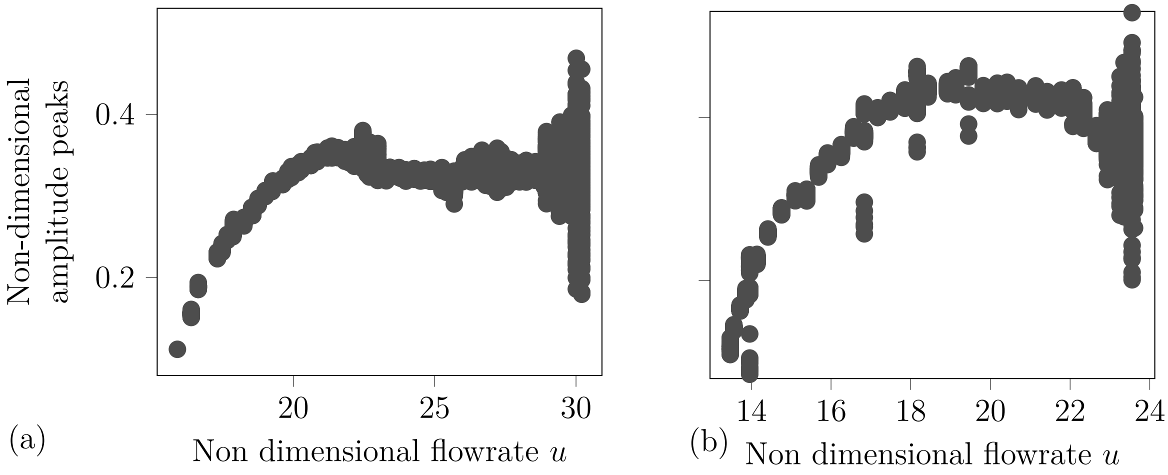


Figure 5.6 Peaks amplitude evolution with the non-dimensional flowrate for (a): pipe No.1 and (b) pipe No.3 from Table 4.2.

5.6(b) and the bifurcations appear later for this first pipe since it is more stable.

To compare the evolution of the amplitude for different pipes more easily we use the average of the amplitude peaks represented in the previous figure. By doing so, we lose information on the period doubling but we can compare the pipes more easily. Figure 5.7 shows the different amplitudes for six different pipes.

In Figure 5.7(a), the three lines represent the amplitude of the pipe for three pipes of same length and different inner diameters, while in Figure 5.7(b), the lines represent the amplitude of three pipes of the same inner diameter and of three different lengths.

Figure 5.7(a) shows that the amplitude increases with the flowrate before stabilizing at higher flowrate. There are two logical explanations for this phenomena. First, at some point, the pipe movement becomes irregular and in this movement mode, the amplitude drops and then does not increase anymore. Another explanation is that, at higher flowrate, the pipe tip tends to be horizontal and sometimes even go above the bottom point of the pipe as in Figure 5.5(c). This causes a problem with the method to extract the position of the pipe with a polynomial function. Indeed, one z height position corresponds to two points on the pipe and the polynomial fit goes in the middle of these two points to minimize the error. The extracted amplitude is then smaller than the actual amplitude because of that error and it explains the amplitude stabilization.

Pipes with smaller inner diameters have a larger amplitude of vibration and reach the amplitude plateau sooner. This can be explained by the fact that these pipes are more unstable because of their smaller β and γ parameters and reach their 3D cyclic motions and then

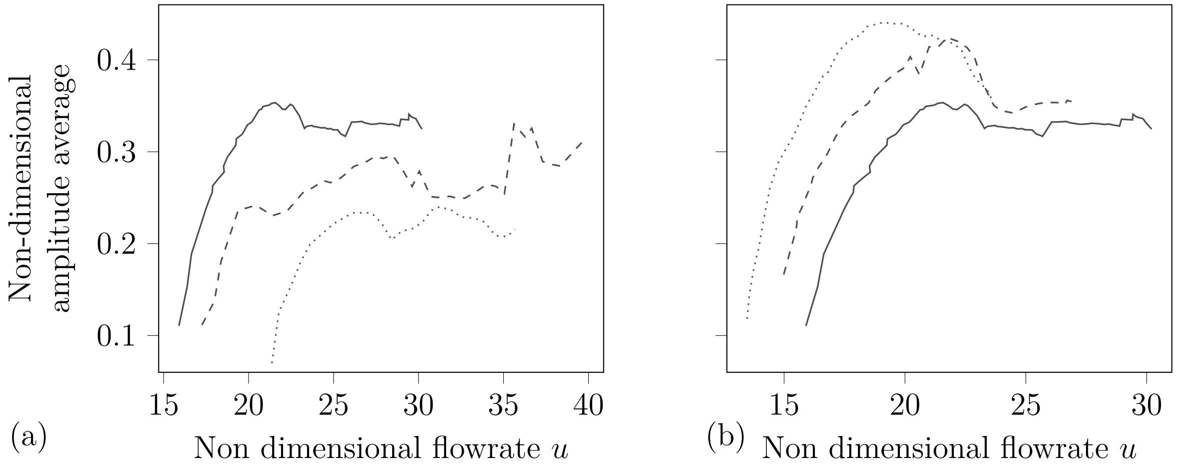


Figure 5.7 Evolution of the average amplitude of the tip of the pipe with the non-dimensional flowrate for different pipes: (a) $L = 46$ cm with three different inner diameters (—) $d = 6.35$ mm, (- - -) $d = 7.9375$ mm and (······) $d = 9.525$ mm and (b) $d = 6.35$ mm with three different lengths (—) $L = 46$ cm, (- - -) $L = 41$ cm and (······) $L = 36$ cm.

erratic motions sooner. The plot corresponding to the largest inner diameter stops before reaching the maximum flowrate. This pipe is much thinner than the others because of its diameter and it is therefore very fragile. During both tests the pipe shredded before reaching the maximum flowrate from the pump.

In Figure 5.7(b), until approximately $u = 22$, the amplitude increases with the flowrate and the longer pipes have a smaller amplitude since longer pipes are more stable because of their larger gravity parameter γ. As in the previous figure, the amplitude reaches a plateau when the movement becomes irregular. It is interesting to notice that shorter pipes have a larger non-dimensional amplitude at the same flowrate. Longer pipes are under more tension which prevents the amplitude from increasing. Indeed, because of the tension, the instability is present at the tip of the pipe and even if the pipe is longer, only a certain portion of the pipe is affected by the instability at the tip.

Frequency evolution

Figure 5.8 follows the evolution of the pipe motion frequency with the flowrate. Figure 5.8(a) shows the frequency for three different inner diameters for pipes of the same length and Figure 5.8(b) represents the frequency for three different lengths and the same inner diameter.

To obtain the frequency, we performed a Fast Fourier Transform (FFT) on the x and y positions of the tip of the pipe along time. We then take the peak of both FFT and its corresponding frequencies to compute the average of both frequencies and to obtain the pipe

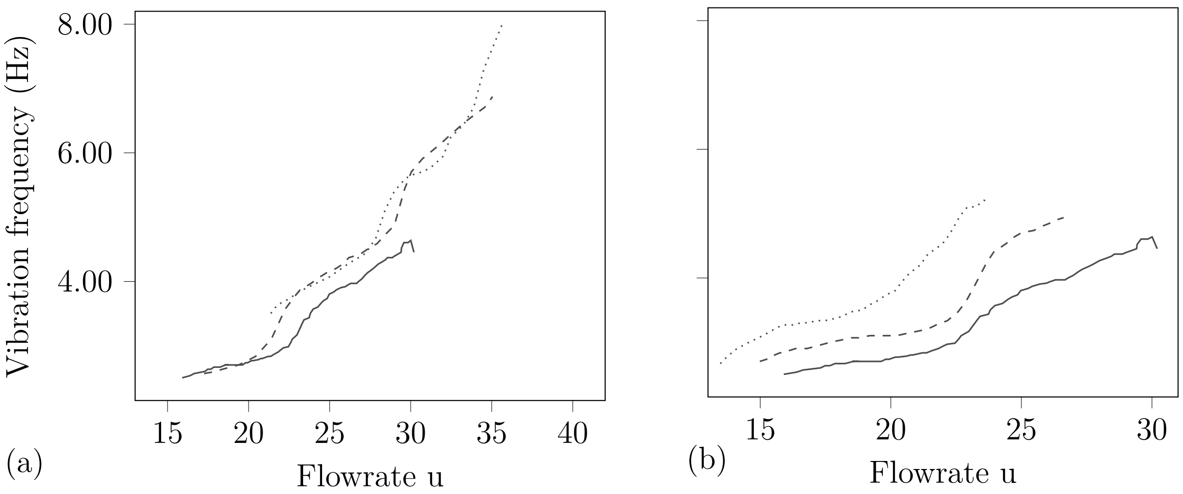


Figure 5.8 Evolution of the frequency of the tip of the pipe with the flowrate for different pipes: (a) L=46 cm with three different inner diameters (—) d=6.35 mm, (- - -) d=7.9375 mm and (······) d=9.525 mm and (b) d=6.35 mm with three different lengths (—) L=46 cm, (- - -) L=41 cm and (······) L=36 cm.

frequency. The peak frequencies of both axis are really close which confirms that this method is appropriate. We then repeat this operation for the different flowrates to obtain the given plots.

In Figure 5.8, the frequency increases linearly with the flowrate. The shorter pipes have a higher vibration frequency and the frequency augmentation with the flowrate is also sharper for these pipes. The inner diameter of the pipe, on the other hand, has limited a impact on the pipe frequency.

CHAPTER 6 METHODOLOGY: DEVELOPMENT OF A PINN MODEL FOR THE PIPE CONVEYING FLUID SYSTEM

In this project, we explore and demonstrate the possibility to use PINN based methods to develop a digital twin of the pipe conveying fluid. To start, we work on the classical linear pipe Equation (2.19) with classic PINNs and modal PINNs. We also develop and use an alternative formulation using a beam mode shapes decomposition and the weak-form of the PDE. Other promising and more advanced PINN formulations will be studied in the follow-up of this project.

6.1 General PINN architecture

All the codes are written in Python 3.6 with the TensorFlow 1.14.0 library which allows us to construct the DNN with variable weights and biases [64]. TensorFlow uses automatic differentiation to compute the terms of the PDE but also the gradients used for the training of the DNN.

In all the following examples, deep feedforward and fully connected neural networks approximate the function of interest. If no contrary information is given, all the NN are made of 3 hidden layers, each composed of 20 neurons. The output and input layers dimensions of the NN depend on the type of PINN. Longer and wider NNs were used without improving the precision and were found to significantly slow down the training process. To make sure that this NN was of the appropriate size, we had to verify that it could approximate the objective function (the deflection of the pipe along time) at a satisfactory level. This procedure is detailed in Appendix C. The general idea of this size study is to use different NNs of different sizes and to train these NNs with only data and no residuals points. We start with small NNs and increase the size until the minimum training loss is not decreasing any more. The activation functions in the NNs are always tanh function and this choice is explained in Appendix B. We compared the results from sin, tanh and ReLU activation functions for all the PINNs architecture and determined the most efficient one. The biases of the NNs are always initialized to 0 while the weights are initialized with the Xavier method that was used in the literature [22, 65]. It is not recommended to initialize the weights to zero to avoid a null gradient value for optimization and this initialization associates random values to the weights with a normal law centred on zero and truncated for values smaller or greater than two standard deviations. The standard deviation of the normal law is equal to $\sqrt{\frac{2}{N_{in}+N_{out}}}$ for each layer where N_{in} and N_{out} are the sizes of the input and output of the layer.

The training is performed using the Adam optimizer which is widely used in the literature for PINNs [29]. This gradient descent algorithm is implemented directly in the TensorFlow library. It is well suited for large data sets and high dimensional problems which make it a good algorithm to train a DNN. It uses moment estimates of the gradient to add inertia to the optimization and to increase the convergence speed. The two parameters used for moments estimation are set to their defaults values which were found to be the most effective. The epsilon numerical stability parameter is also set to its default value. The last parameter of this optimization algorithm is the learning rate which plays an important role in the convergence. If the learning rate is too small, the optimization converge to a local minimum and does not converge to the expected solution. On the contrary if it is too large, the optimization starts to oscillate when close to the solution. To chose the best learning rate value, we started by setting it at a high value of $L_r = 10^{-3}$ and decreased it until no oscillations were observed at the end of the optimization. For all the following cases, the optimal value was found to be $L_r = 10^{-5}$. This optimization algorithm performs one step of the minimization each time it is called on the loss function. It is therefore possible to control the training loop and its termination by setting a maximum number of iteration and an objective value of the training loss at which to stop the training. These values differ according to the case and are specified for each cases. This training algorithm is considered stochastic when randomly chosen mini batch of data are used for training at each iterations. In our case, we used the whole batch of data at each iteration and we checked for possible over-fitting by adding a separate test batch of data which was not used for training and on which the loss function was computed at each iteration of the training. This test loss is supposed to be slightly larger than the training loss. However, if the test loss is significantly larger than the training loss and if this test loss is increasing with the training iterations then it means that the training is over-fitting. It is important to note that we did not use any validation data set during the training. Indeed, PINNs are not prone to over-fitting because of the PDE regularization and a test data set to verify that hypothesis was enough. Random mini batch and adaptive learning rate should be considered during the future to improve the convergence and avoid local minima.

We also tested another minimizing algorithm which is a second order algorithm as opposed to Adam. The L-BFGS algorithm is a quasi-Newton optimization algorithm implemented in TensorFlow. However, this algorithm was keen to get stuck in local minima after only a few iterations and was offering no control on the training loop. It was therefore decided not to use it.

For all the following cases, the loss function is only composed of two terms: the residuals from the linear partial differential equation of the pipe and the data points on the deflection of the pipe. Indeed, since all the following cases use either experimental data or numerically generated data, it was not necessary to use a loss term for the initial and boundary conditions thus demonstrating the capability of PINNs to solve ill posed problems without boundary conditions. However, the boundary conditions at the origin of the pipe were properly enforced with a function multiplying the output of the NNs [37]. The function used is $f_{bc} : z \to \tanh{(z)}^2$ since this function and its first derivative are equals to zero for $z = 0$ hence verifying the clamped boundary conditions of the cantilevered pipe. Moreover, this function is mostly constant after 0 and is not influencing strongly the training of the NNs.

The global loss function is built using a coefficient giving more importance to the data in some cases. Indeed, it was noted that the training sometimes get stuck in a local minima from the trivial solution that verifies the partial differential equation since there is no forcing term. This parameter is chosen empirically by following this procedure: Firstly, set the weight at 0.5, hence giving equals weights to the data and the PDE and then reduce this weight through an iterative process to give more importance to the data until the minimization does not prefer the trivial solution. Depending on the case, the optimal value is different and is specified for each case. The global loss has the following formulation:

$$\mathscr{L} = \alpha_s \mathscr{L}_s + (1 - \alpha_s)\mathscr{L}_d, \tag{6.1}$$

where $\mathscr{L}_s$ and $\mathscr{L}_d$ are the loss terms respectively associated to the residuals of the PDE and to the data.

6.2 Classic PINN

Here, Figure 6.1 presents in details the specific architecture used for the classic PINN. The neural network has two inputs which are the z and t coordinates (or ξ and τ in the non-dimensional formulation). The output of the neural network represents the planar deflection x (or η in the non-dimensional formulation) of the pipe at a given set of coordinates (z, t). This output is then multiplied by $tanh(z)^2$ to force to zero the deflection and its first derivative with respect to the z dimension at $z = 0$. We did not use a second function to force the boundary conditions at the tip of the pipe since it means using complex function that might have an impact on the PINN training and convergence. Figure 6.1 describes the construction of the data loss at the bottom and of the residuals loss on the top which are used to train the weights and biases of the NN. The dashed arrows correspond to the feedback loops used only during training to optimize the weights and biases and to minimize the loss function.

