POLITECNICO DI MILANO
Corso di Laurea Magistrale in Ingegneria Aeronautica
Dipartimento di Scienze e Tecnologie Aerospaziali

Development of a stochastic parametric tiltrotor whirl flutter numerical predictor using general-purpose multibody and optimization software

Relatore: Prof. Pierangelo Masarati
Correlatore: Dott. Andrea Zanoni

Federico Guerroni, matricola: 899512

Anno Accademico 2019−2020

To mom and dad

Abstract

The maximum horizontal flight speed of tiltrotor aircraft is limited by the aeroelastic instability called "whirl flutter". Nowadays, though, the phenomenon understanding is still limited. For this reason, filling such knowledge gap is a well established goal.

In order to gain better understanding of whirl flutter, the current research efforts are focusing on investigating the stability of arbitrary tiltrotor configurations. This is supported by the construction of wind tunnel test-beds designed to model and analyze a wide range of tiltrotor assemblies.

Alongside such experimental research campaigns, the need of developing strong and flexible numerical tools is mandatory to both guide and gain information from the real test-beds. As consequence, in this work, the development of *TiPa* is presented. The software defines a flexible interface to the general-purpose multibody dynamics solver MBDyn to provide a parametric tiltrotor model generation and investigation tool. The tool is combined with DAKOTA, an Uncertainty Quantification (UQ) software. This cooperation gives birth to a complete aeroelastic stochastic predictor conceived to simplify the identification of the system design parameters that mostly affect the stability margin of an arbitrary tiltrotor configuration. The use of the generalized Polynomial Chaos Expansions (gPCE) provides an efficient and versatile forward propagating UQ tool, which delivers useful information through the assessment of the system stochastic response to a set of non-deterministic inputs.

The thesis proposes two versions of the parametric model generator. One is based on the definition of complete MBDyn tiltrotor models while the other one relies on the generation of two different subsystems to be assembled through a substructuring approach. The second strategy relies on adaptations of both the multiblade coordinates (MBC) transformation and the Craig-Bampton approach applied to MBDyn multibody formulation.

Riassunto in italiano

Questo lavoro presenta lo sviluppo di un solutore parametrico e stocastico per lo studio dell'instabilità aeroelastica chiamata "whirl flutter" tipica dei convertiplani. La tesi presenta sia la completa concezione teorica dello strumento che la sua applicazione.

Lo formulazione del solutore si basa sull'interazione di tre elementi: un generatore parametrico di modelli di convertiplano chiamato *TiPa*, il solutore aeroelastico multicorpo MBDyn e il software DAKOTA. L'interazione dei tre è disegnata per fornire un completo strumento di investigazione aeroelastica in grado di valutare gli effetti della presenza di parametri non-deterministici sulla risposta dei modelli testati. Lo scopo principale di ciò è facilitare l'identificazione dei parametri che più contribuiscono all'insorgere del whirl flutter.

Lo sviluppo di *TiPa* è iniziato prendendo ispirazione dal progetto TRAST. L'obiettivo di tale progetto è l'incremento dell'attuale livello di conoscenza del fenomeno aeroelastico, promuovendo l'identificazione dei fattori che più contribuiscono alla nascita del whirl flutter in una generica configurazione di questi velivoli. *TiPa* è stato disegnato appositamente per fornire una interfaccia flessibile alla modellazione di set-up arbitrari in galleria del vento.

La concezione parametrica di *TiPa* è stata fondamentale per introdurre un metodo di propagazione in avanti di parametri incerti. L'implementazione di tale formalismo, possibile grazie al software DAKOTA, produce un nuovo cluster di informazioni che favorisce l'investigazione parametrica dell'instabilità. La tesi riporta una dettagliata revisione dei principali metodi che possono essere usati nella propagazione di parametri incerti, con il fine di individuare lo strumento ottimale da introdurre nel formalismo completo del solutore. La versione non-intrusiva della *generalized Polynomial Chaos Expansions technique* (gPCE) è stata selezionata per lo scopo in quanto metodo più efficente, versatile e adatto alla specifica applicazione.

La completa definizione del solutore parametrico e stocastico è data dalla combinazione di *TiPa* e DAKOTA. Questo permette sia l'esecuzione di com-

plete analisi di sensitività globale e locale, sia l'investigazione della risposta del sistema a input non-deterministici attraverso la definizione delle funzioni di densità di probabilità delle risposte e all'identificazione stocastica delle curve V-f e V-ξ. Le simualazioni di DAKOTA e *TiPa* forniscono quindi un ampio range di informazioni a supporto dell'identificazione dei parametri più pericolosi per lo sviluppo del whirl flutter.

TiPa permette la generazione dei modelli di convertiplano attraverso due approcci differenti. Il primo si basa sulla definizione completa di un modello MBDyn dell'intero set-up desiderato. Il secondo, invece, si basa sulla generazione individuale di due sottomodelli rappresentanti rispettivamente il sistema ala e il sistema rotore. Questi sono uniti in un secondo momento grazie un processo di sottostrutturazione. Il primo approccio, come presenta la tesi, è limitato dalla forte dipendenza del processo dalla convergenza del software multicorpo. Questo, in certi casi, può minare l'identificazione stocastica della condizione di flutter. Il secondo approccio, invece, cerca di superare questo limite. Per fare ciò è stato introdotto un adattamento innovativo della trasformazione in coordinate multipala per i gradi di libertà multicorpo di MBDyn. La sottostrutturazione viene eseguita tramite una versione del metodo di Craig-Bampton applicata a MBDyn. Tuttavia, allo stato di sviluppo attuale, il secondo metodo di modellazione non permette ancora di ottenere una corretta rappresentazione del sistema generato a causa di inesattezze nell'identificazione delle frequenze associate ai modi propri del convertiplano.

Per questo motivo, il completamento dello sviluppo di tale approccio viene lasciato alla ricerca futura.

Acknowledgments

First, I want to thank my thesis supervisor Prof. Masarati for giving me the opportunity to work on this project. The very different phases required in the development of *TiPa* and its combination with DAKOTA allowed me to experience the highs and lows of the researcher life and helped me developing my engineering and logical skills.

Second, I want to thank Alessandro and Andrea for their kind and fundamental assistance in the development of my work. Their help was essential both in overcoming some of the hardest obstacles met during the developing process and in solving the day by day little problems.

I want now thank my family, specially my mom and dad, for being my biggest supporters during all these years. Without your help, I would have never been able to get to this point.

Then, I want to thank all my friends, the oldest one and the most recently met. To the former, I really want to express my gratitude for all these years we have been together. For me, it is like having an amazing second family. Thanks for your constant support and love.

In conclusion, I say a huge "thank you" to Politecnico di Milano for being hard sometimes and teaching me not to give up. These years made me me grow so much both as a rational thinker and as a human. Here, I found an amazing everyday multicultural environment filled with extremely interesting people. Thank you as well for giving me the opportunity to live the greatest experience of my live with the Exchange program to Israel.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 The tiltrotor

A "tiltrotor" is defined as:

"An aircraft with rotors that can be tilted"

(ref. [9]). The term refers in general to machines whose rotors may act both as propeller and as vertical lift generators according to the specific flight condition.
The conceptual definition of such aircraft configurations is connected to the inevitable speed and range limits affecting conventional helicopters. Horizontal rotors represent a perfect design option when specific aircraft performances are required (such as vertical take-off capabilities, high manoeuvrability at low speed...), but they inevitably limit the machine maximum speed. This makes conventional helicopters much slower when compared to fixed wing aircraft since the forward speed is provided only by a small component of the lifting force. Tiltrotors try to overcome such limitation introducing a fixed lifting surface. This one operates when the aircraft is in horizontal flight (the so called "airplane" mode) allowing the complete tilt of the rotor to be perpendicular with respect to the forward flight speed. In this way, the rotors produced forces can be entirely allocated to generate thrust. Consequently, tiltrotors in airplane mode can fly faster than conventional rotorcrafts thanks to their lift/propulsion system which makes them more similar to turboprop aircraft rather than to helicopters.

1.2 Historical frame

The conceptual idealization of tiltrotor aircraft dates back to 1920 when the inventor F. Vogelzang patented its aircraft with tilting rotors (ref. [44]).

Despite the design was never brought to life, a new concept was born. In the following years, other inventors worked on the revolutionary idea. The most relevant outcome of this was the "Heliplane" patented by Baynes in the 1930's (ref. [27]). As its predecessors, this machine was never assembled, but its design included many of today's tiltrotor aircraft. A sketch of it is reported in figure 1.1.

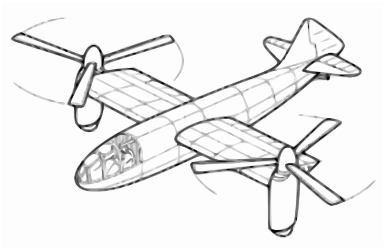

Figure 1.1: Baynes Heliplane

The first tiltrotor prototypes were assembled when well established aircraft companies and governments started to show interest in the configuration. In the 1950's, the Department of Defense of the United States of America commissioned McDonnel Aircraft, Sikorsky Aircraft and Bell Helicopter the design of three "convertiplane" aircraft (this was tiltrotors original name) which purpose was the investigation of the new concept feasibility. The machines were named XV-1, XV-2 and XV-3 respectively (ref. [8]). None of the models entered production, but their development provided a new cluster of information to their engineers. Among them, the Bell XV-3 provided unprecedented understanding of such machines aeroelastic behavior.