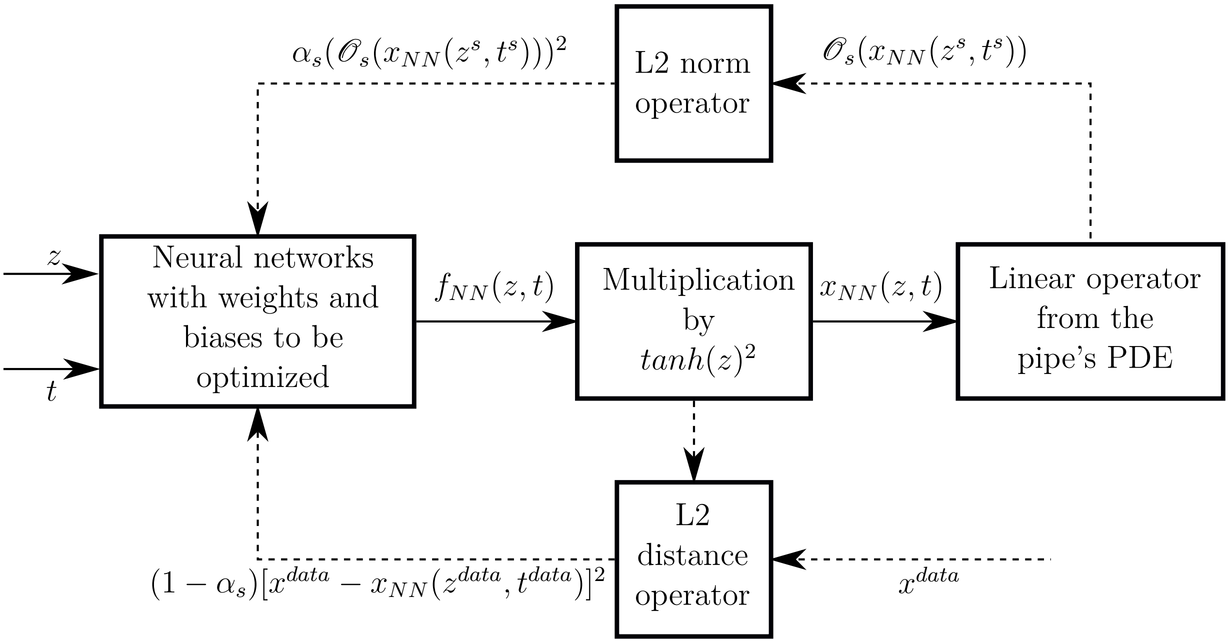

$\alpha_s(\mathscr{O}_s(x_{NN}(z^s,t^s)))^2$
L2 norm operator
$\mathscr{O}_s(x_{NN}(z^s,t^s))$
z
t
Neural networks with weights and biases to be optimized
$f_{NN}(z,t)$
Multiplication by $tanh(z)^2$
$x_{NN}(z,t)$
Linear operator from the pipe's PDE
L2 distance operator
$(1-\alpha_s)[x^{data} - x_{NN}(z^{data},t^{data})]^2$
x^{data}

Figure 6.1 Classic PINN architecture to model the pipe conveying fluid

6.2.1 Classic PINN with numerical data

While waiting for the experimental setup to be ready, we worked with numerically generated data. These data were generated using the Equation (2.41) derived in the literature review. We use the non-dimensional parameters corresponding to the pipe No.1 from Table 4.2. The flowrate is fixed at a certain value for the whole PINN model and cannot be changed later without retraining the model. We first choose the number of data points N_{data} that we want to generate. For each point, we randomly select a ξ^{data} non-dimensional vertical coordinate between 0 and 1 and a τ^{data} timestamp between 0 and τ_{max}. We then compute the non-dimensional deflection η^{data} for each point using Equation (2.41), the selected flowrate and the pipe parameters. This constitutes the data set that is used for the data part of the loss:

$$\mathscr{L}_d = \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} (\eta_i^{data} - \eta_{NN}(\xi_i^{data}, \tau_i^{data}, W, b))^2. \tag{6.2}$$

For the penalization of the PDE, we proceed likewise: we start by choosing the number of points N_s on which we penalize the PDE and then select the associated ξ^s and τ^s coordinates randomly. From the non-dimensional PDE Equation (2.20) and the strong form residuals loss function in Equation (2.9), we write the loss as follows:

$$\begin{aligned}\mathscr{L}_s =&\frac{1}{N_s}\sum_{i=1}^{N_s}\Big\{\alpha\dot{\eta}_{NN}^{''''}(\xi_i^s,\tau_i^s)+\eta_{NN}^{''''}(\xi_i^s,\tau_i^s)+\Big(u^2-\gamma\left(1-\xi_i^s\right)\Big)\,\eta_{NN}^{''}(\xi_i^s,\tau_i^s)\\ &+2\beta^{1/2}u\dot{\eta}_{NN}^{'}(\xi_i^s,\tau_i^s)+\gamma\eta_{NN}^{'}(\xi_i^s,\tau_i^s)+\ddot{\eta}_{NN}(\xi_i^s,\tau_i^s)\Big\}^2.\end{aligned} \tag{6.3}$$

In this case, the ideal weight α_s on the loss function parts was found to be $\alpha_s = 0.01$.

Finally, we build a test data set and a test loss function to check that the training is not over-fitting on the data. To do so, we randomly picked N_s^{test} and N_{data}^{test} penalization points and data points and then we construct the test loss function with these points and the same α_s coefficient.

6.2.2 Classic PINN with experimental data

Once the experimental setup was complete, we started to use data from this setup instead of using numerical data. To acquire the data we proceeded as follows:

- Chose a flowrate below the critical flowrate. Indeed, with the linear model we can only study the pipe behaviour before the destabilization.
- Start the acquisition of the videos and of the flowrate.
- Wait for 30 seconds to measure the flowrate before the perturbation.
- Induce a perturbation and wait until the pipe goes back to its rest position.
- Trim the video to store the part after the perturbation and we store the flowrate value.
- Analyze the video to extract the polynomial function fitting the pipe position on each frame.

With the experimental data, we used the dimensional PDE of the pipe (2.19). We selected the PDE penalization points as before by randomly choosing N_s coordinates z and t respectively between 0 and L and between 0 and the end of the video.

For the data points, we chose a number of points N_{data} and for each points, we select a random frame (with its corresponding t values) and we select a random z value between 0 and L. We use the polynomial fitting the pipe position associated to the frame to compute the x deflection value.

We write the two parts of the loss function as before with the dimensional expressions:

$$\mathscr{L}_d = \frac{1}{N_{data}} \sum_{i=1}^{N_{data}} (x_i^{data} - x_{NN}(z_i^{data}, t_i^{data}, W, b))^2, \tag{6.4}$$

for the data loss and:

$$\begin{aligned} \mathscr{L}_s =& \frac{1}{N_s} \sum_{i=1}^{N_s} \left\{ \frac{\partial^5 x_{NN}(z_i^s, t_i^s)}{\partial t \partial z^4} + EI \frac{\partial^4 x_{NN}(z_i^s, t_i^s)}{\partial z^4} \right. \\ &+ \left\{ MU^2 - g\left(M+m\right)\left(L - z_i^s\right) \right\} \frac{\partial^2 x_{NN}(z_i^s, t_i^s)}{\partial z^2} \\ &\left. +2MU \frac{\partial^2 x_{NN}(z_i^s, t_i^s)}{\partial t \partial z} + g\left(M+m\right) \frac{\partial x_{NN}(z_i^s, t_i^s)}{\partial z} + \left(M+m\right) \frac{\partial^2 x_{NN}(z_i^s, t_i^s)}{\partial t^2} \right\}^2, \end{aligned} \tag{6.5}$$

for the PDE loss. However since the PDE is dimensional, the scaling is not the same and we had to find a new optimal value of the weight on the loss parts $\alpha_s = 0.08$.

We also construct the test data set as explained previously.

6.2.3 Inverse problems with a classic PINN

With both numerical and experimental data, it is possible to solve inverse problems by using a classic PINN. Basically, we can determine the value of any parameters of the pipe PDE. To do so, we define the parameter as a variable and initialize it at a certain value. Then, during the training process, the optimizer optimizes the variable parameter to minimize the residuals loss while fitting the data points to minimize the data loss. Once the training is complete, we access the value of the variable which is the solution of the inverse problem. However, defining too many parameters as variables can be deleterious to the convergence of the training. Indeed, it creates more degrees of freedom and more possible local minima. We also observed that we need more data when parameters are set as variables which was expected as the prior knowledge, reducing the need for data, is not as complete.

In this study, we worked on the inverse problem by setting the flow velocity as a variable. It is interesting since we can then verify the value of the obtained optimized flowrate with the actual measured flowrate. In every cases, we initialized the flow velocity at $U = 1$ m/s and constrained it to stay between 0 and 10 m/s.

Lastly, when working with more than one variable parameter on the PDE, it is important to work with an adequate scaling. For example if the γ and α parameters are set as variables it is suggested to define the $\alpha_{variable}$ as $\alpha_{variable} = 1000 \times \alpha$ to make sure that $\alpha_{variable}$ and $\gamma_{variable}$ are of the same order of magnitude.

6.3 Modal PINN

In this section, we use a modal PINN to model the pipe behaviour [39]. It is a very appropriate formulation for the pipe as its deflection follows a sinusoidal function in time. We suppose that the deflection of the pipe along time has the following shape:

$$x(z,t) = \sum_{k=1}^{N} \phi_k(x) e^{j\omega_k t}, \tag{6.6}$$

where $(\omega_i)_{1 \leq i \leq N}$ are the system eigenfrequencies and $(\phi_i)_{1 \leq i \leq N}$ are the unknown spatial mode shapes.

The architecture of the Modal PINN is shown in Figure 6.2. The neural network has only one input, the space coordinate z (or ξ for the non-dimensional formulation). The neural network has N outputs which are the space mode shapes with respects to the decomposition presented in Equation (6.6). These N mode shapes are also multiplied by $tanh(z)^2$ since they each have to respect the boundary conditions at the origin of the pipe. On another part of the PINN, the N time functions are computed with their analytical formula given the time variable t (or τ for the non-dimensional formulation) and the N eigenfrequencies. This function returns the N time functions. Both the outputs of the NN and of the time function are then multiplied and added according to the Equation (6.6). The result from this operation is then used to compute the two parts of the loss function as with classic PINNs. As with the previous figure, the dashed arrows correspond to the feedback loops used during training to optimize the weights and biases. The time functions are well known and the time derivatives are thus computed manually which improves the training speed. The eigenfrequencies have to be given to the PINN model to compute the time functions as there is no formula to compute them analytically.

6.3.1 Modal PINNs with numerically generated data

We firstly worked with modal PINNs using numerical data as for the classic PINNs formulation. The data generation and the construction of the training and test data sets follow the same principle as for the classic PINNs. We used the first pipe characteristics from Table 4.2 and worked with the non-dimensional PDE. The loss function has also the same formulation as in Equation (6.2) and (6.3), however the derivatives with respect to time are encoded manually in the loss function instead of using automatic differentiation. The optimal weight on the two loss parts was also found to be $\alpha_s = 0.01$. The main difference from the precedent approach is that we need to know the eigenfrequencies of the pipe. In this case, these eigen-

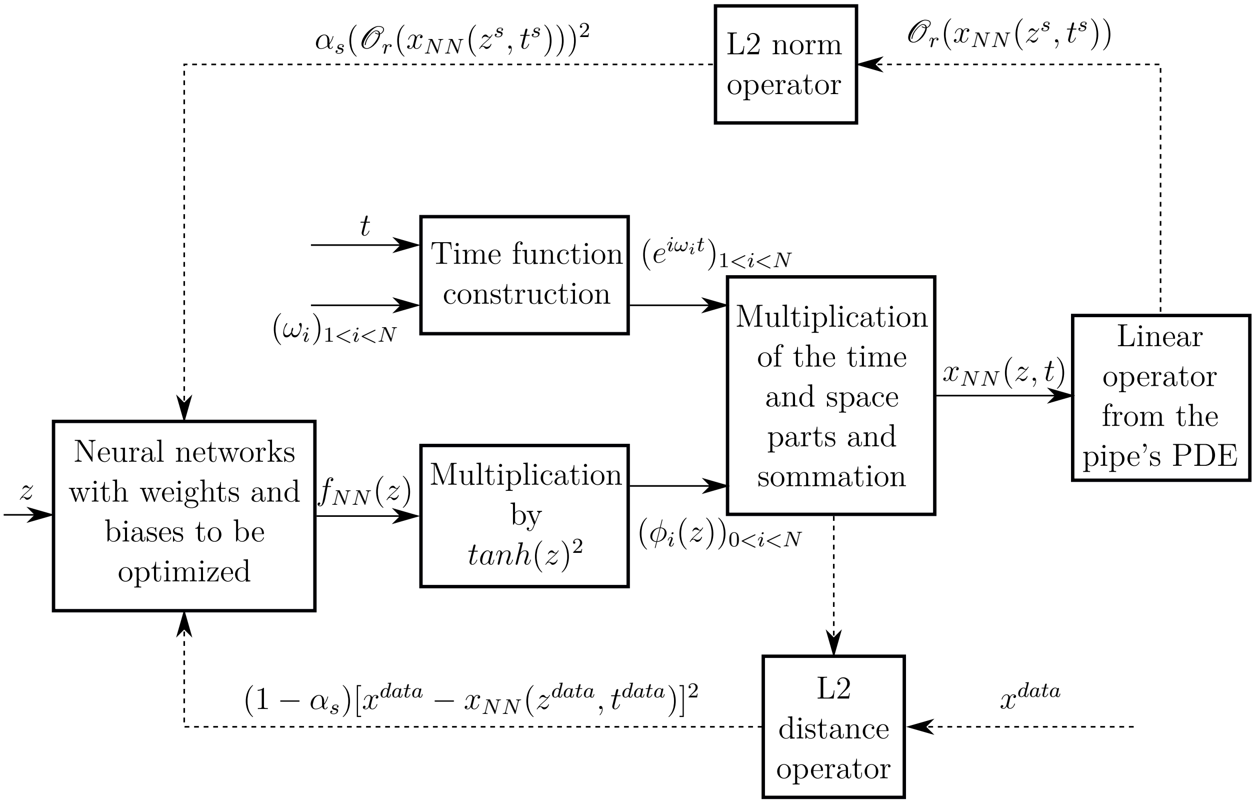

$\alpha_s(\mathscr{O}_r(x_{NN}(z^s, t^s)))^2$
L2 norm operator
$\mathscr{O}_r(x_{NN}(z^s, t^s))$
t
Time function construction
$(e^{i\omega_i t})_{1<i<N}$
$(\omega_i)_{1<i<N}$
Multiplication of the time and space parts and sommation
$x_{NN}(z,t)$
Linear operator from the pipe's PDE
z
Neural networks with weights and biases to be optimized
$f_{NN}(z)$
Multiplication by $tanh(z)^2$
$(\phi_i(z))_{0<i<N}$
$(1-\alpha_s)[x^{data} - x_{NN}(z^{data}, t^{data})]^2$
L2 distance operator
x^{data}

Figure 6.2 Modal PINN architecture to model the pipe conveying fluid

frequencies are numerically obtained with Equation (2.37) and are also used to generate the numerical data.

6.3.2 Modal PINNs with experimental data

To use experimental data with modal PINNs, we needed to measure the complex eigenfrequencies of the system. Indeed, it makes no practical sense to compute numerically the eigenfrequencies to use experimental data afterward. Therefore, we tried to measure the eigenfrequencies by applying a Fast Fourier Transform on the experimental evolution of the tip of the pipe with respect to time. However, this approach failed to capture more than one mode and we could not apply modal PINNs with experimental data.

6.4 Weak form approach

This approach shown in Figure 6.3 is inspired by a combination of the modal PINNs and the weak-form. We suppose that the deflections of the pipe have the following shape:

$$x(z,t) = \sum_{i=1}^{N} \phi_i(z) q_i(t), \tag{6.7}$$

where $(\phi_i)_{1 \leq i \leq N}$ are beam mode shapes and $(q_i)_{1 \leq i \leq N}$ are unknown functions to approximate with the neural network. It follows the same idea as modal PINNs, but instead of assuming the shape of the time function and using the NN to determine the space function, we assumed that the space function is known and that the time functions have to be determined by the NN. The part of the loss function associated to the data has the same formulation as for the other methods. The main difference with modal PINNs is the weak formulation for the residuals loss part. By introducing the solution from Equation (6.7) in the non-dimensional Equation (2.20), we obtain the weak formulation by multiplying it by ϕ_j for $1 \leq j \leq N$, by integrating the resulting N equations between 0 and 1 on the ξ domain and by following the operations from the literature review to solve the linear model. After the operations, we obtain the following weak formulation with the beam mode shapes decomposition:

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{S}\dot{\boldsymbol{q}} + \boldsymbol{K}\boldsymbol{q} = \boldsymbol{0}, \tag{6.8}$$

which is used as the residuals equation. We write the loss part associated to the residuals as:

$$\mathscr{L}_w = \sum_{i=0}^{N_s} \left(\boldsymbol{M}\ddot{\boldsymbol{q}}(t_i^w) + \boldsymbol{S}\dot{\boldsymbol{q}}(t_i^w) + \boldsymbol{K}\boldsymbol{q}(t_i^w) \right)^2, \tag{6.9}$$

where N_w is the number of data points penalizing the matrix equation and t_i^w are the time coordinates at which to penalize the equation. The part of the loss corresponding to the data as the same form as shown in Equation (6.2). This method only uses second order derivatives in the residuals formulation which limits numerical derivations errors. The main advantage of this formulation compared to modal a PINN is that a weak-form PINN does not need the eigenfrequencies of the system to work. Moreover, the residuals equation is only a function of the time and no integrals are present, which means that on the contrary to most weak form approaches, we do not have to estimate integrals with the quadrature rule. In fact, it can still be considered as a strong formulation on the time domain as we have to penalize this new equation on the time domain since we only integrated on the space domain.