The initial version of the XV-3 was provided with three bladed rotors. This implementation proved to be dangerous since the aircraft had different accidents due to the occurrence of rotor dynamic instabilities. This led to the design and construction of a new version provided with two bladed rotors (see Fig. 1.2).
The successive series of tests provided enough data to allow in 1966 [15] the development of an early tiltrotor "whirl flutter" characterization. This term refers to a gyroscopic aeroelastic instability affecting some configurations of fixed wing aircraft. The phenomenon develops in high speed flight

and usually is the limiting factor in an aircraft max speed definition. For further details about the phenomenon see section 1.3. That said, the Bell XV-3 had many flaws in its design. This included the fact that the maximum horizontal flight speed (115 knots) was not enough to justify the new concept and the pilot had to handle an extreme workload to manoeuvre the aircraft. Despite that, the XV-3 successful test campaign managed to prove the feasibility of the new aircraft concept specially of the in flight conversion capability from helicopter to fixed wing configuration [23].

Figure 1.2: Bell XV-3 in airplane mode

In the 1970's the raising interest in the technology lead to the establishment of different tiltrotor related project thanks to the cooperation between NASA and the US Army. The most iconic product of such collaboration was the Bell XV-15. The design process focused mostly on solving the XV-3 problems. A new flight control system was combined with automatic stability augmentation tools in order to overcome the limited handling capabilities of the XV-3. In the XV-15 the installed engines became two and were placed at each wing tip. This completely removed the complex shaft mechanisms connecting its predecessor central engine to the wing tip rotors. This required the engineers to design a transmission connecting each engine to both the three-bladed rotors in order to grant symmetric thrust even in case of an engine failure. These new implementations clearly explained why tiltrotor aircraft dynamic stability can be hard to assess precisely. The large number of interacting elements that are vital to the aircraft correct behaviour introduce high level of complexities in the assessment of its dynamic stability behavior. That is true because the behaviour of each sub component is coupled to many others and influence the overall system response

to specific inputs. As result, a very wide range of parameters can influence the aircraft behaviour making extremely difficult the instability phenomena characterization.
The aircraft flight envelope was far more interesting in terms of real scale possible implementations compared to its predecessor thanks to its new design solution. The XV-15 remained a prototype since only two of them were built. For further reference see [23].

Figure 1.3: Bell XV-15

In 1981 the JVX project was born. JVX was a rotary wing aircraft development program focused on designing a machine to meet various service needs. Such specific requirement could be satisfied by the rising tiltrotor technology (ref. [20].) Within this frame, the partnership between Bell Helicopter and Boeing Helicopters, started in 1983, led to the creation of the V-22: an evolution of the original JVX model (Fig. 1.4). This aircraft, also known as Osprey, is the first example of a tiltrotor machine developed for military purposes. To ensure the aeroelastic stability of the aircraft, a series of experiments were run at NASA Langley Research Center. The many factors contributing to whirl flutter required a very broad series of experiments in which a variety of scaled V-22 model configurations were tested in order to identify the best possible building solution (ref. [21]). This complete aeroelastic parametric investigation process was inevitable due to the complex and sometimes unpredictable interactions among the many assembly subcomponents. The aircraft performances needed to match specific U.S Army and Navy requirements. This forced the engineers to overcome an extensive range of new problems [13] and made the V-22 a very expensive machine. Despite this, the Osprey both provided better understanding of tiltrotor aircraft instability phenomena and proved the commercial value of

such machines with competitive performances. It became operative in 2005 and it is still in service today (ref. [31]).

Figure 1.4: Bell/Boeing V-22

In 1996 a Bell-Boeing cooperation aimed at developing the first tiltrotor aircraft for civil transportation. Conceived to transport a small number of people, the aircraft was meant to grant faster point to point transportation for business purposes and for search and rescue missions. In 1998 the project became Bell-Agusta and eventually AgustaWestland (today owned by Leonardo S.p.A.) (ref. [31]). The outcome of such development is the AW609 (fig 1.5) (earlier called BA609). The aircraft is not operative yet.

Figure 1.5: AW609

In 2013, Bell Helicopter partnership with Lockheed led to the Bell V-280 Valor development. The aircraft was born within the JMR-TD project and is designed to meet specific Army needs. Very little information is avail-

able to the public for confidentiality reasons but the V-280 has the specific purpose to reach "over twice the speed and range of the current vertical lift fleet" (ref. [2]).
Bell in 2016 started working as well on the V-247 Vigilant which is an unmanned tiltrotor for surveillance and reconnaissance purposes. The development is still in a very early stage.

1.3 Whirl flutter

"Whirl flutter" is an aeroelastic instability phenomenon that is generated by the elastic interaction of the wing motor mounting structure and the propeller dynamics.
It was discovered analytically in 1938 (ref. [43]) as precession-type instability in a flexibly mounted aircraft engine-propeller combination. Its discovery was associated to the turbo-prop aircraft configuration.
Both Bell (see section 1.2) and NASA led the investigation of the phenomenon in the 1960's. NASA proved great interest since compromised engine-propeller connections led to the destruction of two Lockheed Electra turboprop aircraft. The company in 1967 wrote an extensive report (ref. [34]) with an detailed review and interpretation of such the whirl flutter instability. The phenomenon is explained through a simple propeller/power plant analytical model. Such schematic representation does not match the more complex dynamics a tiltrotor aircraft, but it provides a very intuitive representation of the simplest manifestation of the phenomenon: the classic propeller whirl flutter instability.

1.3.1 A simple model

The model presented by NASA is composed of a four bladed propeller connected to the ground through a rigid shaft and two springs (S_θ, S_ψ) representing the actual stiffness of the otherwise flexible shaft. The two springs, which are connected at a pivot point at distance a from the rotor disk, allow only the assembly rotations about two perpendicular axes. No independent degree of freedom is assigned to each blade. The entire range of motion of the system can be described by the pitch θ and yaw ψ angles (see Fig. 1.6). The dynamical behaviour of the system is affected by the angular speed Ω. When the rotor is not spinning, the natural vibration modes related to each degree of freedom can be spotted individually. As long as the rotor angular

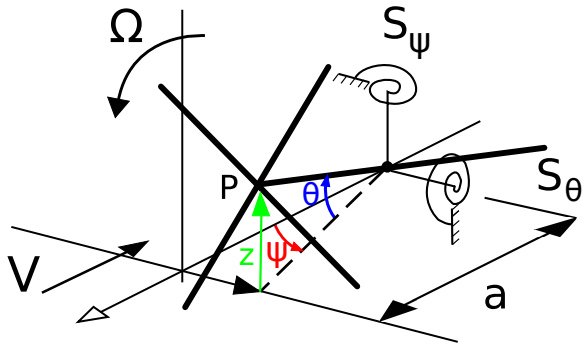

Figure 1.6: Two DOFs model sketch

speed Ω is set different to zero, instead, these two are coupled due to the gyroscopic forces generated by the additional motion. The resulting modes are called "precession" ones. That is because, since some flexibility is introduced in the system by the equivalent springs, the center of the rotor P rotates about the axis connecting the middle point of an ideal rotor rigidly connected to the ground.

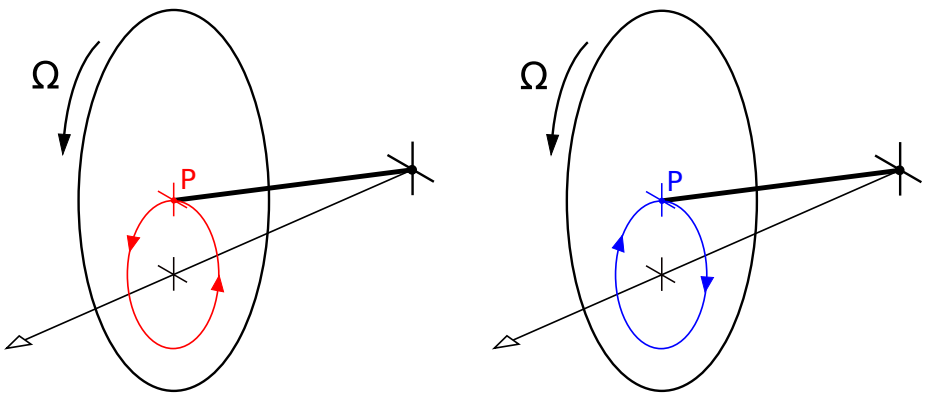

Figure 1.7: Forward (on the left) and backward (on the right) whirl modes visualization in vaquo

This possible precession motions are two and can show up both in the direction of the spinning rotor and in the opposite one. For this reason are called "forward" and "backward" whirl modes respectively (see Fig. 1.7). The modes, by themselves, are not dangerous. The situation can get problematic when the aerodynamic forces are introduced in the system.
The rotor motion triggered by the two whirl modes, directly influence the aerodynamic forces generated by the blades. This happens since the rotor

angular speed and the motions due to the flexibility of the rotor shaft affects the way each blade incoming speed is defined. As consequence, a new set of aerodynamics forces is generated and can lead to a divergent behaviour of the system. The nature of the generated forces is strongly connected to the motion that produced them. To explain the concept, here some combinations of trigger motions and resulting loads are presented.