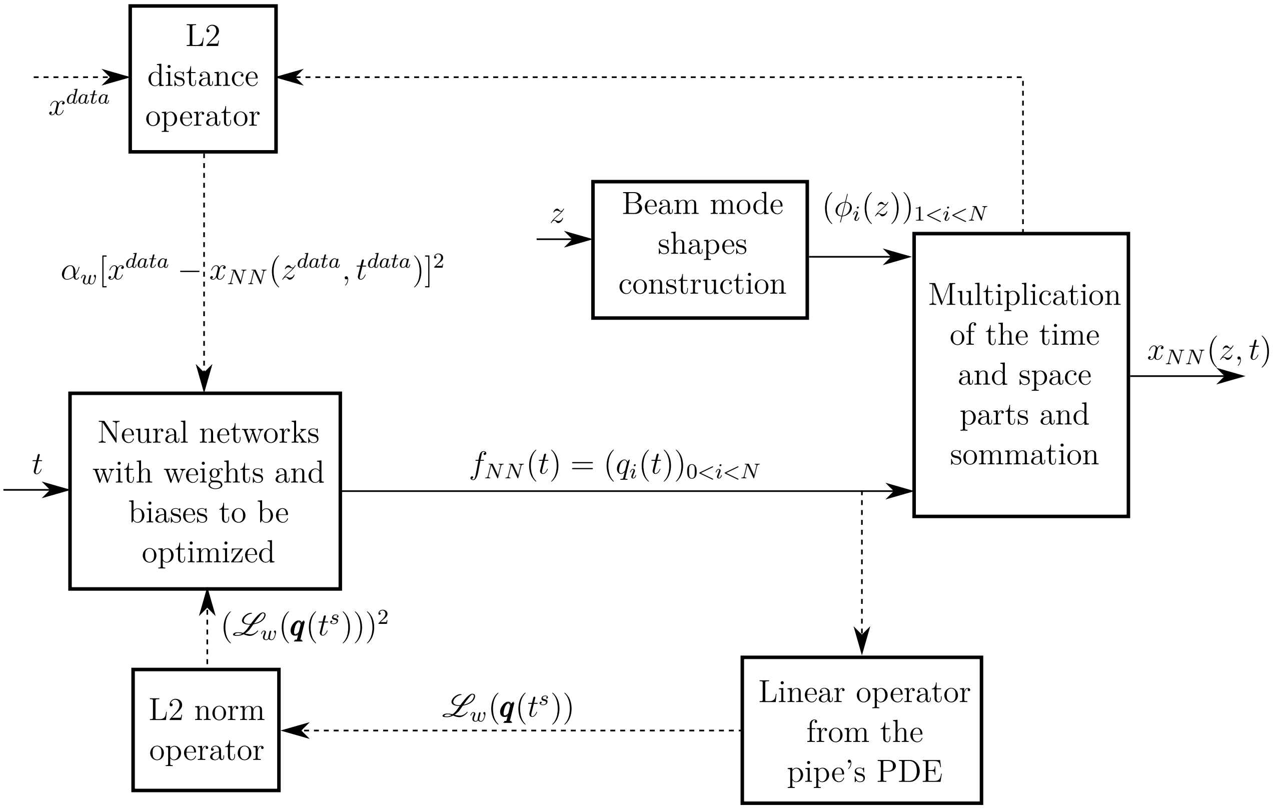

L2 distance operator
x^{data}
$\alpha_w[x^{data} - x_{NN}(z^{data}, t^{data})]^2$
z
Beam mode shapes construction
$(\phi_i(z))_{1<i<N}$
Multiplication of the time and space parts and sommation
$x_{NN}(z,t)$
t
Neural networks with weights and biases to be optimized
$f_{NN}(t) = (q_i(t))_{0<i<N}$
$(\mathscr{L}_w(\boldsymbol{q}(t^s)))^2$
L2 norm operator
$\mathscr{L}_w(\boldsymbol{q}(t^s))$
Linear operator from the pipe's PDE

Figure 6.3 Weak-form PINN architecture to model the pipe conveying fluid

6.4.1 Weak-form PINN with numerical data

This method was firstly used with numerically generated data. The data and penalization points are randomly selected and computed as for the previous method. The only difference is that only time coordinates are picked for the residuals part of the loss since Equation (6.9) is only dependant of time. However, the value of the loss part from the penalization of Equation (6.9) is really high during the first iterations of the training after the initialization of the NN. This high value compared to the data part of the loss strongly disrupts the convergence of the optimization and forced us to use a very strong weight on the two loss parts. Indeed, we decreased the weight on the residuals part down to $\alpha_w = 10^{-7}$ to achieve convergence and avoid the trivial solution local minima. However, this very small weight also reduces the effectiveness of the PDE penalization later in the training. This formulation would be greatly improved by the implementation of adaptive loss weights that would be low at the beginning of the training and increase during the training once the optimization has passed the trivial solution. Despite this issue, we believe this formulation to be interesting especially for problems with forcing terms where the PDE does not admit a trivial solution.

6.4.2 Weak-form PINNs with experimental data

Because of the difficulties to achieve convergence and of the time constraint we did not use the weak-form PINNs formulations with experimental data. Indeed, experimental data are noisy and not in complete agreement with the linear model, thus making the convergence difficult of an already sensitive optimization problem. However, this formulation is more adequate than modal PINNs for the use of experimental data as it does not require the knowledge of the eigenvalues of the problem.

CHAPTER 7 PINN RESULTS AND DISCUSSION

This project serves as an introduction to the use of PINNs to construct a digital twin of the pipe conveying fluid. In this section, we present results obtained with a classic PINN, a modal PINN and a weak-form PINN applied with numerical and experimental data.

7.1 Classic PINN, modal PINN and weak-form PINN comparison

We firstly demonstrate the effectiveness of PINNs to represent the pipe behaviour and to compare a classic PINN, a modal PINN and a weak-form PINN. Then, we show how the addition of the physical model improves the accuracy, reduces the need for data and reduces the over-fitting phenomenon. Finally, we demonstrate the advantage of modal PINNs and weak-form PINNs against classic PINNs.

7.1.1 Impact of the penalization of the PDE

For these purposes, we first trained PINN models with deflection data numerically generated for $U = 2$ m/s. The deflection data used for training are obtained as explained in the methodology and are represented in Figure 7.1.

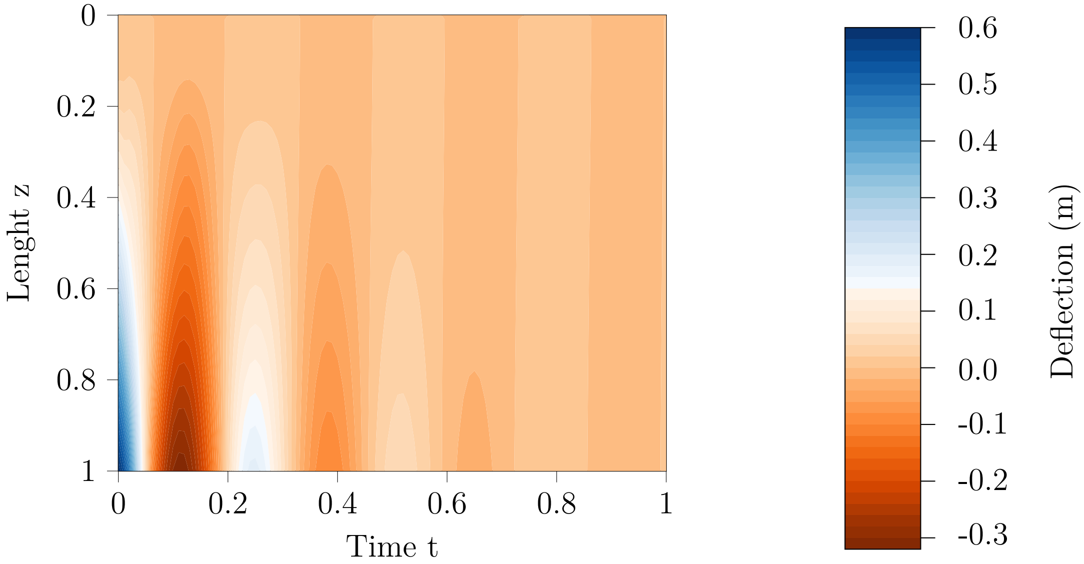


Figure 7.1 Numerical deflection from the linear model used for PINN models trained with numerical data.

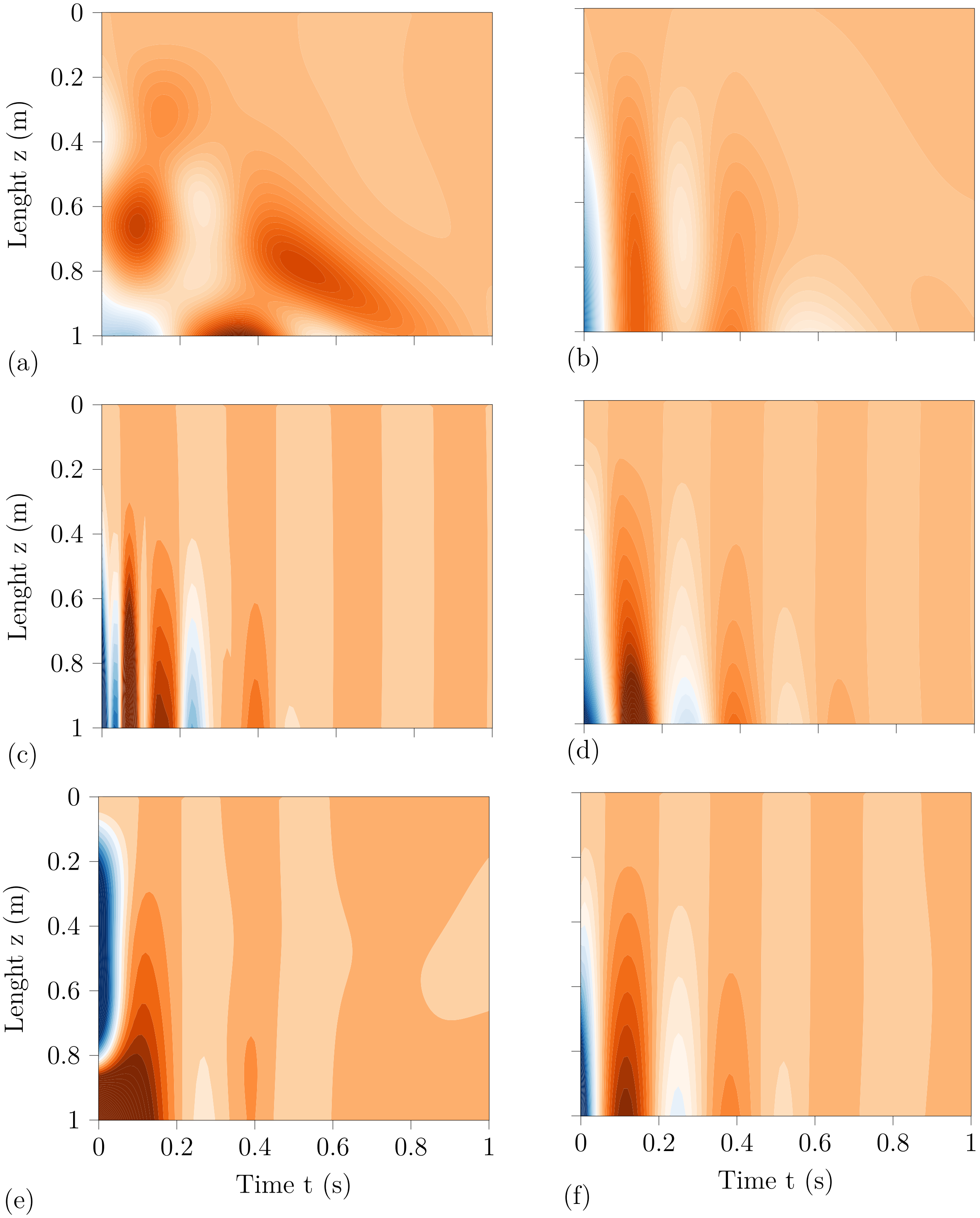


Figure 7.2 Deflection of the pipe from different PINN models with 50 data points: (a) and (b) classic PINN with 5 and 200 residuals points, (c) and (d) modal PINN with 5 and 200 residuals points and (e) and (f) weak-form PINN with 5 and 200 residuals points.

We used the three PINN formulations with 50 data points and changed the number of residuals penalization points from 1 to 300 points. The test set is composed of 1000 points. The models were trained to reach a training loss of 2×10^{-5}. Figure 7.2 shows the deflection of the pipe computed by the classic, modal and weak-form PINN models with small and large sets of penalization points of the PDE. The deflection obtained with large penalization sets is closer to the actual deflection from Figure 7.1 demonstrating the added value of the prior knowledge from the PDE. We also see that the weak-form PINN and the modal PINN in Figures 7.2(c), (d), (e) and (f) are more effective and precise than the classic PINN in Figure 7.2(a) and (b).

To compare these methods more accurately, Figure 7.3(a) represents the test loss reached at the end of the training for each models with the different penalization sets and Figure 7.3(b) shows the number of iterations needed to train the models. The results are obtained by computing the average of ten training for each case.

Figure 7.3(a) demonstrates that the test loss is reduced for weak, modal and classic PINNs when the penalization set is larger. Furthermore, the test loss values and evolution are similar for weak, modal and classic PINNs. Figure 7.3(b) indicates that when the size of the penalization set is larger, the number of iterations needed to reach the termination criterion of the training increase as well. However, the increase is much steeper for weak-form PINNs than classic PINNs and the iterations required by the modal PINNs models are even lower. The large number of iteration needed for weak-form PINNs to converge illustrate the difficult convergence of this method. This disadvantage could be improved by implementing adaptive weights on the loss parts.

From this trial, we demonstrate that the prior knowledge from the residuals helps improve the PINNs precision in low data conditions. The error induced by the gap in the data is countered by the regularization effect of the physical knowledge which reduces over-fitting. The integration of the prior knowledge with weak-forms PINNs, modal PINNs and classic PINNs has similar effects on the accuracy of the results. However, the addition of this knowledge means a longer training as the PINNs models are more heavily constrained. In this matter, modal PINNs are better than classic PINNs which are better than weak-form PINNs as they need less iterations to converge and their training time is less affected by the penalization set size. Finally, the reduction of the test loss with the increase of penalization points is much slower when it reaches 200 points which means that it is not interesting to increase further the penalization set as it would increase the training time for little benefit.

However, the time for each iteration is not studied here and is also an important factor. Indeed, the number of iteration needed to reach the objective value of the loss cannot be the

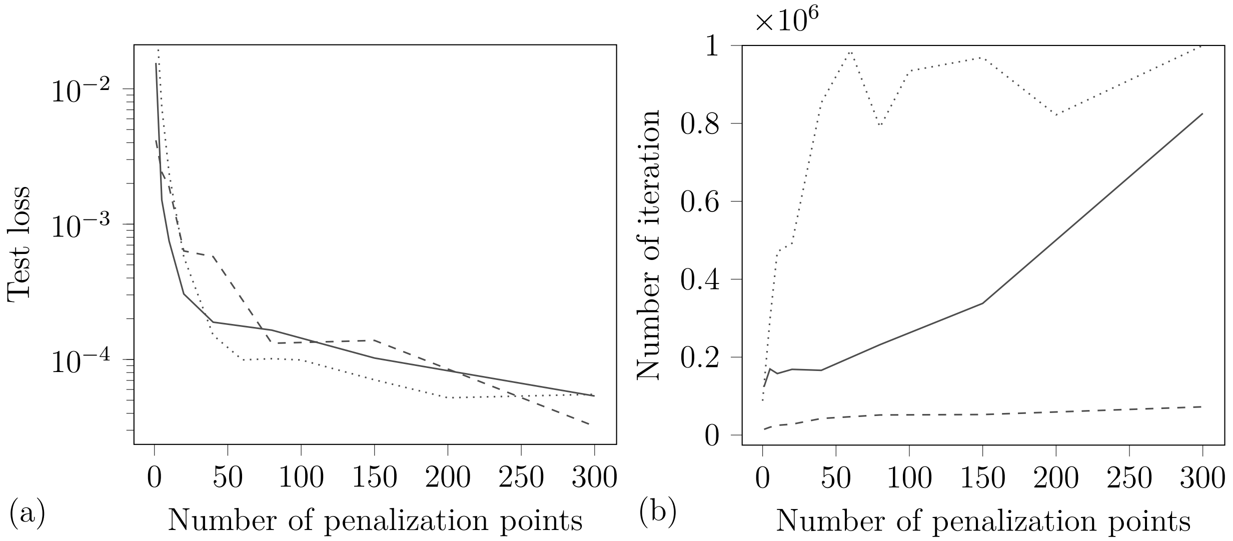


Figure 7.3 Influence of the PDE penalization for (- - -) modal (with 50 data points), (——) classic PINNs (with 50 data points) and (······) weak-form PINNs (with 50 data points): (a) Evolution of the test loss after training to a training loss of 2×10^{-5} and (b) evolution of the number of iterations required to reach this training loss.

sole criterion to evaluate the speed of the training process. To estimate this difference for the three methods we trained a PINN model with 100 data points, 100 penalization points and 100 test points for 500 000 iterations. The average time to complete these 500 000 iterations on ten trainings is 129 seconds for weak-form PINNs, 405 seconds for classic PINNs and 709 seconds for modal PINNs. Weak-form PINNs is much faster which is expected since only order two time derivatives are used with the weak formulation. On the other hand, modal PINNs and classic PINNs are slower since derivatives up to the fourth order in space are computed. Modal PINN is the slowest formulation because of the size of the output layer. Therefore, even if weak-form PINNs need more iterations to converge than modal PINNs, they are not significantly slower since each iteration is faster.

7.1.2 Use of data

To further compare weak, modal and classic PINN models, we compare how the three formulations assimilate data set and how it affects their convergence and accuracy. To do so, we trained ten PINNs models with numerical data generated for $U = 2$ m/s and for different data set sizes from 10 to 500 data points. We used the same number of penalization points for the PDE and stopped training when a 2×10^{-5} training loss is reached. As before, Figure

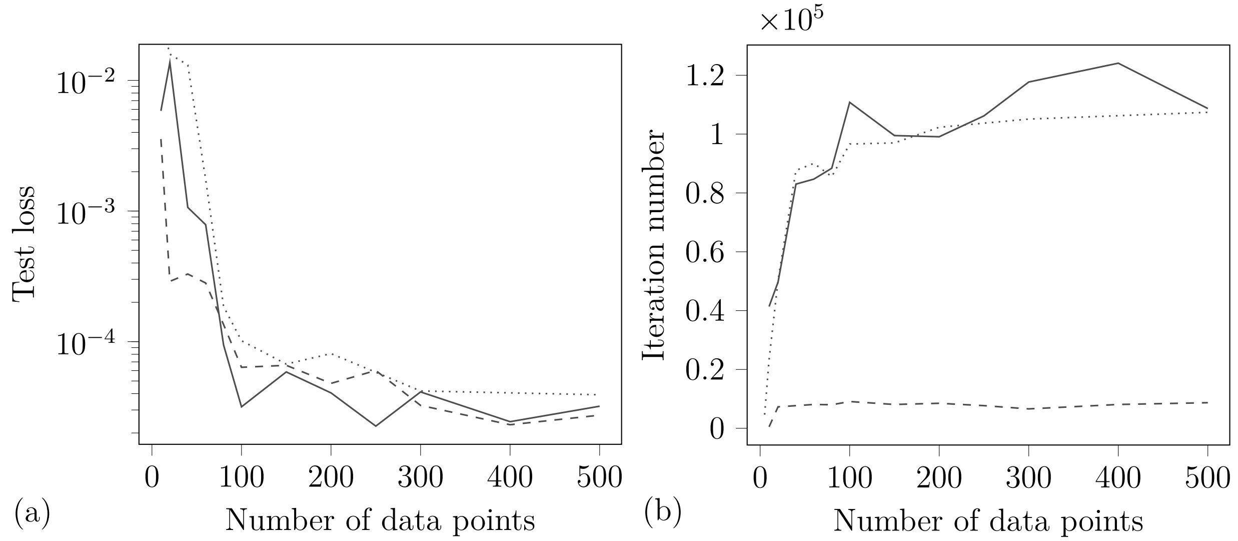


Figure 7.4 Influence of the data points for (- - -) modal PINNs, (——) classic PINNs and (······) weak-form PINNs: (a) Evolution of the test loss after training to a training loss of 2×10^{-5} and (b) evolution of the number of iterations required to reach this training loss.

7.4(a) shows the evolution of the test loss as a function of the number of data points and Figure 7.4(b) represents the number of iterations needed to converge.

In Figure 7.4(a), the test loss decreases steeply as the number of data points increases. Then, from around 100 points, the test loss decrease is much slower. Moreover, as for the residuals penalization, the data set size has a similar impact on the weak-form, modal and classic PINNs formulations. For the iterations needed to converge, Figure 7.4(b) demonstrates that the training is longer for larger data sets and that modal PINNs need less iterations to converge than classic and weak-form PINNs.

These results show that weak-form, modal PINNs and classic PINNs accuracy is influenced analogously by the data set size. However, the modal PINNs are again faster to converge as they need less iterations to converge even when the data set are largely increased. For around 100 data points, the PINNs manage to capture the pipe behaviour and that it is not effective to use more data as it increases the training time.

7.2 Extrapolation with PINNs

In this section, we demonstrate PINNs capacity to extrapolate outside the training data range. We show that PINNs can extrapolate either in space domain or in the time domain.

To do so, we trained two PINNs models with 200 data points on the whole time domain but only from the top of the pipe to 75% of its length as shown in Figure 7.5(a). One of the PINN is also trained with 200 penalization points of the PDE spread on the whole domain while the other does not use any penalization points. We plot the test loss along the length of the pipe for both cases. Figure 7.5(b) repeats the same method but with data points evenly spread on the whole length of the pipe and only between $t = 0$ and 75% of the experiment duration. We plot the test loss along the time of the experiment in this case.

Both Figures 7.5(a) and (b) show that the PINN trained without penalization of the PDE fails to capture the pipe behaviour in the area where no data were used for the training since the test loss dramatically increases after 75% of the axis. On the contrary, the PINN with penalization of the PDE keeps a low test loss value even where no data were used for the training. For the space extrapolation, the test loss increases slightly but to a reasonable value. Moreover, the increase reaches a plateau and does not diverge, which means that even if the PINN is less accurate, it still manages to follow the pipe movements.

This shows that the addition of the prior knowledge from the PDE helps to extrapolate outside of the training data range. This could be usefull in industrial applications when it is difficult to obtain data at certain positions and to extrapolate in time, meaning predict the state of the system in the near future.

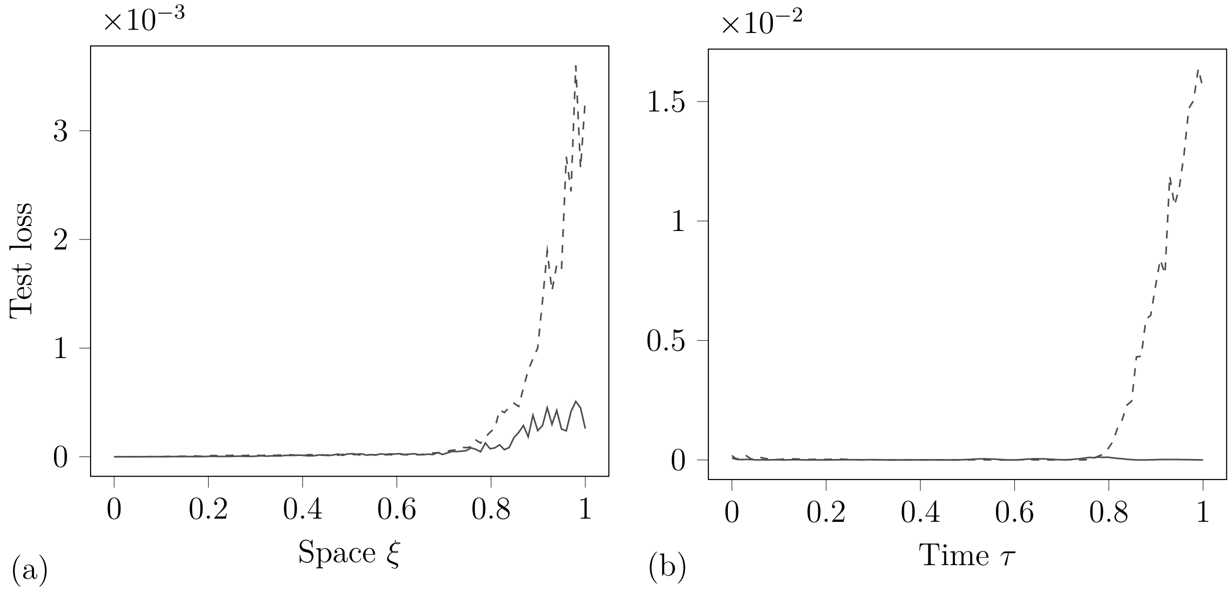


Figure 7.5 Extrapolation capacities of classic PINNs: (a) Test loss value in space for (- - -) a PINN trained with only data between 0 and 0.75L and (—) for a PINN trained with data between 0 and $0.75L$ and 200 points of residuals penalization. (b) Same principle with data only between 0 and $0.75T_{max}$.

7.3 Determination of hidden parameters with PINNs

In this section, we demonstrate the capability from PINNs to solve inverse problems by determining hidden parameters of the pipe conveying fluid from the data. We determine the value of the flowrate that is going through the pipe at the moment when the data were acquired. This is interesting since determining the flowrate in the pipe is equivalent to determining the water added damping in the pipe since this damping is depending on the flowrate. Determining the water added damping is important for hydraulic turbines applications as the water added damping influences the vibration frequencies of the turbines and therefore influences the fatigue evolution. However, water added damping is a quantity that is difficult to compute or to measure, determining it from classic sensor data with PINNs would be really usefull to predict and avoid turbines failures by fatigue.

7.3.1 Use of numerical data

Firstly, we work with 5000 numerically generated data and 500 penalization points where the PDE residuals are evaluated. The data points were generated for two flowrates values. We use classic PINNs for this application. Indeed, modal PINNs are more effective but needs to be given the eigenfrequencies of the modes of the pipe. However, these eigenfrequencies are hard to obtain experimentally for all the modes and there is no point in solving the PDE to get the eigenfrequencies to then use PINNs. Since we want to test the same method with numerical and experimental data we decided to use only classic PINNs in this case.

Figure 7.6 shows the result from this study for $U = 3$ m/s and $U = 4$ m/s. Figures 7.6(a) and (b) show the evolution of the training and test losses during the training of the PINN. There is no over-fitting as the test loss always decreases during the training of the PINN. Moreover, the test loss is very close to the training loss, which proves the effectiveness of PINNs.

The optimization of the flowrate value is shown in Figures 7.6(c) and (d) for both cases. The flowrate is initialized at $U = 1$ m/s and converged to 3.1 m/s and 3.9 m/s which corresponds to a relative error of 4.4% and 1.6%. The flowrate value stabilizes close to the actual value and is then almost completely constant meaning that it converged to the finale value.

Lastly, Figures 7.6(e) and (f) show the deflection of the tip of the pipe obtained from the linear model and from the PINNs output. The PINNs match the numerical data, which means that determining the flowrate does not reduce the PINNs capacity to simulate the pipe motion. Moreover, the oscillations are shorter for the plot with $U = 4$ m/s, which is coherent with the fact that the damping increases with the increase of the flowrate.

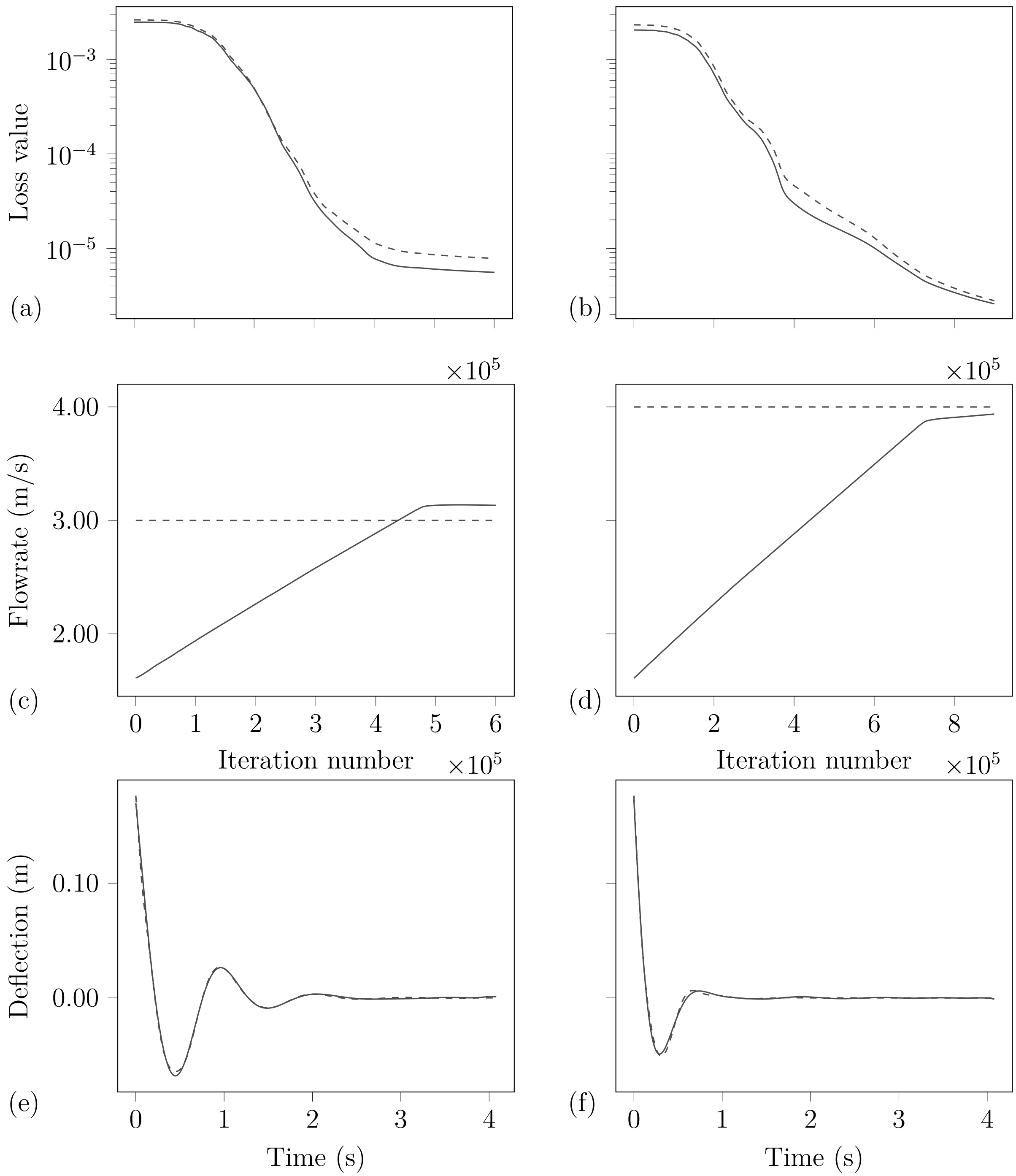


Figure 7.6 Classic PINN to determine the flowrate and simulate the pipe motion from numerical data. In (a) and (b): (—) and (- - -) respectively represents the training and the validation loss for $U = 3$ m/s and $U = 4$ m/s. In (c) and (d): (—) and (- - -) respectively represents the optimized flowrate from the PINN and the actual value for $U = 3$ m/s and $U = 4$ m/s. In (e) and (f): (—) and (- - -) respectively represents position of the tip simulated by the PINN and the numerical data.