The first case to introduced here shows the effects produced by the forces generated by the blade due to a vertical motion (pitch) of the propeller. Keeping the simplified model (rigid shaft plus equivalent springs), this motion can be described through the pitch angle θ only. Considering the rotor as a rigid disk with the airspeed hitting it perpendicularly, the inclination of the shaft by an angle θ tilts as well the orientation of the rotating propeller. This inevitably generates an angle between the incoming air and such disk and, consequently, a velocity component V_θ parallel to the latter. Since the propeller is spinning, assuming it is composed by an even number of blades, it is always possible to identify two of them rotating at opposite sides with respect to the disk center. In case one of the two, due to the rotation, is directly facing V_θ, its airfoils perceive both a higher AoA and an incoming speed with respect to the zero pitch angle condition. This generates a local increase in lift in such areas. The opposite will happen at the other side of the propeller center. As results, a "yaw" moment $M(\theta)$ is generated about the axis aligned with the vertical displacement of the beam in addition to a force $L(\theta)$ collinear with the direction of the "pitch" motion. The former component is the most dangerous for the development of the whirl flutter instability. That is because such moment acts in the direction of the "backward" whirl mode motion. This may triggers the unstable phenomenon.

In case the propeller is shifted vertically with a $\dot{z}$ velocity, the speed component $V_{\dot{z}}$ parallel to the disk comes from the direction opposite to the motion. The way this new speed influences the propeller behaviour is the opposite with respect to the one connected to V_θ. This means that the airfoils in the blades moving upwards with respect to the vertical motion are hit by air with greater speed at an increased AoA with respect to the ones on the opposite side. As consequence a "yaw" moment $M(\dot{z})$ and a vertical force $L(\dot{z})$ are generated in the direction opposite to those generated by a pitch rotation $M(\theta)$ and $L(\theta)$. This means that such contributions will stabilize such the motion that generated them.

The last situation presented here is given by the influences of the extra air-speed components due to a pitch rate. From the side, with the pitch axis going out of plane, a positive pitch rate variation makes the disk tilt with respect to such axis. One of its sides (the bottom one for a positive

pitch rate) moves towards the airspeed while the other one in the opposite direction. The former is affected by a reduction in its sections' AoA with a consequent lowering of the generated forces, while the latter reacts in the opposite way. As result, a moment $M(\dot{\theta})$ rises whose orientation contrasts the motion that originated it.

From these considerations it is possible to understand how the aerodynamic forces and moments acting on the propeller may tent sometimes amplify to its oscillations. Since they do it by triggering the whirl motion of the propeller, this aeroelastic phenomenon is called whirl flutter.

A key point that emerges from NASA report, is the strong dependence of the phenomenon on a very large number of parameters whose connection to whirl flutter may be highly unpredictable. This required (and still requires today) **parametric analysis** of such variables to correctly estimate their specific influence on the phenomenon. In a simple model like the one presented here, the analytical assessment of how different parameters influence the stability of the system is possible. Once the model complexity rises, such opportunity inevitably disappear. The consequence of this, for instance, led to an extensive and inevitable wind tunnel testing campaign for the V-22 Osprey (see 1.2) with the specific purpose of spotting the best possible building configuration to optimize the aircraft aeroelastic behaviour.
This simple formulation is a very drastic approximation of the real scale problem, but it helps understanding how the whirl flutter is triggered by some specific interactions between the propeller rotating components dynamics, aerodynamics and the engine mounting structure. A tiltrotor simplified model can be little complication of the model presented here. This is true since both propellers and tiltrotor basic principles are similar. What really differs is the complexities in the realization of the propulsion systems of the latter which makes titrotor aircraft much more complicated in terms of number of interacting elements and connections. The direct consequence of this is that larger numbers of interactions may give birth to new trigger mechanisms to such instability phenomena.
Some critical elements in that make tiltrotor aeroelastic investigations more complicated with respect to turboprop aircraft ones are:

1. the increased pitch flexibility due to the need to rotate the engine

2. the tilting of thrust vector through the hub gimbal joint

3. the extra degrees of freedom due to flapping hinges and gimballed hub

4. the much larger dimensions of the propeller blades

These inevitable design solutions reduce the proprotors aeroelastic stability. That is because, due to the joints implemented, the shaft, pylon and wing structure cannot directly provide restoring moments to the rotor in case moments affecting the rotor stability rises. Another key element not to neglect is the high velocity inflow typical of proprotor "airplane mode". This is the most common flight condition in which whirl flutter may rise. That is because tiltrotors in cruise configuration combine the high speed incoming air with large flexible blades. Specific forces and moments rise due to these factor that requires extreme care in tiltrotors design (ref. [18]). Hence, investigating whirl flutter instabilities is both a fundamental and a very complex component in the design and certification of tiltrotor aircraft.

What emerges from the simple analysis above and applies to real scale tiltrotors as well is that, as in many others aeroelastic instability phenomena, the incoming air speed is crucial in defining the thin layer between a stable flight condition and an unstable one. This usually is the key limiting factor in the definition of the aircraft maximum flight speed. This was immediately clear since the first real scale tests on the XV-3 and has been a limit to the effectiveness of the tilrtotor technology since then. Investigation of aeroelastic instabilities are then one of the most crucial challenges in the development the next generation of proprotors.

1.3.2 Tiltrotor whirl flutter investigation

As already mentioned in section 1.2, whirl flutter instability has been a key element alongside the entire history of tiltrotor aircraft. Despite the phenomenon investigation and correct assessment has been a priority since its first individuation during the XV-3 testing phase, the purposes of whirl flutter preliminary analysis changed in time. In the 1960's, tiltrotor aeroelastic behaviour needed to be correctly addressed in order to prove the feasibility of the concept. This led to analysis focus on the specific models (the XV-3 and the XV-15) and to their early development. From the 1980's, instead, whirl flutter investigation changed goal. The tiltrotor concept was mature enough to allow engineers to focus its implementation in commercial and military aircraft. Alongside with this need, new analytical and numerical tools were developed with the specific purpose of investigating the phenomenon predictability. Different strategies and new were adopted which led to today's and tomorrow's whirl flutter investigation strategies. To avoid unnecessary

digressions, in this sections are briefly presented the most important investigation projects and strategies adopted from the 1980's.

1980's-today

In 1977 NASA published a technical paper [22] trying to asses whether it was possible to correctly predict the whirl flutter occurrence in a tiltrotor aircraft thanks to simple models. The work proved its point by comparing the analytical results with experimental assessments and showed how the important was to gain better theoretical understanding of the aeroelastic phenomenon.
Starting from the 1980's Bell leadership in the field, led to the company cooperation to the JVX project. The V-22 was born within this frame (see 1.2). In the early 2000's the JVX became part of the WRATS project. The *Wing and Rotor Aeroelastic Test System* was a whirl flutter analysis test bed designed to investigate whirl flutter phenomena. (ref. [48]) The project started in 1994 and was developed by a collaboration of NASA and Bell with the specific purpose to investigate aeroelastic phenomena limiting the commercial implementation of tiltrotors. The main focus was on the active vibrations control techniques to enhance proprotors performances, but investigation of passive instability damping techniques was part of the program as well. The wind tunnel model was updated and modeled as long as experiments were run.
WRATS studies successfully increased the understanding of tiltrotors aeroelastic phenomena. Parametric influence of some tiltrotors design parameters (including blade precone, flapping stiffness...) were assessed. Some limitations, though, affected the program. First, the tiltrotor model was derived from the V-22, a military aircraft. This made its design data classified limiting possible external research contributions. Second, the WRATS model was not very flexible in its configuration including the type of rotors that could be installed and the measurements data that could be extracted. These factors limited the some of the possible additional positive outcomes of the project.
Politecnico di Milano university contributed to the project modelling WRATS with the multibody general purpose software MBDyn [26].

Future

Successive tiltrotor implementations in the years provided a lot of data and information about possible specific designs. The desire to develop the next generation of tiltrotors, though, moved researchers focus on seeking the best

possible design configurations capable of pushing the concept even further. This requires the best possible understanding of the whirl flutter instability. In order to gain better insights on the phenomenon, the TRAST (*TiltRrotor Aeroelastic Stability Testbed*) project was conceived. TRAST relies on an extremely versatile proprotor wind tunnel test-bed to assess the parametric influence of different components on the model aeroelastic response. The system, designed starting from the XV-15 and meant to represent a generic tiltrotor model, is placed at NASA Langley Transonic Dynamics Tunnel (TDT). TRAST is designed specifically for research purposes. For this reason, there are no confidentiality restrictions related to the project experimental set-up and research outcomes (ref. [20]).
Numerical parametric analysis of TRAST aeroelastic behaviour have already been conducted (ref.[50]).

1.3.3 Numerical prediction tools

The TRAST project provides the optimal environment to increase the whirl flutter phenomenon understanding thanks to the open access to the test-bed design. Thanks to this, researchers all around the world can develop numerical tools to assess the real wind tunnel model behaviour with two important benefits.
First, the formulation of *ad hoc* prediction tools is fundamental for the effectiveness of the wind tunnels tests since they can help identifying which of the real scale simulation provide the most useful information. Second, TRAST test-bed experiments help validating the numerically estimated results improving the level of confidence in the simulation tools. This unprecedented level of interaction is granted by the possibility to share information without confidentiality restrictions and it is crucial to eventually develop an effective generic tiltrotor configuration aeroelastic behaviour prediction tool.