7.3.2 Use of experimental data

In this last application of PINNs, we use actual experimental data from the setup for the first time. We use the same classic PINNs structure as for the previous example with 5000 data points and 500 points for the penalization of the PDE. The data points correspond to the movement of the pipe after a perturbation at a given flowrate below the critical speed. Figure 7.7 shows the result of the PINNs trained on two sets of experimental data.

In Figures 7.7(a) and (b) the curves represent the evolution of the training and test losses during the training of the PINNs with the experimental data obtained respectively with $U = 3.2$ m/s and $U = 4.6$ m/s. As with numerical data, the test loss always decreases during the training meaning that there is no over-fitting. However, both losses increase sharply by approximately two orders of magnitude. This is caused by a method used to achieve convergence. Indeed, the data are slightly noisy and some experimental phenomena are not described perfectly by the linear model. This caused the PINNs to get stuck into a local minima during training. The PINNs are converging to a trivial solution which minimize the loss part from the PDE since the PDE admits a trivial solution. This problem was handled in other cases by using a weight on the data part of the loss to avoid the trivial solution by giving more importance to the data and forcing the solution to fit the data before fitting the linear model. However, with experimental data this was not sufficient and we had to start the training without the PDE part of the loss. Basically, we trained the PINN like a regular neural network only on the data until a certain value of the training loss and then we added the penalization of the PDE. This allows the solution to steer away from the trivial solution by fitting the data first. The addition of the penalization of the PDE to the loss function accounts for the spike in the loss curves.

Figures 7.7(c) and 7.7(d) show the evolution of the flowrate during the training. As with the numerical data case the flowrates are initialized at $U = 1$ m/s. The flowrate value is stable and equal to 1 m/s for the first part of the training which corresponds to the part of the training without the PDE penalization. Indeed, without the PDE, the value of the flowrate does not intervene in the loss value and is not optimized. As for the case with numerical data, the flowrate stabilizes to a constant value close to the actual experimental flow speed. The values obtained are 3.4 m/s and 5.0 m/s which represent a relative error of 6.6% and 8.0%. These values are the flowrate values that minimize the PDE loss, however it also depends on other parameters such as the length of the pipe and the Young modulus of the pipe rubber. It is likely that these parameters are not exactly initialized at their actual values and that would affect the optimal value of the flowrate explaining the error. An interesting approach to solve this problem would be to define these other uncertain parameters as variables to be

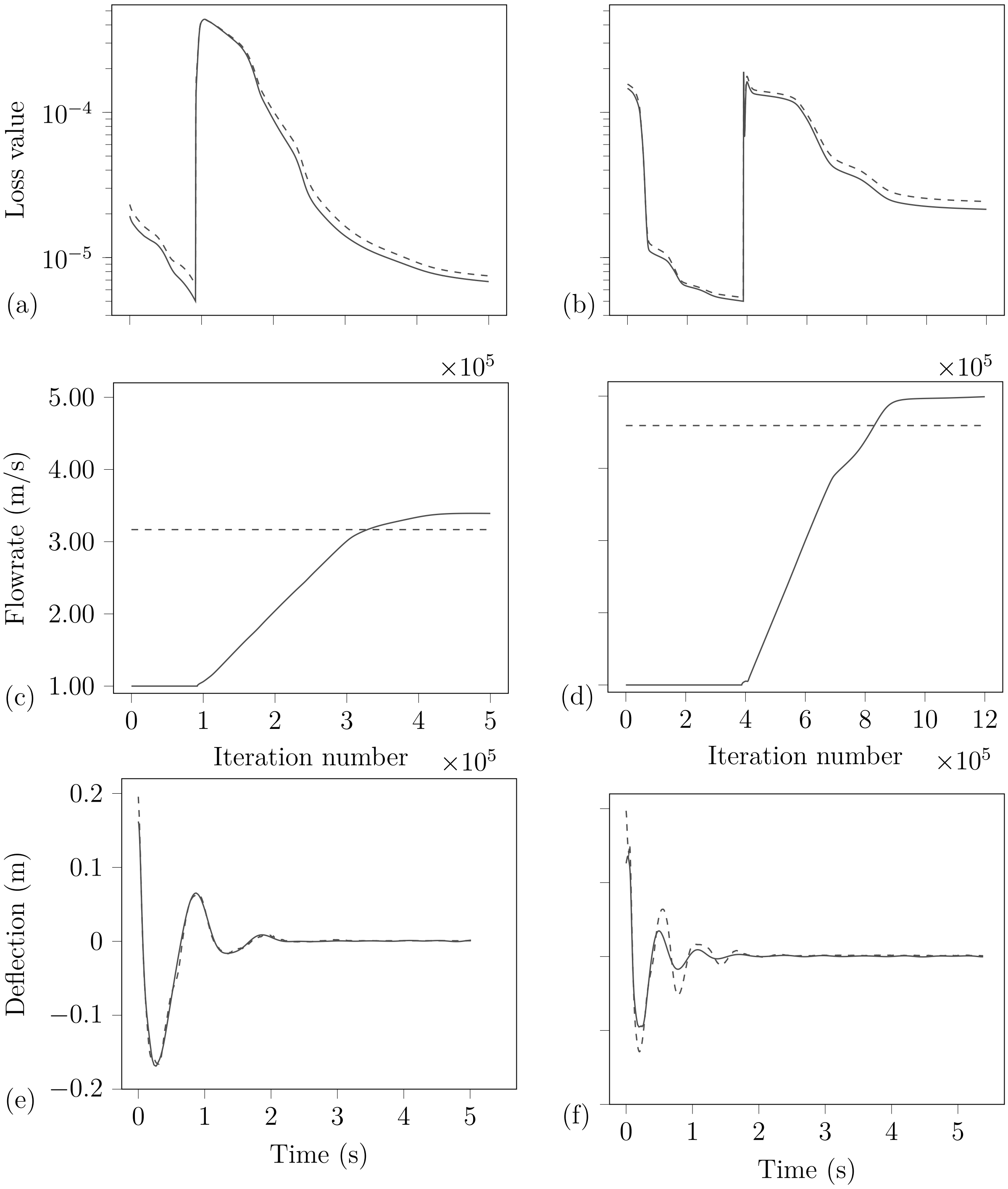


Figure 7.7 Classic PINN to determine the flowrate and simulate the pipe motion from experimental data. In (a) and (b): (—) and (- - -) respectively represents the training and the validation loss for $U = 3.1$ m/s and $U = 4.6$ m/s. In (c) and (d): (—) and (- - -) respectively represents the optimized flowrate from the PINN and the actual value for $U = 3.2$ m/s and $U = 4.6$ m/s. In (e) and (f): (—) and (- - -) respectively represents position of the tip simulated by the PINN and the experimental data.

optimized during the PINNs training.

Lastly, Figures 7.7(e) and 7.7(f) show a comparison of the position of the tip of the pipe from the experimental data and from the PINNs simulation. On the second figure the error between the PINNs and the experimental data is quite significant. This comes from phenomena that are not predicted by the PDE. To avoid this kind of problem it is important to use the most complete model. We also notice that the initial condition seems quite different which can be explained by the complexity to impose the proper initial condition to the pipe before releasing it experimentally.

Ultimately, we want to define all the parameters of the PDE with a variable part initialized at 0 to build a PINN model that can optimize the parameters of the system. For example, we can define the Young modulus of the pipe as $E = E_0(1 + \epsilon_E)$ where ϵ_E is variable and initialized at 0 and E_0 is initialized at the estimated value of the Young modulus. This method allows us to build a PINN model that works with uncertain parameters in the PDE and that can potentially re-optimize the value of the Young modulus to model the evolution of the pipe Young modulus caused by wear and tear.

CHAPTER 8 CONCLUSION

8.1 Summary of Works

In this thesis, we conducted technological research, designed and built a pipe conveying fluid experimental setup with its instrumentation and data acquisition system. The current experimental setup not only allows us to study the behaviour of the classic cantilevered pipe but can also be easily modified to implement other boundary conditions or pipe features. The measurement system is composed of two high speed cameras and one flowmeter that allows to reconstruct the three dimensional behaviour of the pipe for any flowrate.

We then conducted experimental campaigns on pipes of different lengths and inner diameters to observe interesting phenomena and compare them with the theoretical results. Namely, we observed the behaviour before the critical speed, measured the critical speed and compared these experimental results with the pipe linear model. We also measured the evolution of the amplitude and of the vibrations frequency as a function of the flowrate in the non-linear domain and observed interesting phenomena on the trace of the tip and the mode shapes. For instance, we observed the transition from planar movement to circular planar movement to chaotic movement. The measurements capabilities demonstrated here represent the physical half of a digital twin test bench.

Alongside this experimental work, we applied classic PINNs, modal PINNs and weak PINNs to model the pipe in the linear case. We demonstrate that these methods capture the pipe behaviour below the critical speed, work with numerical or experimental data from the setup, regularize the solution and help to extrapolate behind data. We also showed that these methods solve inverse problems by determining the value of the flowrate. Lastly, we showed the advantages from both classic and weak PINNs compared to the classic PINNs.

8.2 Limitations

The experimental setup was designed to be easily modified to allow the use of other boundary conditions and pipe feature. However, its main current limitation is the data acquisition and especially the cameras. Indeed, the current method fits the pipes position with a polynomial function and does not work if the tip of the pipe goes over the bottom point of the pipe since there would be two points for the same vertical coordinate. This situation has been observed at high flowrates. Another limitation is the high level of noise in the signal from the flowmeter which renders difficult any real time reading of the flowrate.

Another limitation concerns the pipe characterization. The determination of the pipe characteristics was achieved only by observing the free vibration of the pipe after a perturbation. This vibration was only composed of the first mode and was used to get the characteristics by using the linear model.

The last limitations concern the PINN methods that were used in this work. First, the PDE used to enforce the prior knowledge is the linear equation of the pipe, thus allowing the PINNs to work only for low flow velocity (before destabilization) and for a perturbed pipe. Moreover, the convergence of the PINNs was sometimes complex to obtain and was often stuck in local minima.

8.3 Future Research

This project is only the beginning of a large project aiming to develop digital twins. Therefore, there are many possible improvements and new possibilities to explore.

8.3.1 Experimental setup improvements

The experimental setup could be improved to display a larger variety of phenomena and to improve the quality of the experiments. We could implement the following new features:

- Simply supported or clamped boundary conditions at the tip of the pipe.
- Spring fixed on the pipe either at the tip of the pipe or in the middle.
- Modification of the mould and of the pipe to embed added masses on the pipe.

These new features would allow future students to experiment with other models and behaviour for the pipe digital twins.

It would also be interesting to add a plastic mesh at the bottom of the tank to limit the splashes from the water flowing out of the pipe.

Finally, the characterization of the pipe should be improved by observing higher modes vibrations and by using the pipe non linear models to access the values of the pipe parameters.

8.3.2 Data acquisition improvements

The acquisition of the data could be improved firstly by improving the image treatment method. As we explained before, the current method does not work if the tip of the pipe goes over the bottom point of the pipe which happens at high flowrates. Moreover, this method does not give any information on the Lagrangian coordinate s along the pipe which is used in most non-linear models. To solve this problem, an intern has developed a new method that uses numerically generated photos of the pipe to train a convoluted neural network (CNN) that analyzes actual images from the setup after its training. To do so, the images numerically generated show a pipe generated by using a polynomial function for each of the three x, y and z coordinates. These three polynomials are a function of the Largrangian coordinate s, thus it fits any shape of pipe and we have access to the position x, y and z of the pipe depending on the s position along the pipe.

It could also be of interest to add a thermometer to have more precise results on the mass flowrate since we use a volumic flowmeter. During long operation of the system, the temperature of the water in the circuit increases by more than 10 degrees because of the pump.

The current signal from the flowmeter is relatively noisy and it would be important to reduce the noise to have more accurate flowrate values and also to be able to measure the flowrate on shorter period of time.

Finally, an accelerometer could be added on the pipe fixation. It would provide interesting data to train the PINNs since it would be data that are more comparable to actual data retrieved in a industrial application. Moreover, it would allow us to demonstrate that the PINNs are capable to capture the pipe behaviour with data that are limited and not consisting of videos of the entire pipe motion.

8.3.3 PINNs future

The first improvement of the PINNs model would be to implement a three dimensional non-linear model as the PINNs PDE to have a PINN that works at a flowrate higher than the critical speed. It would be best to set the parameters of this PDE with uncertainty as variables so that the PINN optimizes them. It would also be important to model the pipe fixation as a ball fixation with springs of a given stiffness that can be optimized by the PINNs with the other PDE parameters.

Another step would be to implement dynamic learning rates and loss weights and also investigate other hyper parameters to improve the convergence.

Lastly, other PINNs formulations should be investigated, for example a weak-form formulation of the pipe model with domain decomposition. These methods are investigated in the literature and seem promising since they reduce the PDE orders and account for discontinuities. The neural networks themselves shall also be improved with adaptive activation functions or different architectures for example.

REFERENCES

[1] M. P. Païdoussis, *Fluid-Structure Interactions - Slender Structures and Axial Flow, Volume 1 (2nd Edition)*. Elsevier, 2014. [Online]. Available: https://app.knovel.com/hotlink/toc/id:kpFSISSAF1/fluid-structure-interactions/fluid-structure-interactions (Cited on Page xii, 15, 17, 20, 21, 23)

[2] M. H. Albadi and E. El-Saadany, "Overview of wind power intermittency impacts on power systems," *Electric power systems research*, vol. 80, no. 6, pp. 627–632, 2010. (Cited on Page 2)

[3] Z. Chen, "Issues of connecting wind farms into power systems," in *2005 IEEE/PES Transmission & Distribution Conference & Exposition: Asia and Pacific*. IEEE, 2005, pp. 1–6. (Cited on Page 2)

[4] C. Trivedi, B. Gandhi, and C. J. Michel, "Effect of transients on francis turbine runner life: a review," *Journal of Hydraulic Research*, vol. 51, no. 2, pp. 121–132, 2013. (Cited on Page 2)

[5] M. Gagnon, S. Tahan, P. Bocher, and D. Thibault, "Impact of startup scheme on francis runner life expectancy," in *IOP conference series: earth and environmental science*, vol. 12, no. 1. IOP Publishing, 2010, p. 012107. (Cited on Page 2)

[6] U. Dorji and R. Ghomashchi, "Hydro turbine failure mechanisms: An overview," *Engineering Failure Analysis*, vol. 44, pp. 136–147, 2014. (Cited on Page 2)

[7] A. Luna-Ramírez, A. Campos-Amezcua, O. Dorantes-Gómez, Z. Mazur-Czerwiec, and R. Muñoz-Quezada, "Failure analysis of runner blades in a francis hydraulic turbinecase study," *Engineering Failure Analysis*, vol. 59, pp. 314–325, 2016. (Cited on Page 2)

[8] X. Liu, Y. Luo, and Z. Wang, "A review on fatigue damage mechanism in hydro turbines," *Renewable and Sustainable Energy Reviews*, vol. 54, pp. 1–14, 2016. (Cited on Page 2)

[9] C. Seeley, S. Patil, A. Madden, S. Connell, G. Hauet, and L. Zori, "Hydro francis runner stability and forced response calculations," in *Turbo Expo: Power for Land, Sea, and Air*, vol. 58684. American Society of Mechanical Engineers, 2019, p. V07AT36A006. (Cited on Page 2)

[10] D. Li, X. Duan, N. Yue, R. Gong, H. Wang, D. Qin, and X. Wei, “Investigation on runner resonance and fatigue life of a high–head pump–turbine,” *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, p. 09544062211065371, 2022. (Cited on Page 2)

[11] L. He, Y. He, Y. Luo, and Z. Wang, “Investigation on fluid added mass effect in the modal response of a pump-turbine runner,” in *IOP Conference Series: Materials Science and Engineering*, vol. 52, no. 2. IOP Publishing, 2013, p. 022038. (Cited on Page 2)

[12] A. S. Dehkharqani, J.-O. Aidanpää, F. Engström, and M. J. Cervantes, “A review of available methods for the assessment of fluid added mass, damping, and stiffness with an emphasis on hydraulic turbines,” *Applied Mechanics Reviews*, vol. 70, no. 5, 2018. (Cited on Page 2)

[13] D. Valentín, A. Presas, C. Valero, M. Egusquiza, and E. Egusquiza, “Selection and optimization of sensors for monitoring of francis turbines,” in *IOP Conference Series: Earth and Environmental Science*, vol. 774, no. 1. IOP Publishing, 2021, p. 012028. (Cited on Page 2, 15)

[14] C. Valero, E. Egusquiza, A. Presas, D. Valentin, M. Egusquiza, and M. Bossio, “Condition monitoring of a prototype turbine. description of the system and main results,” in *Journal of Physics: Conference Series*, vol. 813, no. 1. IOP Publishing, 2017, p. 012041. (Cited on Page 2)

[15] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, “Physics-informed machine learning,” *Nature Reviews Physics*, vol. 3, no. 6, pp. 422–440, 2021. (Cited on Page 3, 7, 12)

[16] S. Cai, Z. Mao, Z. Wang, M. Yin, and G. E. Karniadakis, “Physics-informed neural networks (pinns) for fluid mechanics: A review,” *Acta Mechanica Sinica*, pp. 1–12, 2022. (Cited on Page 3, 7)

[17] C. Semler, “Pipe conveying fluid: a paradigm of nonlinear dynamics,” Ph.D. dissertation, McGill University, 1996. (Cited on Page 4, 29)

[18] C. Trivedi and M. J. Cervantes, “Fluid-structure interactions in francis turbines: A perspective review,” *Renewable and Sustainable Energy Reviews*, vol. 68, pp. 87–101, 2017. (Cited on Page 4)

[19] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. (Cited on Page 7)

[20] R. Collobert and J. Weston, “A unified architecture for natural language processing: Deep neural networks with multitask learning,” in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 160–167. (Cited on Page 7)

[21] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, “Deep reinforcement learning framework for autonomous driving,” *Electronic Imaging*, vol. 2017, no. 19, pp. 70–76, 2017. (Cited on Page 7)

[22] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations,” *Journal of Computational physics*, vol. 378, pp. 686–707, 2019. (Cited on Page 7, 55)

[23] X. Jin, S. Cai, H. Li, and G. E. Karniadakis, “Nsfnets (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations,” *Journal of Computational Physics*, vol. 426, p. 109951, 2021. (Cited on Page 7)

[24] S. Cai, Z. Wang, S. Wang, P. Perdikaris, and G. E. Karniadakis, “Physics-informed neural networks for heat transfer problems,” *Journal of Heat Transfer*, vol. 143, no. 6, 2021. (Cited on Page 7)

[25] S. Goswami, C. Anitescu, S. Chakraborty, and T. Rabczuk, “Transfer learning enhanced physics informed neural network for phase-field modeling of fracture,” *Theoretical and Applied Fracture Mechanics*, vol. 106, p. 102447, 2020. (Cited on Page 7)

[26] E. Kharazmi, D. Fan, Z. Wang, and M. S. Triantafyllou, “Inferring vortex induced vibrations of flexible cylinders using physics-informed neural networks,” *Journal of Fluids and Structures*, vol. 107, p. 103367, 2021. (Cited on Page 7)

[27] X.-D. Bai and W. Zhang, “Machine learning for vortex induced vibration in turbulent flow,” *Computers & Fluids*, vol. 235, p. 105266, 2022. (Cited on Page 7)

[28] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989. (Cited on Page 7)

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. (Cited on Page 8, 56)

[30] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on scientific computing*, vol. 16, no. 5, pp. 1190–1208, 1995. (Cited on Page 8)

[31] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: a survey," *Journal of Marchine Learning Research*, vol. 18, pp. 1–43, 2018. (Cited on Page 8)

[32] A. D. Jagtap, K. Kawaguchi, and G. E. Karniadakis, "Adaptive activation functions accelerate convergence in deep and physics-informed neural networks," *Journal of Computational Physics*, vol. 404, p. 109136, 2020. (Cited on Page 9)

[33] E. Kharazmi, Z. Zhang, and G. E. Karniadakis, "Variational physics-informed neural networks for solving partial differential equations," *arXiv preprint arXiv:1912.00873*, 2019. (Cited on Page 11)

[34] Y. Zang, G. Bao, X. Ye, and H. Zhou, "Weak adversarial networks for high-dimensional partial differential equations," *Journal of Computational Physics*, vol. 411, p. 109409, 2020. (Cited on Page 11)

[35] E. Kharazmi, Z. Zhang, and G. E. Karniadakis, "hp-vpinns: Variational physics-informed neural networks with domain decomposition," *Computer Methods in Applied Mechanics and Engineering*, vol. 374, p. 113547, 2021. (Cited on Page 11)

[36] S. Wang, Y. Teng, and P. Perdikaris, "Understanding and mitigating gradient flow pathologies in physics-informed neural networks," *SIAM Journal on Scientific Computing*, vol. 43, no. 5, pp. A3055–A3081, 2021. (Cited on Page 11)

[37] W. Peng, W. Zhou, J. Zhang, and W. Yao, "Accelerating physics-informed neural network training with prior dictionaries," *arXiv preprint arXiv:2004.08151*, 2020. (Cited on Page 12, 57)

[38] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995. (Cited on Page 12)

[39] G. Raynaud, S. Houde, and F. P. Gosselin, "Modalpinn: an extension of physics-informed neural networks with enforced truncated fourier decomposition for periodic

flow reconstruction using a limited number of imperfect sensors," *Journal of Computational Physics*, p. 111271, 2022. (Cited on Page 12, 61)

[40] M. Paidoussis, "Dynamics of tubular cantilevers conveying fluid," *Journal of Mechanical Engineering Science*, vol. 12, no. 2, pp. 85–103, 1970. (Cited on Page 13)

[41] R. W. Gregory and M. P. Paidoussis, "Unstable oscillation of tubular cantilevers conveying fluid i. theory," *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, vol. 293, pp. 512 – 527, 1966. (Cited on Page 13)

[42] R. Gregory and M. Paidoussis, "Unstable oscillation of tubular cantilevers conveying fluid ii. experiments," *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, vol. 293, no. 1435, pp. 528–542, 1966. (Cited on Page 13)

[43] O. Doare´ and E. de Langre, "Effect of length on the instability of hanging pipes," in *ASME International Mechanical Engineering Congress and Exposition*, vol. 36592, 2002, pp. 1223–1232. (Cited on Page 13, 48)

[44] M. H. Ghayesh, M. P. Païdoussis, and M. Amabili, "Nonlinear dynamics of cantilevered extensible pipes conveying fluid," *Journal of Sound and Vibration*, vol. 332, no. 24, pp. 6405–6418, 2013. (Cited on Page 13)

[45] M. Païdoussis and C. Semler, "Nonlinear and chaotic oscillations of a constrained cantilevered pipe conveying fluid: a full nonlinear analysis," *Nonlinear Dynamics*, vol. 4, no. 6, pp. 655–670, 1993. (Cited on Page 13)

[46] M. Païdoussis and F. Moon, "Nonlinear and chaotic fluidelastic vibrations of a flexible pipe conveying fluid," *Journal of Fluids and Structures*, vol. 2, no. 6, pp. 567–591, 1988. (Cited on Page 13)

[47] M. Wadham-Gagnon, M. Païdoussis, C. Semler *et al.*, "Dynamics of cantilevered pipes conveying fluid. part 1: Nonlinear equations of three-dimensional motion," *Journal of fluids and structures*, vol. 23, no. 4, pp. 545–567, 2007. (Cited on Page 13)

[48] Y. Modarres-Sadeghi, M. P. Paı, C. Semler *et al.*, "Three-dimensional oscillations of a cantilever pipe conveying fluid," *International Journal of Non-Linear Mechanics*, vol. 43, no. 1, pp. 18–25, 2008. (Cited on Page 13)

[49] O. Doaré and E. de Langre, "The flow-induced instability of long hanging pipes," *European Journal of Mechanics-A/Solids*, vol. 21, no. 5, pp. 857–867, 2002. (Cited on Page 14)

[50] M. Paidoussis and C. Semler, "Non-linear dynamics of a fluid-conveying cantilevered pipe with a small mass attached at the free end," *International Journal of Non-Linear Mechanics*, vol. 33, no. 1, pp. 15–32, 1998. (Cited on Page 14, 26)

[51] S. Rinaldi and M. Païdoussis, "Dynamics of a cantilevered pipe discharging fluid, fitted with a stabilizing end-piece," *Journal of Fluids and Structures*, vol. 26, no. 3, pp. 517–525, 2010. (Cited on Page 14, 36)

[52] M. H. Ghayesh and M. P. Pai¨ doussis, "Dynamics of a fluid-conveying cantilevered pipe with intermediate spring support," in *Fluids Engineering Division Summer Meeting*, vol. 54518, 2010, pp. 893–902. (Cited on Page 14, 36)

[53] Y. Sugiyama, Y. Tanaka, T. Kishi, and H. Kawagoe, "Effect of a spring support on the stability of pipes conveying fluid," *Journal of Sound and Vibration*, vol. 100, no. 2, pp. 257–270, 1985. (Cited on Page 14)

[54] S. I. Lee and J. Chung, "New non-linear modelling for vibration analysis of a straight pipe conveying fluid," *Journal of sound and vibration*, vol. 254, no. 2, pp. 313–325, 2002. (Cited on Page 14)

[55] G. Kuiper and A. Metrikine, "On stability of a clamped-pinned pipe conveying fluid," *Heron, 49,(3)*, 2004. (Cited on Page 14)

[56] C. An and J. Su, "Dynamic behavior of pipes conveying gas–liquid two-phase flow," *Nuclear Engineering and Design*, vol. 292, pp. 204–212, 2015. (Cited on Page 14)

[57] M. Paidoussis and C. Sundararajan, "Parametric and combination resonances of a pipe conveying pulsating fluid," *Journal of Applied Mechanics*, 1975. (Cited on Page 14)

[58] T. Jiang, H. Dai, and L. Wang, "Three-dimensional dynamics of fluid-conveying pipe simultaneously subjected to external axial flow," *Ocean Engineering*, vol. 217, p. 107970, 2020. (Cited on Page 14)

[59] M. Pastor, M. Binda, and T. Harčarik, "Modal assurance criterion," *Procedia Engineering*, vol. 48, pp. 543–548, 2012. (Cited on Page 20)

[60] M. Demenois. (2022) Python codes and data used in this project. [Online]. Available: https://github.com/MorganDemenois/Digital-twin-project (Cited on Page 20)

[61] M. Paidoussis and P. Des Trois Maisons, "Free vibration of a heavy, damped, vertical cantilever," *Journal of Applied Mechanics*, 1971. (Cited on Page 32)

[62] K. Yamashita, H. Furuya, H. Yabuno, and M. Yoshizawa, “Nonplanar vibration of a vertical fluid-conveying pipe (effect of horizontal excitation at the upper end),” *Journal of Vibration and Acoustics*, vol. 136, no. 4, 2014. (Cited on Page 36, 91)

[63] J. Rousselet and G. Herrmann, “Dynamic behavior of continuous cantilevered pipes conveying fluid near critical velocities,” *Journal of Applied Mechanics*, 1981. (Cited on Page 45)

[64] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, “{TensorFlow}: A system for {Large-Scale} machine learning,” in *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, 2016, pp. 265–283. (Cited on Page 55)

[65] G. Raynaud, “Study of physics-informed neural networks to solve fluid-structure problems for turbine-like phenomena,” Ph.D. dissertation, Polytechnique Montréal, 2021. (Cited on Page 55)

[66] J. Cusumano and F. Moon, “Chaotic non-planar vibrations of the thin elastica: Part i: Experimental observation of planar instability,” *Journal of Sound and Vibration*, vol. 179, no. 2, pp. 185–208, 1995. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022460X85700132 (Cited on Page 91)

APPENDIX A USE OF THE SHAKER TO CHARACTERIZE THE PIPE

In this appendix we present the work conducted to characterize the pipe higher modes of vibration with a shaker.

We use a shaker to determine the PDE parameters for higher modes of the pipe. Indeed, the free motion of the pipe as shown in the previous method mostly displays first mode behaviour. With the shaker, we determine the parameters valid for higher modes, verify if they differ and if we need for example to differentiate the PDE for the different modes of the pipe.

We first built a setup to submit the pipe to the vibration of the shaker. The shaker controls the acceleration of the vibration as well as the frequency and the amplitude. One high speed camera, orthogonal to the shaker axis, measures the amplitude of the tip of the pipe.

The shaker setup is shown in Figure A.1. The setup is basically an aluminum structure with a trolley on two rails. The pipe is fixed on the trolley the same way it is fixed in the actual setup. The trolley is attached to the shaker rod that induces the vibration.

The idea is to use a deep neural network to determine the parameters of the pipe from the experimental values. We first determine the form of the pipe PDE without the water flowing in the pipe (i.e. $u = \beta = 0$) and obtain the following PDE:

$$\alpha\dot{\eta}^{''''} + \eta^{''''} - \gamma(1-\xi)\eta^{''} + \gamma\eta^{'} + \ddot{\eta} = 0. \tag{A.1}$$

The displacement η of the pipe is composed of the displacements of the pipe itself and of the shaker as follows:

$$\eta(\xi,\tau) = \eta_0(\xi,\tau) + \eta_s exp(i\omega_s\tau), \tag{A.2}$$

where η_s is the amplitude of the shaker vibration and ω_s is the vibration frequency of the shaker.

We rewrite the matrix Equation (2.36) from literature review with the second member added by the shaker by inserting the Equation (A.2) into Equation (A.1) and applying the same manipulations shown in the literature review:

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{S}\dot{\boldsymbol{q}} + \boldsymbol{K}\boldsymbol{q} = \boldsymbol{E}\eta_s\omega_s^2 exp(i\omega_s\tau), \tag{A.3}$$

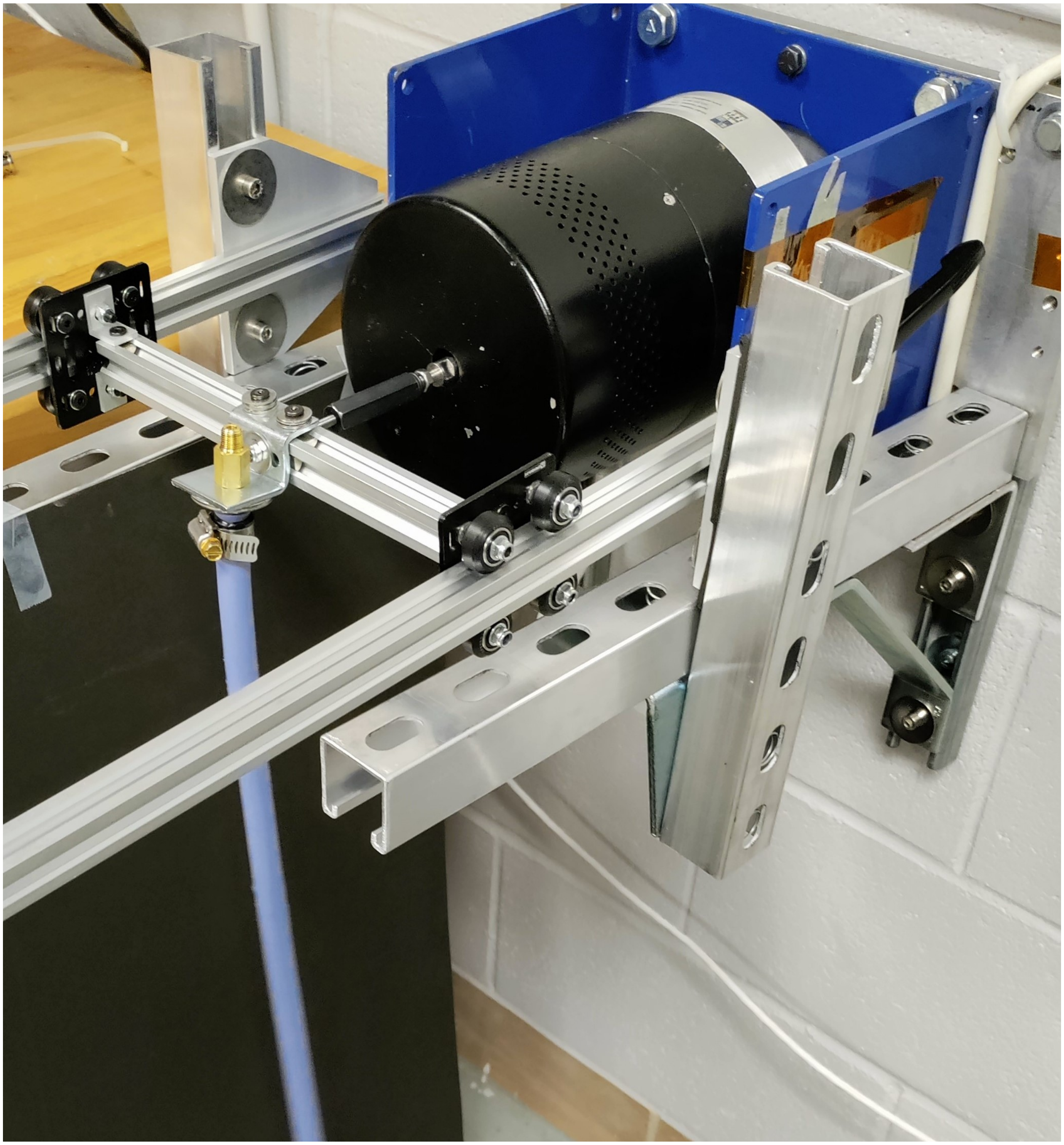


Figure A.1 Photo of shaker experimental setup for the pipe characterization.

with

$$\boldsymbol{E} = \begin{pmatrix} \int_0^L \phi_1(\xi) d\xi \\ \int_0^L \phi_2(\xi) d\xi \\ . \\ . \\ . \\ \int_0^L \phi_N(\xi) d\xi \end{pmatrix} \tag{A.4}$$

We then suppose the solution with the following form:

$$\boldsymbol{q} = \boldsymbol{q}_p(\tau) + \boldsymbol{q}_h(\tau), \tag{A.5}$$

where $\boldsymbol{q}_h$ is the solution of the homogeneous system without the second member. It is therefore constant and equal to zero as it is a solution of the system without excitation and without water flowing in it.