Over the last two decades, different modern numerical tools have been successfully used to model whirl flutter (ref.[49]). Among them, multibody software proved to be accurate in modelling proprotors complex dynamics (ref. [12]). That is true thanks to such numerical methods precise modeling of:

1. Large displacements
2. geometrical non-linearities
3. beam elements to model slender bodies (blades, wings)

4. large rotations

which are necessary to depict such aircraft behaviour.

MBDyn (`http://www.mbdyn.org/`) is a free general purpose multibody software developed by Politecnico di Milano in the last 20 years. It was born with the specific purpose of providing autonomous modeling capabilities of generic problems related to the dynamics of complex aeroelastic systems, specifically rotorcraft and tiltrotor systems. MBDyn typical application is related to the solution of initial value problems, in the form

$$\begin{aligned} \mathbf{M}\dot{\mathbf{x}} &= \dot{\mathbf{p}} \\ \dot{\mathbf{p}} &= \mathbf{f}(\dot{\mathbf{x}}, \mathbf{x}, t) + \boldsymbol{\phi}_{/x}^{T}\boldsymbol{\lambda} \\ \boldsymbol{\phi}(\mathbf{x}) &= \mathbf{0} \end{aligned} \tag{1.1}$$

where the first equation represent the nodes momenta definitions, the second one their Newton-Euler equilibria and the last one the algebraic constraints. The latter are directly imposed at configuration level, without any time differentiation, resulting in an index-3 Differential Algebraic Equations (DAE) system. The current software implementation is able to predict tiltrotor aeroelastic stability and for this reason it has been used extensively the whirl flutter instability (ref. [26]) and is used as investigation tool in this work.

1.4 *TiPa*

Within the current research frame, Politecnico di Milano developed *TiPa*. The name stands for *Tiltrotor Parametric model generator* and refers to a MATLAB tool coded at the Dipartimento di Scienze e Tecnologie Aerospaziali. The purpose of *TiPa* is to automatize both the generation and the aeroelastic assessment of arbitrary MBDyn tilrotor wind tunnel models. The software is designed to be as modular as possible in order to give access to whatever system design feature.
The formulation is enriched with DAKOTA software's Non-intrusive Polynomial Chaos Expansions (PCE) based sensitivity assessment tools.
TiPa ultimate goal is providing a flexible multibody based tiltrotor whirl flutter investigation tool with forward propagating uncertainty quantification capabilities.

1.4.1 The approach

This section aims at briefly introducing *TiPa* main working principles. A more detailed formulation of the concepts presented here can be found in section 4.

The user can interact with *TiPa* through the use of three input cards. Two of them control the geometrical and structural definition of, respectively, the wing and the rotor parts of the complete system. The third one, instead, controls the software behaviour.

Two main tiltrotor modelling approaches are implemented in *TiPa*. One relies on the complete definition of the tiltrotor system. The other one makes use of two independently designed aeroelastic sub–models representing both wing and rotor/pylon systems to be later assembled through a substructuring process.

Despite the selected approach, the tool transforms the user defined data into input files for the software MBDyn. The information required to assess the dynamical behaviour of the system is extracted from MBDyn eigenanalysis (ref. [24]) through the collection of the matrices describing the system.

In case the two subsystems are defined independently, the rotor degrees of freedom are transformed through an original adaptation of the Multiblade Coordinate Transformation (MBC) (ref. [4]) applied to the MBDyn matrices describing such subsystem. This introduce a more consistent and reliable point of view about the rotor description removing as well the periodicity from the system. The two aeroelastic models can consequently be assembled. The process is based on a Craig-Bampton (ref. [5]) like substructuring process developed for the MBDyn multibody matrices. As consequence, a representation of the complete tiltrotor is defined.

After the generation of the tiltrotor matrices according to either one or the other approach, the stability of the system is assessed solving the associated eigenvalue problem.

Non-intrusive generalized Polynomial Chaos Expansion (PCE) based Uncertainty Quantification methods are introduced in the formulation thanks to DAKOTA (`https://dakota.sandia.gov/`), an open source software under GNU LGPL licence widely used in the research community to perform uncertainty quantification and optimization analysis (ref. [1]). The combination of *TiPa*, MBDyn and DAKOTA allows the formulation of a complete parametric aeroelastic predictor with forward propagating uncertainty quantification. The gPCE formulation flexibility allows the computation of the desired system stochastic responses to whatever set of non-deterministic in-

puts the user desires. The execution of local and global sensitivity analysis is possible as well.

The complete cooperation among the software provides a set of very useful investigation parameters to help the individuation of the most critical design features in the development of whirl flutter.

Chapter 2

Uncertainty Quantification

The strong connection between aeroelastic instabilities and aircrafts safety and performance makes such phenomena investigation a fundamentental part of the design and certification phases.

The system aeroelastic response is usually influenced by a large number of parameters. In a relistic scenario, though, not all of them may be known in a deterministic sense. The problem can be partially addressed through the integration of uncertainty quantification formulations (UQ) in the aeroelastic models. The estimation of the aircraft aeroelastic response in a non-deterministic way provides better and more reliable observations of the system behaviour. For this reason, the implementation of UQ methods in aeroelasticitity is a current topic of investigation (ref. [32], [7]).

Within this research frame, the development of reliable numerical whirl flutter investigation tools with uncertainty quantification capabilities is discussed in the chapter. This is meant to both increase the numerical results robustness with respect to the system missing information and allow the execution of parametric sensitivity analysis.

This chapter starts with an introduction to the different uncertain parameters that may affect a generic system. The focus is then shifted on some of the most important UQ technique available to eventually identify the most suitable method to implement alongside *TiPa* formulation.

2.1 Uncertainty types

Different types of uncertainty can be classified according to the way they are introduced in a system model.
The most important sources of uncertainty are:

1. lack of knowledge
2. randomness

"Lack of knowledge" (also called "ignorance"), in general, refers to the disparity between what is known and what needs to be known to correctly understand and model a specific phenomenon. Predictions derived from a model with some levels of missing information inevitably lack in terms of robustness and reliability. This type of uncertainty cannot be modeled probabilistically (ref. [3]) and is usually classified as **epistemic**.

"Randomness", instead, refers to all the sources of uncertainty derived by the non-deterministic definition of some system parameters. In case the probabilistic content of such parameters is known, their influence can be accounted in the system response hence providing a useful cluster of information. This uncertainty type is usually called **aleatory**.

2.2 Propagation methods

In this section are presented the different methodologies that can be used to handle uncertainty within a given model. Two families of uncertainty propagation analysis are defined.

Forward propagating UQ models are designed to assess the way an uncertain input value affects the system response. This methodology, in general, requires the knowledge of the probabilistic content of the identified value from which the output statistical content is estimated.

Backward propagating UQ models, instead, work in the opposite direction. From the probabilistic knowledge of the system response they evince the previously unavailable stochastic description of a random input variable. For this reason, these methods are used to solve the so called called "inverse characterization" problems.

The definition of forward and backward propagating UQ models shows how their implementation may fits entirely different purposes and depends on:

- the precision of the model definition
- the available information about the system responses
- the desired UQ analysis outputs

Backward propagating models requires the availability of an extensive collection of the modeled system responses. This method is generally used to complete the characterization of a numerical model using the data collected from an intense experimental investigation campaign. That makes this approach unsuitable for the development of *TiPa*.

For such purpose, instead, forward propagating UQ models are more appropriated. Given that *TiPa* is designed to generate numerical predictions starting from a set of data defined by its user, the introduction of uncertain input variables through forward propagating methods allows:

1. computing the system non-deterministic response to such inputs
2. run sensitivity analyses over the parameters variability

Consequently, the implementation of forward propagating methods inevitably increases *TiPa* estimations robustness and improves its parametric investigation capabilities.

2.3 UQ methods classification

"UQ methods" refers to a very large number of mathematical and statistical tools. According to the way each specific method works, they are classified in specific families. The focus of this section is to introduce such methods classification and what are the basic principles behind each one of them. The main families are:

1. Sampling methods
2. Interval analysis
3. Stochastic expansion methods
4. Info-gap theory

According to section 2.2 discussion, all these methods are presented in this section in their forward propagating version. This is done to analyze their possible implementation in *TiPa*.

In the discussion, a numerical model of a generic phenomenon is considered.

2.3.1 Sampling methods

Sampling methods relies on the definition of some samples. This set of variables are generated according to user defined probability distribution functions providing a non-deterministic description of the uncertain parameters. The samples are iteratively used as input to the model, which, provides specific response variables. The latter are collected and used to compute a non-deterministic description of the response.

Sampling methods are among the oldest and most established UQ techinques. They represent the easiest implementation of UQ tools to assess a general model probabilistic response. The main reason for this is that sampling methods do not require the modification of whatever is inside the numerical model. This makes them a very robust option since, if properly tuned and designed, they always provide the desired response statistical content.

The biggest drawback of sampling methods is the strong dependency of the analysis results on both the dimensions and on the specific components defining the samples cluster. To make sure the UQ analysis provides reproducible results, it is often necessary to provide to the simulation a very large number of samples. This inevitably increases the time required to complete the UQ assessment making sampling methods very impractical in most situations.

The best known sampling techniques are the **Monte Carlo** (MC) and the **Latin Hypercube Sampling** (LHS) methods.