Then we write the particular solution as:

$$\boldsymbol{q}_p(\tau) = \boldsymbol{Q} exp(i(\omega_s \tau + \theta)). \tag{A.6}$$

If we apply this solution in the Equation (A.3), we obtain the following:

$$\boldsymbol{Q} exp(i\theta) = \eta_s \omega_s^2 (-\omega_s^2 \boldsymbol{M} + i\omega_s \boldsymbol{S} + \boldsymbol{K})^{-1} \boldsymbol{E}. \tag{A.7}$$

After determining $\boldsymbol{Q}$ and θ, we compute the amplitude at the tip of the pipe given the frequency of the shaker, the non-dimensional parameters and the amplitude of the shaker.

We generate data of amplitude at the tip of the pipe for a large range of parameters and frequency. We then train a neural network with these data of amplitude at the tip and frequency of the shaker as inputs and the non-dimensional parameters as outputs. Therefore, after we measure the amplitude at the tip of the actual pipe for a certain shaker frequency, we are able to determine the parameters using the neural network.

The shape of the NN is shown in Figure A.2. The inputs of the NN are N values of the amplitude of the pipe from N different values of the shaker frequency. The outputs are the damping parameter and the gravity parameter that corresponds to E and E^*.

The pipe No.1 from Table 4.2 was tested in the shaker setup. The shaker was set to keep a constant displacement amplitude 0.01 m and to shake at a frequency from 3 to 20 Hz with a 0.05 Hz/s increment. Figure A.3(a) presents the evolution of the amplitude of the tip of

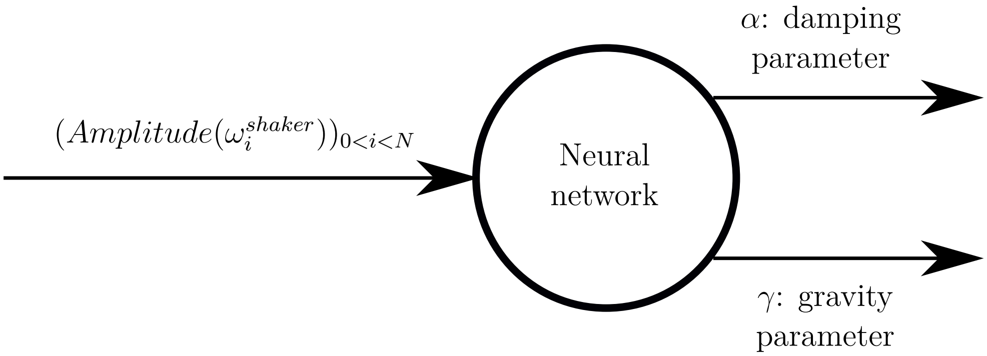

α: damping parameter
$(Amplitude(\omega_i^{shaker}))_{0<i<N}$
Neural network
γ: gravity parameter

Figure A.2 Shape of the NN used to determine the parameters of the pipe from the amplitude of the tip of the pipe and the frequency of the shaker.

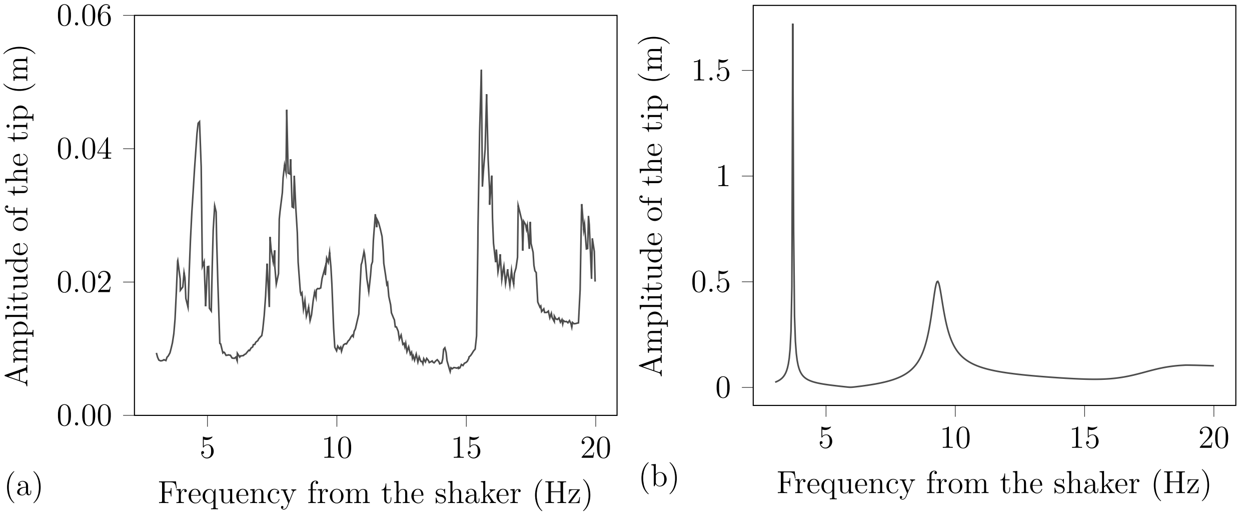


Figure A.3 Amplitude of the tip after excitation: (a) Amplitude of the tip of the pipe number 1 from Table 4.2 as a function of the shaker frequency from the videos and the experimental setup and (b) from the linear model on the right.

the pipe along the shaker frequency. Figure A.3(b) shows the evolution of the amplitude computed with the linear model, the same shaker parameters and the parameters from pipe No.1 obtained with the free vibration method. The two methods give dramatically different results. Figure A.3(a) displays regular spikes of similar amplitude while Figure A.3(b) has only two spikes corresponding to the first and second modes of the pipe where the first mode amplitude is three time larger. This difference is caused by two factors:

- The linear model fails to represent correctly the behaviour of the pipe subjected to this kind of solicitations. The amplitude at the eigenfrequencies is aberrantly high. This comes from the fact that, at higher amplitude around the eigenfrequencies, the damping phenomena are mainly non-linear and are not accounted for in the linear model.

- In the experimental shaker setup, different disruptive non-linear phenomena are observed at certain frequency and are not considered by the theoretical model. Three dimensional movements appear at certain frequencies despite the shaker solicitation being planar [66]. At some points these three dimensions movements become circular [62]. Moreover, at the highest frequency the structure with the trolley was vibrating, inducing a parasite oscillation of the pipe.

Due to the limitation of the shaker at these low frequencies, the difficulties to build a stiff enough structure and the limitation of the linear model, we decided to drop this technique in our current study.

APPENDIX B ACTIVATION FUNCTION CHOICE

In this section, we aim to select the best activation function for the neural networks of the PINNs. Firstly, we tested the ReLU activation function, the sinus function and the tanh function. It appears that the ReLU function was not effective at the first tests. We then compare thoroughly the tanh function and the sinus function for both classic and modal PINNs as shown in Figure B.1.

We generated numerical data from the linear model with $N = 4$. For the training, we used 100 data points and 100 points for the penalization of the PDE. We trained the PINN during 500 000 iterations with a constant learning rate $L_r = 10^{-5}$. The entire training set (100 data points and 100 penalization points) was used during the training. A test set was also constructed with 100 data points to compute the loss function on this set at each 100^{th} iteration and verify that the PINN is not over-fitting. The neural networks are composed of 3 layers of 20 neurons as defined before. We trained 10 PINNs for the 6 cases (classic PINNs with tanh, modal PINNs with tanh, weak PINNs with tanh, classic PINNs with sin, modal PINNs with sin and weak PINNs with sin) and used the average values of these 10 training.

Figure B.1 follows the evolution of the average of the training loss and of the test loss during the training for both activation functions and for classic, modal and weak PINNs.

Figure B.1(a) shows the loss for the classic PINNs. The test loss follows the training loss and does not increase with the epochs which means that there is no over-fitting. As expected the test loss is larger than the training loss. The tanh activation function is more effective for a low iteration number as both the training and test losses decrease faster at the beginning. However, at higher iterations, the training and test losses from the sinus activation function becomes smaller as the tanh PINN reaches a plateau.

Figure B.1(b), on the other hand, represents the loss for the modal PINNs. As with the classics PINNs, the test loss keeps decreasing which means that the PINN does not over fit. The tanh activation function is also more effective at the beginning and converge faster. However, the sinus activation function PINN does not catch up its delay on the contrary to the classics PINNs.

Finally, Figure B.1(c), shows the loss for the weak PINNs. Here again, the test loss always decreases which means that the model is not over-fitting on the data. Both activation functions have a very similar effectiveness for this formulation and the loss reaches the same values for the training and the testing part. The loss first reaches a plateau until approximately 150 000

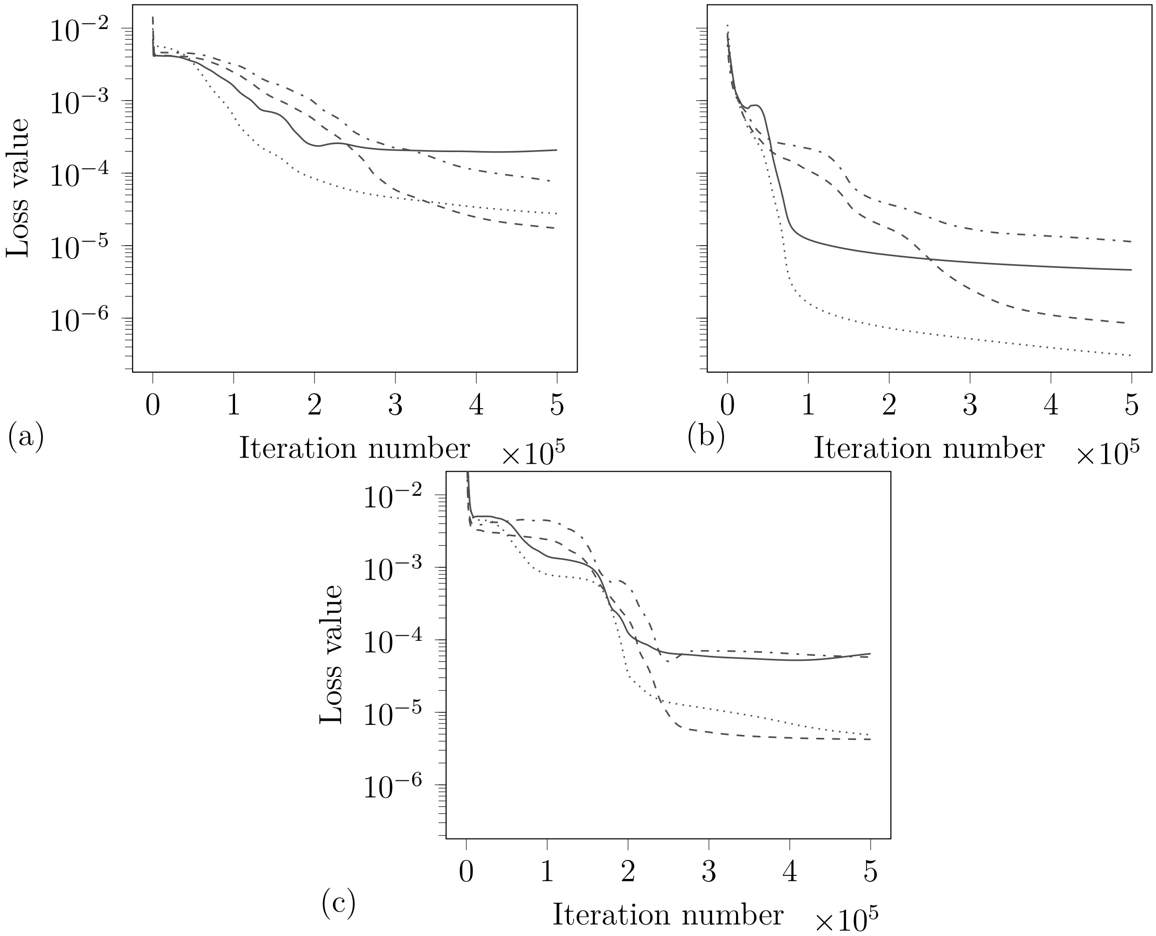


Figure B.1 Comparison of activation functions: evolution of the loss values during the training for (a) classic PINN, (b) modal PINN and (c) weak PINN. (- - -) Training loss evolution with the sinus activation function, (······) training loss evolution with the tanh activation function, (· - · - ·) test loss evolution with the sinus activation function and (——) test loss evolution with the tanh activation function.

iterations before decreasing again, reflecting the difficult convergence of this formulation.

When we compare Figure B.1(a) and (b), we see that for both activation functions, the modal PINNs are more efficient than classic PINNs since both the training and test loss decrease faster and reach a smaller minimum. On the other hand, the weak PINN formulation is more effective that classic PINNs but less effective than modal PINNs. However, this formulation might actually be the best compromise as it is more effective than classic PINNs and does not require the use of the problem eigenvalues.

In the rest of this study we use the tanh activation function. Indeed, the sinus function only presents a slight advantage for the classics PINNs and since we compare the three methods, we keep the same activation functions to keep the comparison relevant.

APPENDIX C NN SIZE CHOICE

To chose the size of the neural networks, we had to find the smallest NN that approximates correctly the function of interest which is the function describing the pipe movement. For that purpose, we trained the NN with only the data points and without the PDE residuals and checked what training loss value could be reached. We then reduced the size of the NN until the minimal training loss started increasing which means that the NN was getting too small to approximate the function correctly. We used 500 data points to train the NNs in all the cases below.

We first performed this size study on the classic PINNs formulation in Figure C.1. We trained 6 different NNs with 2 layers and 6 different numbers of neurons per layer and ploted the training loss evolution in 1.5×10^6 iterations in Figure C.1(b). The training loss reaches a plateau around 600 000 iterations meaning that the NN has reached its best approximation of the function. The plateau is reached around the same training loss values for NNs larger than 20 neurons meaning that this size is sufficient. With smaller NNs, the minimum training loss is slightly larger. We then repeated this operation with 5 different NNs which are all 20 neurons wide but have 5 different depths. Figure C.1(a) represents the training loss for these 5 NNs and shows a plateau as before. The optimal depth seems to be 3 layers as the minimum loss does not decrease anymore with a larger NN. The optimal size of the NN for the classic PINNs is therefore a 3 layers NN made of 20 neurons per layers.