MC is the easiest implementation of the concept since the samples are simply selected at random from a defined probability distribution function (ref. [28], [36]).

LHS formulation is more complex (ref. [41]). This sampling method was developed with the purpose of reducing the number of samples used by MC to compute the system response probability content while maintaining the same results accuracy. The key idea behind this is the "stratification" of the input probability distribution function. The concept is based on dividing the input cumulative distribution function into equally probable intervals and randomly sampling input variables from each sub interval. Only one sample is selected inside a single stratification. The concept is better visualized extending the problem to a two dimensional one. In this case, since two random input variables are considered, the sampling process results in the generation of a grid where each axis represents the cumulative distribution function of one of the two variables. The technique is called "Latin Square" sampling if and only if one sample is picked from each row and column

of the 2D grid matrix. From the generalization of this concepts to multi-dimensional grids the Latin Hypercube Sampling was born.

LHS differs from MC in the sense that it has "memory". This means that in such family of methods is important for each new sample to remember where each of the previous values were selected. This does not happen in MC introducing an extra level of randomness. When compared to MC, LHS proves to be more accurate since it requires less samples to provide the same level of information. This is mostly due to the fact that LHS forces the system to generate input parameters from each part of the specific probability distribution function (including the tails). This is not certain in a Monte Carlo method due to the shape of the probability curve. As the number of uncertain variables grows, the same happens to the difference between the number of samples required by the two methods to obtain comparable results in terms of accuracy. Despite this, LHS still requires the simulations to be run with a very wide amount of samples underlining once again how these methods are not practical in very complex applications.

2.3.2 Interval analysis

This methodology is based on the assumption that the value of an uncertain parameter is defined within an **interval**. This means that its actual definition lies in between two possible realizations of it.
A generic interval is usually defined in this form:

$$\mathbf{x} = [\underline{x}, \overline{x}] \tag{2.1}$$

Where $\underline{x}, \overline{x} \in \mathbb{R}$ and $\underline{x} \leq (\geq) \ \overline{x}$. Any value inside such gap is equally possible to be assigned to the uncertain parameter. This does not mean that the variable is modelled in a non-deterministic sense through a uniform probability density function. Instead, the model represents in a reasonable way the presence of epistemic uncertainty since no knowledge is required about the variable distribution.

At first, interval analysis may look like a simple and easy to implement approach since there is no need to propagate the statistical content of any input through the model. What is needed, instead, is the computation of the interval assessing the epistemic uncertainty of the output. This does not provide any information about its statistical behaviour since each of the values between the output interval limits is equally possible to be selected.

The approach, though, is far from being immediate since this type of analysis requires a specific branch of mathematics called *interval arithmetic*. This include a series of mathematical operators acting on real valued closed

intervals. Interval uncertainties require then a specific and precise set of tools to be manipulated effectively. For this reason, the implementation of this family of methods in codes based on conventional mathematical operators is really complicated. Despite this, interval arithmetic can be used to run sensitivity analyses (ref. [29]). A quick explanation and application of interval operators can be found at ref. [17].

A not negligible possible problem to face when working with interval analysis is the divergence of the output interval. This happens when the latter is defined between numbers so far away from each other that the propagation may fail or the output may carry no useful information. Some factors contributing to this possible outcome are:

- the number of propagation steps inside the given code
- the complexity of each computation step

In general, raising the code complexity increases the possibility of diverging results. For this reason, interval analysis based UQ methods are inevitably affected by the specific application they are used for. This makes them less robust if compared to sampling methods whose convergence is always granted despite which model uncertainty is being assessed.

2.3.3 Stochastic Expansion methods

Stochastic Expansion (SE) based methods are non-sampling UQ tools. Starting from the knowledge of the probabilistic uncertainty affecting a model variable, the methods make use of an approximation of the uncertain input-output relationship to evince the statistical content of the system response. Different methods are classified according to the specific formulation used to generate such system representation.

The **Polynomial Chaos Expansions** (PCE) methods use *orthogonal polynomial chaoses* to generate an approximated representation of the system. There are different possible implementations of these methods. Here, the generalized PCE (gPCE) approach is presented. The gPCE method is based on the use of the Wiener-Askey polynomial chaos.

In 1938, Wiener realized that Hermite polynomials orthogonality properties with respect to the probability density function of a Gaussian variable could be used to solve problems describing second order random processes with finite variance. The method could be effectively applied in engineering since the majority of random physical problems have finite variance. The approach was based on the definition of an approximated description of the system behaviour based on an expansion of Hermite polynomials

(ref. [45]). The formulation was later extended by Askey which understood that also other families of polynomials are orthogonal with respect to specific probability density functions. They are grouped in the so called Askey (or Wiener-Askey) scheme of polynomials. This increased the range of possible uncertain input parameters to introduce in the formulation (ref. [11]).

A UQ techninque based on the "non-intrusive" gPCE formulation is presented here. The term in quotes refers to a method which evince the system response probability content without taking into accont whatever happens inside the existing model. From now on, this variant of the technique is simply called gPCE. The importance of this version of the formulation is explained in section 2.4.3.

According to the gPCE formulation, the outcome of a second order random process $X(\theta)$ is represented as function of the uncertain input variable θ in this way:

$$X(\theta) = \sum_{j=0}^{\infty} c^j \Phi^j(\boldsymbol{\xi}(\theta)) \tag{2.2}$$

where:

- c^j is the j-th real deterministic coefficient
- Φ^j is the j-th Wiener-Askey polynomial basis of order P
- $\boldsymbol{\xi}(\theta)$ is the random vector $\boldsymbol{\xi}(\theta) = \Big(\xi_1(\theta), \xi_2(\theta), ...\xi_n(\theta)\Big)$
- θ is the random parameter

Equation 2.2 shows the simplest description of such expansion.
The random vector $\boldsymbol{\xi}(\theta)$ collects all the uncertain variables that show up in the model. Among them we could find, for instance, geometrical parameters, external forcing elements and material properties. For sake of generality, the k-th random variable $\xi_k(\theta)$ uncertainty is modeled as function of another sub-variable θ which may be the actual trigger for the randomness.

The generation of the Wiener-Askey polynomial basis Φ^j depends on the probability distribution function describing the k-th random variable $\xi_k(\theta)$ behaviour. Assuming that such variables are modeled with continuous probability density functions, table 2.1 shows which family of Wiener-Askey polynomials must be used in the definition of the basis Φ^j. For sake of brevity, the actual generation process of Φ^j is not presented here. Specific information can be found in references [46, 47].

Random variables ξ pdf	Wiener-Askey polyn $\Phi(\xi)$	Range
Gaussian	Hermite	$(-\infty, \infty)$
Uniform	Legendre	$[a, b]$
Gamma	Laguerre	$[0, \infty)$
Beta	Jacobi	$[a, b]$

Table 2.1: Wiener-Askey polynomials and matching random variable types

The definition of orthogonal polynomials is explained in Appendix A to provide better insights on the PCE formulation .

The system response approximated model $\widetilde{X}(\theta)$ derivation is now presented. This is done by tuning the coefficients c_j such that the new system behaviour matches the real one. This can be expressed as:

$$\widetilde{X}(\theta) = X(\theta) \tag{2.3}$$

The process can be done in different ways. The basic way to do it relies on the minimization of an error definition between $\widetilde{X}(\theta)$ and $X(\theta)$. This can be done, for instance, projecting the system response against each basis function through inner products using the orthogonality of the polynomial basis or introducing a mean square minimization of the error esteem. The exact way this is done really depends on the specific application and it is not presented here. It is important to point out, though, that, for whatever method is selected, the expansion coefficients tuning relies on the combination of the input-output data collected from the actual system. For this reason, consequently, some numerical simulations have to be executed. This does not mean that stochastic expansion methods work similarly to sampling methods. The number of input-response combinations required by gPCE are much less compared to those necessary to (for instance) MC to complete the UQ assessment. The gPCE method uses the simulations only as an instrument for the tuning of its expansion coefficients c_j. This, as a rule of thumbs, requires a number of simulations close to the number of such coefficients.

The approximated system behaviour ($\widetilde{X}(\theta)$) exactly matches the real model one ($X(\theta)$) only if all the infinite terms and coefficients are considered in equation 2.2. In reality, such series must and can be truncated without losing too much information. This eventually allows:

- the computation of the c_j coefficients
- the generation of the orthogonal basis Φ^j

The infinite expansion defined in equation 2.2 is then truncated at a given value S. Hence, the model approximation is now defined as:

$$\widetilde{X}(\theta) = \sum_{j=0}^{S} c^j \Phi^j(\xi(\theta)) \approx X(\theta) \tag{2.4}$$

The way S is defined depends on:

1. the order P of the Wiener-Askey polynomial chaos
2. the number n of random variables $\xi_k(\theta)$

S (ref. [46]) is equal to:

$$S = \frac{(n+P)!}{n!\ P!} \tag{2.5}$$

So, the computational cost of gPCEs method strongly depends on n and P which both affects the time required to generate the complete orthogonal polynomial chaos basis and the number of simulations required to tune the expansion coefficients. This makes stochastic expansion methods very flexible since there is no theoretical limitation to the number of uncertain variables modeled in the system. The big drawback of introducing many of them, though, is the exponential increase in the computational cost required to generate the approximated model $\widetilde{X}(\theta)$.

The great advantage of the stochastic expansion methods with respect to the sampling methods lays in how the system response stochastic content is computed. $\widetilde{X}(\theta)$ is a so called *surrogate model* of the real $X(\theta)$. Such term refers to a function that represents the behaviour of a system whose formulation, though, provides easy access to information about the real model which were not otherwise available. The definition of a surrogate model using gPCE exposes the statistical content of the approximated system response. It is possible to prove, in fact, that the output stochastic content can be derived analytically from the gPCE coefficients c_j. The formulation is the following:

$$E[X] = \mu_X = c^0 \tag{2.6}$$

$$\sigma^2[X] = \sigma_X^2 = \sum_{j=1}^{S} (c^j)^2 \tag{2.7}$$

Where μ_X and σ_X^2 are the mean and variance of the system response. It can also be proved that even higher order output statistical moments can be computed analytically from the expansion coefficients. Despite having no

particular physical interest, they can be used within stochastic expansion methods to evince analytically properties as the Skeweness and the Kurtosis of the output probability distribution (ref. [42]) which, respectively, represents the asymmetry (to the left or to the right) and the shape of the given distribution.

An available *surrogate* representation generated with PCE provides another important advantage as well. Since the approximated model mimics the input-output relationship of the original system, its polynomial shape makes it much more easy to handle. For instance, running a MC simulation on the approximated model requires much shorter amounts of time with respect to the same analysis run on the real model. This can be very convenient since not all the interesting statistical indices can be computed directly from PCE coefficients, but they require successive sampling methods assessments which, under this approximations, are extremely inexpensive.

For this reason, the use of gPCE allows a very efficient computation of the Sobol indices (ref. [40]). They are used in variance-based sensitivity analysis since they provide an esteem of how much a specific random input is affecting the variance of the entire system response. This can be a very useful tool within parametric investigations of specific phenomena. The indices can be computed directly from the expansion coefficients as well (ref. [6]).

2.3.4 Info-gap Theory

Info-gap Theory was born in the 1980's (ref. [35]) as a non-probabilistic decision theory to support decision making in situations where extensive lack of knowledge is present. This limits the applicability of the method to situations where epistemic uncertainty is present (see section 2.1).

An Info-gap assessment starts with the definition of a model that must be used to enforce a specific decision. Some information, though, is missing from the represented system definition. The purpose of the Info-gap Theory is to define some indices to guide its user to make the "right" choice despite the extensive lack of information.

To do so, uncertain parameters are classified with specific Info-gap models. Such models do not represent the statistical content of unknown parts (called "Info-gaps"). Their purpose is instead to identify how each parameter definition can vary around an "exact" value which is supposed to be the best guess of it. The most simple Info-gap model $\mathbb{U}(h, \overline{u})$ is the fractional error model. According to it, the unknown variable $u(t)$ is so defined:

$$\mathbb{U}(h, \overline{u}) = \{u(t) : |u(t) - \overline{u}| \leq hu(t)\} \tag{2.8}$$

with $h \geq 0$. Where:

- $\overline{u}$ is the best guess available of the uncertain parameter
- $u(t)$ is the actual and unknown value of the parameter evaluated at time t
- h is the *horizon of uncertainty* whose value is assessed as the difference between $u(t)$ and $\overline{u}$ within the given Info-gap model

The way equation 2.8 defines the unknown parameter $u(t)$ can look very similar to the way Interval Analysis does. The variable, after all, is defined in between by two values. This, though, is not true for two reasons.

First, Info-gap models definition is not limited only to the one presented above. The fractional error model is the simplest option, but many other ways to define uncertain parameters exist within the theory (ref. [3]). In case a series of uncertain parameters is present, the entire model uncertainty is defined trough the union of all their Info-Gap models.
Second, Info-gap Theory does not require specific operators in the spreading of the uncertainty through the model. This allows much more flexibility in the entire process.

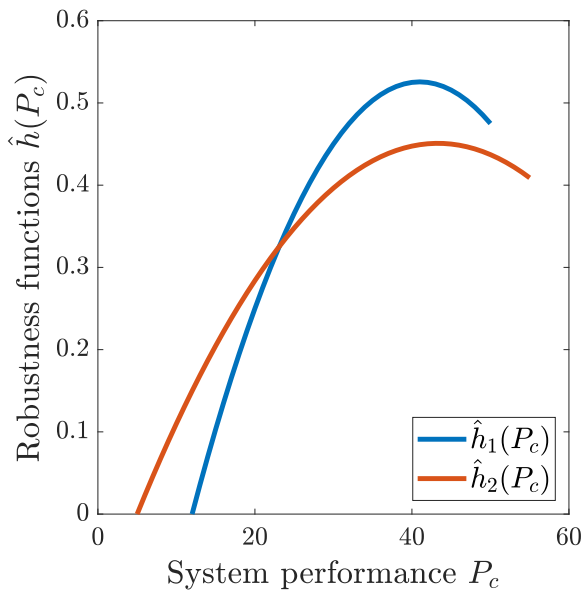

Figure 2.1: Robustness functions plot

Info-gap Theory provides information through the definition of a desired **system performance** P_c. This term is usually represented by a user defined threshold that the system response must never exceed. Imposing that

the uncertain process, whose Info-gaps models are defined, always satisfies the performance requirements, lead to the formulation of the so called **robustness function** $\hat{h}(P_c)$. This function provides the limit value of the horizon of uncertainty h which defines the maximum possible percentage difference between the best guess $\overline{u}$ of the uncertain parameter and the actual parameter value $u(t)$ to still satisfy the desired performance P_c. It basically estimates the system "immunity to failure".

This formulation gives very useful information in case the decision maker has to identify the best available solution to a problem. An example is presented in figure 2.1. In the plot, $\hat{h}_1(P_c)$ and $\hat{h}_2(P_c)$ represent the robustness functions of two different solutions with respect to the system expected performance P_c. The figure provides a visual comparison between the safety margin expected from the two implementations. The most robust solution is identified by setting the desired performance value P_c and finding the curve with the higher $\hat{h}(P_c)$ value.

2.4 UQ methods for multibody software

Section 2.3 introduces some of the different options available to estimate the uncertainty propagation of random variables through defined models.
Here the specific use of UQ methods in multibody software is discussed. The purpose of this analysis is strictly connected to the design of *TiPa*. For this reason, the multibody solver MBDyn working principles are taken as reference in the discussion.

The optimal UQ method must:

- be compatible with MBDyn solver formulation
- allow the simulation of a vast range of uncertain parameter
- grant the propagation of multiple random input variables
- have good convergence properties
- be as time efficient as possible

The investigated options are once again:

1. Sampling methods
2. Interval analysis
3. Stochastic expansion methods

4. Info-gap theory

They are now discussed one by one.

2.4.1 Sampling methods

Sampling methods are introduced in section 2.3.1. There, both the Monte Carlo (MC) and the Latin Hypercube Sampling (LHS) methods are presented. Due to their similarities, the discussion here focuses only on the Monte Carlo method.

Introducing Monte Carlo in MBDyn formulation can be very easy. That is because MC is not actually affecting the way the software works. Once the solver input variables are generated and the outputs of the simulation are extracted, what happens inside the software has no importance. Moreover, the solidity and robustness of the method make it very suitable for its combination with MBDyn. It would allow to propagate virtually any desired number of unknown parameters and to collect the required information about the response behaviour.

As already explained, though, its most important drawback is the strong dependency of the entire analysis results on the dimensions and elements populating the generated sample. For this reason, to avoid these influences, the analysis is run drawing the tested parameter values from a very large sample. *TiPa* may take some minutes to generate the MBDyn model and extract the useful information from the simulations, for this reason the use of MC would inevitably make the UQ assessments very time demanding.

In an attempt to make the UQ assessment as flexible and fast as possible, the MC approach is not considered to be the best option for the specific purpose. That said, due to its robustness, uncertainty propagation analysis results obtained with MC can always be used as reference to assess the precision of other UQ implemented tools.

2.4.2 Interval Analysis

There are many reasons for which this method is problematic to combine with MBDyn.

First, implementing this UQ method within or along an existing multibody software which does not relies on interval arithmetics is very complicated. MBDyn is not designed to handle it. For this reason, due to the generic orientation of the software, the introduction of this formulation would require rewriting almost the entire code.

Second, interval analysis does not gives any statistical information about the system response. This may be acceptable when epistemic uncertainty is hidden in the system model, but it is a big limitation in case the probability distribution of the uncertain parameter is known since such information is not relevant for Interval Analysis. This reduces the flexibility of the uncertainty assessment tool contrasting with the parametric design of *TiPa*.

Third, the formulation of a multibody software with Interval Analysis may be very dangerous since there is nothing to grant the convergence of the method. Multibody formulation is complex. This, as explained in 2.3.2, may lead to the inevitable divergence of the response interval providing no useful information.

Despite all the just mentioned cons, though, a working formulation based on interval uncertainties would be of great interest. That is because sensitivity analysis is possible using intervals. This is done assessing the relationship between the intervals of the input and output variables (ref. [29]). Sensitivity analyses, as already pointed out, are of great interest within this particular research frame.

The non negligible problems carried by implementing this UQ theory in MBDyn, though, are too many to make Interval analysis an eligible solution.

2.4.3 Stochastic Expansion methods

PCE based UQ tools are very fast and versatile. As explained in section 2.3.3, many interesting statistical indices and properties of the response can be assessed easily through these methods.