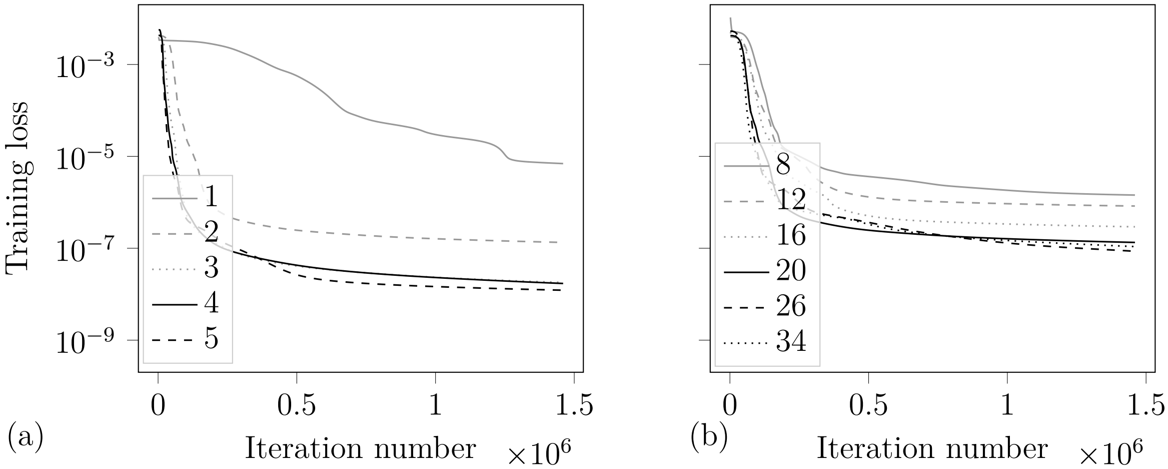


Figure C.1 Training loss evolution with classic PINNs for (a) a 20 neurons wide NN with 5 different number of layers and (b) a 2 layers deep NN for 6 different widths.

Figure C.2 shows the results of the same study with the modal PINNs formulation. We firstly trained 6 NNs composed of 2 layers and of 6 different widths in Figure C.2(b). We observed the same plateau as for classic PINNs but this plateau is attained at the same moment for the 6 different widths and is approximately 100 times smaller. We deduce that the modal PINNs formulation is way more effective than the classic PINNs formulation since the minimum loss is much smaller even with a small NN. A width of only 4 neurons is enough to approximate the pipe governing equation. Figure C.2(a) shows the results from the depth analysis with 5 different depths and 6 neurons per layers. Here again, the loss reached a plateau at the same level by the 5 different depths. We conclude that, with the modal PINN formulation a NN as small as 1 layer with 6 neurons could be enough to approximate the function of interest. However, since we want to compare modal PINNs and classics PINNs reliably we chose the same size of NNs as for classic PINNs, hence 3 layers deep and 20 layers wide NNs. This choice slightly increases the computation time but should no affect the overall results.

Finally, we conducted this study with the weak architecture as shown in Figure C.3. We trained 6 NNs composed of 2 layers and with 6 different widths. Figure C.3(b) shows the training loss evolution over 500 000 iterations for these 6 NNs. The minimum value of the plateau is reached for a 20 neurons wide NN which is the optimal size for this formulation. We then conducted the depth analysis with 5 different NNs composed of 20 neurons per layer and of 5 different lengths. The result are shown in Figure C.3(a) and the minimum of the plateau is reached for 3 layers deep NNs. The optimal size for this formulation is a 3 layers and 20 neurons per layer size. The minimum of the training loss has a similar value to the one from classic PINNs and is reached for the same NN size which means that the

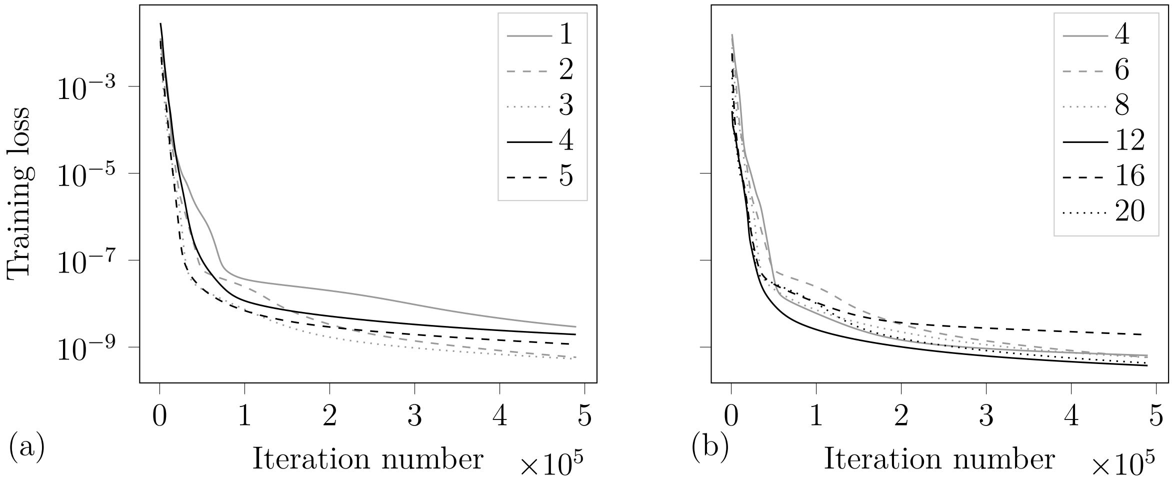


Figure C.2 Training loss evolution with modal PINNs for (a) a 6 neurons wide NN with 5 different number of layers and (b) a 2 layers deep NN for 6 different widths.

beam mode shape decomposition is as effective as the classic formulation to approximate the function of interest. However, this formulation is less effective than the modal decomposition to approximate the pipe movement. However, the goal here is to use PINNs and not just classic NN and therefore the weak form might still have the upper hand when using the PDE.

All the NNs in the following studies are 3 layers deep and 20 neurons wide.

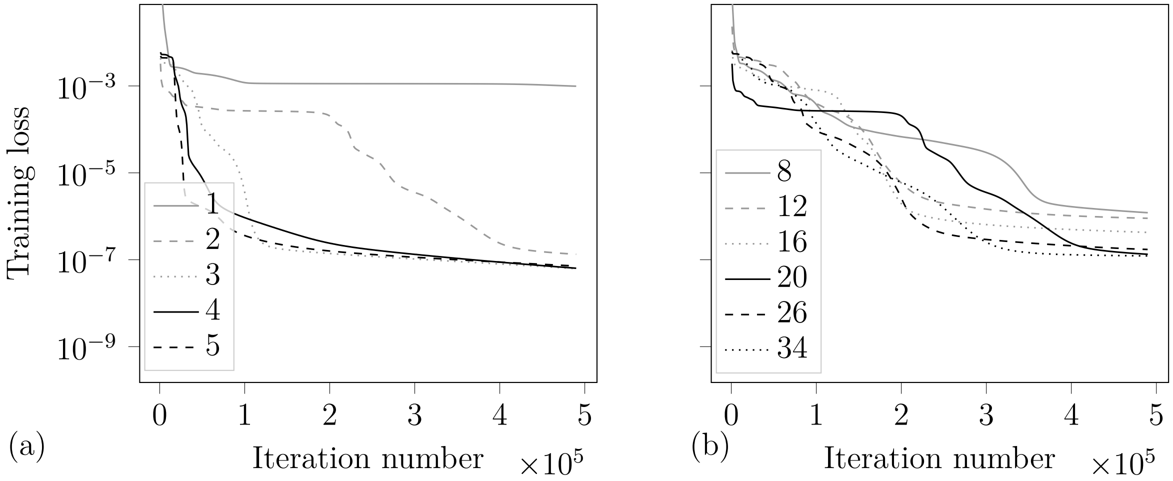


Figure C.3 Training loss evolution with weak PINNs for (a) a 20 neurons wide NN with 5 different number of layers and (b) a 2 layers deep NN for 6 different number of neurons per layer.

APPENDIX D PARAMETERS USED IN THE DIFFERENT PINNS MODELS

In this appendix we give the characteristics of the different PINNs models that are used in this thesis.

Table D.1 PINNs models used to find the activation function in Figure B.1.

Characteristics	1	2	3	4	5	6
Input Dimension	2	1	1	2	1	1
Output Dimension	1	8	4	1	8	4
NN Depth	3	3	3	3	3	3
NN Width	20	20	20	20	20	20
Training data points	100	100	100	100	100	100
Test data points	100	100	100	100	100	100
Penalization points	100	100	100	100	100	100
Penalization weight	0.01	0.01	10^{-7}	0.01	0.01	10^{-7}
Activation function	Tanh	Tanh	Tanh	Sin	Sin	Sin
Type	Classic	Modal	Weak	Classic	Modal	Weak

Table D.2 Classic PINNs models used to find the optimal classic PINNs NNs size in Figure C.1.

Characteristics	1	2	3	4	5	6	7	8	9	10
Input Dimension	2	2	2	2	2	2	2	2	2	2
Output Dimension	1	1	1	1	1	1	1	1	1	1
NN Depth	1	2	2	2	2	2	2	3	4	5
NN Width	20	8	12	16	20	26	34	20	20	20
Training data points	500	500	500	500	500	500	500	500	500	500
Test data points	0	0	0	0	0	0	0	0	0	0
Penalization points	0	0	0	0	0	0	0	0	0	0
Penalization weight	0	0	0	0	0	0	0	0	0	0
Activation	Th	Th	Th	Th	Th	Th	Th	Th	Th	Th

Table D.3 Modal PINNs models used to find the optimal modal PINNs NNs size in Figure C.1.

Characteristics	1	2	3	4	5	6	7	8	9	10
Input Dimension	2	2	2	2	2	2	2	2	2	2
Output Dimension	1	1	1	1	1	1	1	1	1	1
NN Depth	1	2	2	2	2	2	2	3	4	5
NN Width	20	8	12	16	20	26	34	20	20	20
Training data points	500	500	500	500	500	500	500	500	500	500
Test data points	0	0	0	0	0	0	0	0	0	0
Penalization points	0	0	0	0	0	0	0	0	0	0
Penalization weight	0	0	0	0	0	0	0	0	0	0
Activation	Th	Th	Th	Th	Th	Th	Th	Th	Th	Th

Table D.4 Weak PINNs models used to find the optimal weak PINNs NNs size in Figure C.2.

Characteristics	1	2	3	4	5	6	7	8	9	10
Input Dimension	1	1	1	1	1	1	1	1	1	1
Output Dimension	4	4	4	4	4	4	4	4	4	4
NN Depth	1	2	2	2	2	2	2	3	4	5
NN Width	20	8	12	16	20	26	34	20	20	20
Training data points	500	500	500	500	500	500	500	500	500	500
Test data points	0	0	0	0	0	0	0	0	0	0
Penalization points	0	0	0	0	0	0	0	0	0	0
Penalization weight	0	0	0	0	0	0	0	0	0	0
Activation	Th	Th	Th	Th	Th	Th	Th	Th	Th	Th

Table D.5 Classic PINNs models used to study the impact of the residuals in Figure 7.3.

Characteristics	1	2	3	4	5	6	7	8
Input Dimension	2	2	2	2	2	2	2	2
Output Dimension	1	1	1	1	1	1	1	1
NN Depth	3	3	3	3	3	3	3	3
NN Width	20	20	20	20	20	20	20	20
Training data points	50	50	50	50	50	50	50	50
Test data points	1000	1000	1000	1000	1000	1000	1000	1000
Penalization points	1	5	10	20	40	80	150	300
Penalization weight	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
Activation	Th	Th	Th	Th	Th	Th	Th	Th

Table D.6 Modal PINNs models used to study the impact of the residuals in Figure 7.3.

Characteristics	1	2	3	4	5	6	7	8
Input Dimension	1	1	1	1	1	1	1	1
Output Dimension	8	8	8	8	8	8	8	8
NN Depth	3	3	3	3	3	3	3	3
NN Width	20	20	20	20	20	20	20	20
Training data points	50	50	50	50	50	50	50	50
Test data points	1000	1000	1000	1000	1000	1000	1000	1000
Penalization points	1	5	10	20	40	80	150	300
Penalization weight	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
Activation	Th	Th	Th	Th	Th	Th	Th	Th

Table D.7 Weak PINNs models used to study the impact of the residuals in Figure 7.3.

Characteristics	1	2	3	4	5	6	7	8
Input Dimension	1	1	1	1	1	1	1	1
Output Dimension	4	4	4	4	4	4	4	4
NN Depth	3	3	3	3	3	3	3	3
NN Width	20	20	20	20	20	20	20	20
Training data points	50	50	50	50	50	50	50	50
Test data points	1000	1000	1000	1000	1000	1000	1000	1000
Penalization points	0	5	10	20	40	80	150	300
Penalization weight	10^{-7}	10^{-7}	10^{-7}	10^{-7}	10^{-7}	10^{-7}	10^{-7}	10^{-7}
Activation	Th	Th	Th	Th	Th	Th	Th	Th

Table D.8 Classic PINNs models used to study the impact of the data in Figure 7.4.

Characteristics	1	2	3	4	5	6	7	8	9	10
Input Dimension	2	2	2	2	2	2	2	2	2	2
Output Dimension	1	1	1	1	1	1	1	1	1	1
NN Depth	3	3	3	3	3	3	3	3	3	3
NN Width	20	20	20	20	20	20	20	20	20	20
Training data points	10	20	40	60	80	100	150	200	300	400
Test data points	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
Penalization points	1	1	1	1	1	1	1	1	1	1
Penalization weight	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
Activation	Th	Th	Th	Th	Th	Th	Th	Th	Th	Th

Table D.9 Modal PINNs models used to study the impact of the data in Figure 7.4.

Characteristics	1	2	3	4	5	6	7	8	9	10
Input Dimension	1	1	1	1	1	1	1	1	1	1
Output Dimension	8	8	8	8	8	8	8	8	8	8
NN Depth	3	3	3	3	3	3	3	3	3	3
NN Width	20	20	20	20	20	20	20	20	20	20
Training data points	10	20	40	60	80	100	150	200	300	400
Test data points	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
Penalization points	1	1	1	1	1	1	1	1	1	1
Penalization weight	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
Activation	Th	Th	Th	Th	Th	Th	Th	Th	Th	Th

Table D.10 Weak PINNs models used to study the impact of the data in Figure 7.4.

Characteristics	1	2	3	4	5	6	7	8	9	10
Input Dimension	1	1	1	1	1	1	1	1	1	1
Output Dimension	4	4	4	4	4	4	4	4	4	4
NN Depth	3	3	3	3	3	3	3	3	3	3
NN Width	20	20	20	20	20	20	20	20	20	20
Training data points	10	20	40	60	80	100	150	200	300	400
Test data points	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
Penalization points	1	1	1	1	1	1	1	1	1	1
Penalization weight	10^{-7}	10^{-7}	10^{-7}	10^{-7}	10^{-7}	10^{-7}	10^{-7}	10^{-7}	10^{-7}	10^{-7}
Activation	Th	Th	Th	Th	Th	Th	Th	Th	Th	Th

Table D.11 PINNs models used to study extrapolation in Figure 7.5.

Characteristics	1	2	3	4
Input Dimension	2	2	2	2
Output Dimension	1	1	1	1
NN Depth	3	3	3	3
NN Width	20	20	20	20
Training data points	100	100	100	100
Test data points	200	200	200	200
Penalization points	200	0	200	0
Penalization weight	0.01	0.01	0.01	0.01
Activation function	Tanh	Tanh	Tanh	Tanh
Type	Classic	Classic	Classic	Classic

Table D.12 PINNs models used to solve inverse problems in Figure 7.6 and Figure 7.7.

Characteristics	1	2
Input Dimension	2	2
Output Dimension	1	1
NN Depth	3	3
NN Width	20	20
Training data points	5000	5000
Test data points	100	100
Penalization points	500	500
Penalization weight	0.01	0.08
Activation function	Sin	Sin
Type	Classic	Classic

APPENDIX E LIBRARIES AND VERSIONS USED IN THE CODE

In this appendix we summarize the libraries used in the codes and the versions.

Table E.1 Used libraries

Name of the library	Version
Python	3.6.13
Numpy	1.19.5
OpenCV	4.5.5.62
Spyder	5.0.5
TensorFlow	1.14.0
TensorFlow GPU	1.14.0
Tikzplotlib	0.9.12
Scipy	1.5.4
Scikit-Image	0.17.2
Matplotlib	3.3.4