Among all the presented UQ theories, the combination of the PCE formulation with multibody solvers has been investigated the most. The reasons of this lie in the many ways the two methods are compatible.

Different ways have been proposed to approach the interaction of multibody and PCE. Sandu (ref. [37, 38]) developed a formulation in which PCE is introduced directly in the multibody solver code. The idea, is based on modeling the input uncertain parameter through a PCE expansion. As consequence, all the problem state variables depending on the random parameter are as well expressed as function of PCE coefficients. This transforms the conventional multibody DAEs equations into a new stochastic formulation. The coefficients of the PCE expansions are eventually computed using the orthogonality properties of the polynomial basis. Through this formulation, it is possible to obtain results both in time and frequency domain and, in theory, to model as many uncertain variables as desired as long as they are defined in the form showed in equation 2.4.

The formulation just explained provides huge amount of information about the random multibody problem. Basically Sandu proposes to solve the stochastic multibody DAEs equations. This type of formulation, though, can be extremely complicated to add to an existing solver. The idea is really powerful when used to assess randomness in multibody simple problems where it is possible to directly write and solve the equations in their stochastic version. Implementing this formulation for arbitrary uncertainty in a general purpose solver, instead, would require rewriting the entire code.

With this design, the simulation is executed only once. The number of computations required for the single assessment, though, can be very high according to the number of existing random variables. In addition, solving the actual stochastic multibody equations provides the uncertainty content of all the output state variables. This results in redundant information since usually only a few responses uncertain parameters are relevant to the specific problem.

This method, though, is not the only option to combine multibody formulation and PCE uncertainty quantification. Sandu's approach is a so called "intrusive" method. That is because the PCE formulation is directly introduced in the software formulation. On the contrast, "non-intrusive" methods exist. These ones consider the model as a "black box" and generates an approximation of the system only from the collection of some system input-outputs combinations. Such methods rely on the formulation explained in section 2.3.3. The non-invasivity is an advantage for many reasons.

First, it does not require the modification of the actual solver.
Second, it allows to assess the uncertain output statistics only on the desired family of responses. This on one side increases the number of time the actual solver is executed since the coefficients of the PCE are assessed from the combination of input-responses pairs, but on the other it avoids the computation of the statistical content of the entire multibody state variables.

So, due to the extreme versatility of the PCE method, the non-intrusive version of this UQ tool is considered the most suitable option to be combined with MBDyn. For this reason, this formulation is introduced in *TiPa* design.

2.4.4 Info-gap theory

Info-gap theory has been extensively used in decision taking problems related to engineering. Including this theory in *TiPa* and MBDyn formulation would surely provide useful new information. Some concepts must be explained though.

First, the method was never implemented before into the complex multibody formulation of a general purpose software. The variety of Info-gap models could well represent the different uncertain variables and propagate them through the code formulation, but the results may not converge.

Second, only epistemic uncertainty can be represented. This is very limiting in a general purpose formulation.

Third, the information provided by Info-gap theory cannot be compared to the one derived by other methods since they are mostly limited to enhancing the decision making process.

This makes Info-gap theory interesting for *TiPa* development only if placed alongside a more robust and studied method. For this reason, the theory is not implemented in the first place.

2.5 PCE and MBDyn

The argumentations of section 2.4 led to the conclusion that the non-intrusive polynomial chaos expansions based UQ techniques are optimal to propagate uncertainty within the multibody solver MBDyn. The focus is now switched on the best possible techniques to adopt in the PCE implementation process.
The options are two:

1. Adding the PCE formulation directly the multibody solver
2. Coupling the solver with an external UQ tool

2.5.1 Adding PCE to the solver

The way non-intrusive PCE works is explained briefly in 2.4.3. The big advantage of this version of the PCE formulation is connected to the no longer necessary existing code modification. This technique requires:

1. an orthogonal polynomial basis generator
2. an algorithm for the PCE coefficients computation

Both of them, though, are not so easy to build.
The orthogonal basis depends on many variables including:

- the order of the polynomial chaoses
- the number of random variables
- the probability distribution functions describing each of them

Formulating a general algorithm containing all these elements makes the generation of such basis complicated. For details about the formulation see reference [47].

Assuming such bases are generated an correctly working, though, precisely matching the PCE coefficients to the input-output relationship can be tricky as well. A possible approach to the problem is, for instance, the spectral projections technique (ref. [39]). This methodology exploits the polynomial orthogonality properties using inner products to project the response against each basis function. This requires multi-dimensional integrals computation and, per se, may not represent such a hard obstacle. The hard part would be the required testing and validation phase of the algorithm. Since many variables affect both the UQ formulation and the software MBDyn, obtaining a general purpose working implementation may require a really long time. For this reason, this solution is not an option within *TiPa* development.

2.5.2 Coupling the solver with an external UQ tool

Coupling the existing solver with an external UQ tool is considered here. Since non-intrusive PCE techniques are the selected ones, the job of the external tool would just be related to accomplishing the tasks explained in the previous section. Once again, the existing software (*TiPa* and MBDyn) is not touched in any way, only different collections of random input-output combinations must be evaluated by the external tool.

The software DAKOTA is designed to do so. The tool "wraps around" whatever software the user provides and runs complete non-intrusive PCE based UQ assessments. DAKOTA both generates random inputs matching the user defined uncertain parameters trend and collects the simulation desired outputs. DAKOTA, on addition, features a wide range of Uncertainty Quantification algorithms based on almost all the methodologies described in section 2.3 (excluding Info-Gap Analysis). This means, that other UQ methods can be used as well within the software to verify the accuracy of the PCE formulation in *TiPa* analyses.

DAKOTA provides the complete gPCE based assessment introduced in section 2.3.3. This includes the computation of the Sobol indices as well as the Kurtosis and the Skeweness of the random output probability density function. This makes DAKOTA the perfect candidate for UQ investigation within *TiPa* frame. For further information about the software DAKOTA, see 2.6.

2.6 What is DAKOTA

DAKOTA is an open source software which provides a flexible interface between the user simulation code and a wide range of analysis methods. It can be used in optimization, UQ, reliability and sensitivity analysis problems.

For the specific purpose, DAKOTA allows to introduce uncertainty quantification analysis including basically all the methodologies discussed above without requiring the user to implement it's own code. With more than 20 years of development behind, DAKOTA provides the user well established and working analysis methods.

The software works in two ways: either using its internal solvers or adapting on the user provided ones. The former (called **direct** mode) is limited of course by the finite number of the existing solvers implemented within DAKOTA itself. The latter, instead, allows the combination of the software with whatever external tool the user may want to use. This is called **fork** mode.

In this mode, the software refers to the user-provided solver as a "black box". This means that DAKOTA simply provides to the other software a set of tuned inputs and collects the simulation outputs without taking into account the tool internal structure. This works thanks to an **interface** the user defines which allows DAKOTA to communicate with the existing solver. This happens twice since there are two different information exchanges during the combined analysis: DAKOTA talking to the user's solver and the opposite. The exchange of information takes places thanks to successive writing and reading of short text file.

The process is so defined. DAKOTA basically provides inputs to the black box expecting it to be able to read what it is saying. This short input DAKOTA provides contains the value of the input uncertain parameter to be tested. The solver runs the actual analysis and generate some response parameters. The desired random output parameter is eventually fed to DAKOTA as short text file. So, the correct establishment of the interface requires coding a function capable of both importing information from DAKOTA input text files and generating an output file storing the simulation outputs.

The software provides an extremely well designed User's guide (ref. [1]). For further details about the DAKOTA and its capabilities, please see: `https://dakota.sandia.gov/`.

Chapter 3

Multiblade Coordinates for multibody software

Extracting useful information about a linear time invariant (LTI) mechanichal system is (in most of the cases) straightforward. By rewriting the given equations in matricial form, collecting informations about its dynamical behaviour simply requires to set the proper eigenvalue problem.

The process is more complex when dealing with linear time periodic (LTP) mechanical systems. The equations gain dependency on time due to periodic variating coefficients embedded in the system matrices. The stability of LTP systems requires specific methods to be assessed correclty. The most famous tool in this sense is provided by the Floquet Theory (ref. [10]). In general these processes can be less intuitive and straightforward with respect to the simple istruments used for LTI investigations.

Helicopters rotors are by nature periodic mechanical systems due to their rotating motions. For this reason, their dynamical behaviour must be assessed carefully. The periodicity rises since each blade is usually described individually by the system equations of motion. The definition of a new coordinate set called Multiblade Coordinates (MBC) can, under certain conditions, remove the system periodicity. The resulting model dynamical behaviour can be assessed with the standard procedure for LTI systems.

The MBC transformation has been used in rotorcraft modelling for many decades. In this chapter, the development of an innovative adaptation of the theory is presented. The MBC transfomation is applied to the DAE system of equations describing a multibody helicopter rotor model generated with the general-purpose software MBDyn. The procedure is implemented in *TiPa*.

3.1 The concept

Multiblade coordinates were introduced in 1943 by Coleman [4] and fully developed by Hohenemser and Yin in 1974 [16].
MBC refers to a new set of coordinates designed to describe the motion(s) of the rotor as a whole system instead of representing its individual blade behaviour. The rotor motion is depicted with respect to a fixed reference frame. This description through a "non-rotating" frame simplifies the interaction between the rotor and other helicopter fixed elements dynamics.
Under specific conditions, the MBC transformation removes the periodicity from the rotor equations of motion.

The following two sections are inspired by the book "Rotorcraft Aeromechanics" by W. Johnson. For further details about the MBC formulation please see ref. [19].

3.1.1 Working principle

The new set of coordinates is introduced. Despite many similarities appears between theoretical frame behind MBC transformation and the Fourier Series expansion, they are not the same.

Defining a rotor with N_b equally spaced blades, the azimuth angle ψ_k assessing the position of the k-th blade is:

$$\psi_k = \psi + (k-1)\frac{2\pi}{N_b} \tag{3.1}$$

where ψ is the azimuth angle of the first blade.
Assuming that the k-th blade motion is described by a single and generic degree of freedom q^k, that DOF can be rewritten as a linear combination of the system multiblade coordinates in the following form:

$$q^k = q_0 + \sum_n (q_{nc}\cos n\psi_k + q_{ns}\sin n\psi_k) + q_{N_b/2}(-1)^k \tag{3.2}$$

Where $n = 1 : (N_b - 1)/2$ if N_b is odd and $n = 1 : (N_b - 2)/2$ if N_b is even. The $q_{N_b/2}$ DOF appears only if N_b is even as well. The coordinates q_0, q_{1c} and q_{1s} are referred as collective and cyclic modes and are the most relevant descriptors of the rotor dynamical behaviour. All the other coordinates $q_{N_b/2}$, q_{ns} and q_{ns} with $n > 1$ are the reactionless coordinates.

In general, it is possible to describe the change of coordinates of the equations of motions through the multiblade transformation matrix $\mathbf{T}(\psi)$ in this form:

$$\boldsymbol{q} = \mathbf{T}(\psi)\boldsymbol{q}_{mb} \tag{3.3}$$

where $\boldsymbol{q} = \{q^1\ q^2 \dots q^k\}^T$ is the vector storing the individual blades DOFs and $\boldsymbol{q}_{mb}$ collects the non-rotating frame MBC degrees of freedom. The transformation matrix $\mathbf{T}(\psi)$ changes according to the number of blades and is defined from the system of equations 3.2.

3.1.2 New point of view

The introduction of the new set of coordinates changes the way the rotor is described. As index of the rotor behaviour, here the eigenvalues and eigenvectors of a simple rotor model are used.

Here, we consider that each blade is described by the flapping in hover equation. Assuming the rotor has N_b blades, the motion of the k-th one, in case some arbitrary damping is introduced in the system, is described by the equation:

$$\ddot{\beta}^k + \frac{\gamma}{8}\dot{\beta}^k + \nu^2\beta^k = 0 \tag{3.4}$$

where ν is the system natural frequency. The forcing terms are neglected since not relevant.
The eigenvalues of equation 3.4 are:

$$\lambda_r = -\frac{\gamma}{16} \pm i\sqrt{\nu^2 - \left(\frac{\gamma}{16}\right)^2} \tag{3.5}$$

Once the transformation is applied, the system described with respect to the non-rotating frame maintains the same number of coordinates but introduces new DOFs. In the new frame, the system of equations 3.4 takes the form:

$$\ddot{\beta}_0 + \frac{\gamma}{8}\dot{\beta}_0 + \nu^2\beta_0 = 0 \tag{3.6}$$

$$\begin{Bmatrix} \ddot{\beta}_{nc} \\ \ddot{\beta}_{ns} \end{Bmatrix} + \begin{bmatrix} \frac{\gamma}{8} & 2n \\ -2n & \frac{\gamma}{8} \end{bmatrix} \begin{Bmatrix} \dot{\beta}_{nc} \\ \dot{\beta}_{ns} \end{Bmatrix} + \begin{bmatrix} \nu^2 - n^2 & n\frac{\gamma}{8} \\ -n\frac{\gamma}{8} & \nu^2 - n^2 \end{bmatrix} \begin{Bmatrix} \beta_{nc} \\ \beta_{ns} \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \tag{3.7}$$

$$\ddot{\beta}_{N_b/2} + \frac{\gamma}{8}\dot{\beta}_{N_b/2} + \nu^2\beta_{N_b/2} = 0 \tag{3.8}$$

The eigenvalues of equations 3.6 and 3.8 match those coming from equation 3.4 since only different coordinates appear in the formulation.

In equations 3.7 some extra coupling terms appear. The eigenvalues of that system of equations are:

$$\lambda_{nr} = -\frac{\gamma}{16} \pm i\sqrt{\nu^2 - \left(\frac{\gamma}{16}\right)^2} \pm in = \lambda_r \pm in \tag{3.9}$$

λ_{nr}, the eigenvalues associated to the β_{nc} and β_{ns} degrees of freedom, appears to be similar to the rotating frame ones λ_r. The former are simply shifted by $\pm in$ with respect to the latter.
The eigenvectors associated to these coordinates have $\pi/2$ phase difference since:

$$\beta_{nc}/\beta_{ns} = \pm i \tag{3.10}$$

for the corresponding eigenvalues $\lambda_r \pm in$.

3.2 Multibody formulation

The transformation of a multibody dynamical system degrees of freedom to MBC is presented in this section. The general purpose formulation of MBDyn is used as reference in the explanation process since the software is used in *TiPa* development. Please do note that the procedure presented here can be adapted and extended to other multibody solvers.

The actual transformation of multibody DOFs to MBC is presented in section 3.2.5. All the other parts introduce the concepts required for the correct process design.

3.2.1 Multibody dynamics basics

Many applications of multibody dynamics rely on the writing of the assembly sub components equations of motion independently from the definition of their interactions. The latter are assessed through specific constraint equations. This means that each component is defined in a "free-free" condition while the assembly constraints are enforced through a set of algebraic relationships (Lagrange multipliers).

In MBDyn, the resulting DAE system is written in a precise order. The entire assembly "free-free" equations of motions are always placed before the constraints enforcing ones. This directly affects the way MBDyn orders the current simulation DOFs and, consequently, the way it assembles the matrices representing the modeled system.
Naming $\boldsymbol{q}$ the generalized MBDyn **state vector**, its element are always

stored in this specific order:

$$\boldsymbol{q} = \begin{Bmatrix} \boldsymbol{x} \\ \boldsymbol{\lambda} \end{Bmatrix} \tag{3.11}$$

Where $\boldsymbol{x}$ vector contains all structural DOFs while $\boldsymbol{\lambda}$ contain the joints (Lagrange multipliers) DOFs.
The equations describing the generic multibody system in MBDyn have the following structure.

$$\mathbf{E}\ \dot{\boldsymbol{q}} = \mathbf{A}\ \boldsymbol{q} \tag{3.12}$$

3.2.2 Rotating systems

To introduce the multiblade transformation, we assume that equation 3.12 describes the dynamics of a N_b bladed rotor. The blades are equally spaced and their azimuthal position is assessed by equation 3.1.
To understand how to transform the system DOFs to the non-rotating frame, it is first important to correctly assess how the system DOFs are stored in $\boldsymbol{x}$.

Nodes

The vector $\boldsymbol{x}$ stores the description of each node motion $\boldsymbol{x}_n$. This means that it can be decomposed as follows:

$$\boldsymbol{x} = \begin{Bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_n \end{Bmatrix} \tag{3.13}$$

for $n = 1 : N_{tot}$ with N_{tot} equals to the entire assembly nodes number.
The rotor nodes can be classified in two groups:

- **recursive nodes** ($\boldsymbol{x}^r$) are the nodes that generate the rotor blades
- **non-recursive nodes** ($\boldsymbol{x}^0$) are the nodes appearing only once in the assembly

All blades are geometrically identical. This means that each one of them have the same number of nodes and DOFs associated. For this reason, they create a recursive pattern in the model definition and their behaviour can be recombined through the multiblade transformation.

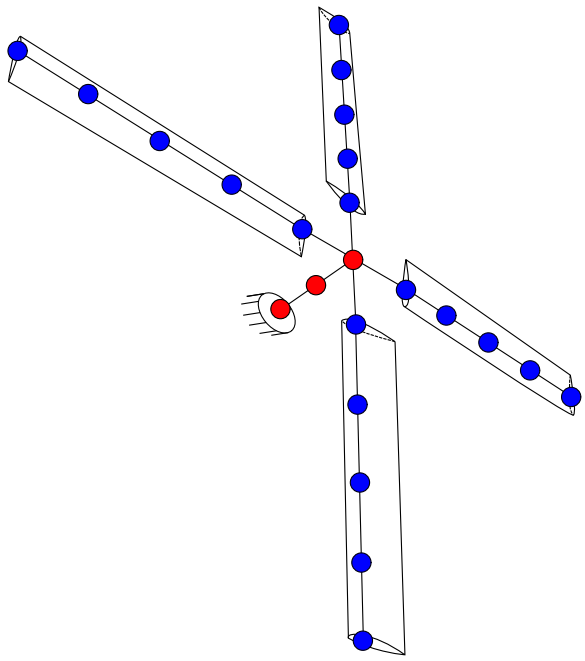

Figure 3.1: Schematic representation of recursive (in blue) and non-recursive (in red) nodes in a four bladed rotor

Non-recursive nodes include for instance the hub node and swashplate nodes. Their associated DOFs do not appear more then once, hence they cannot be rewritten in the non-rotating frame.
A visualization of the two node families is showed in fig. 3.1.

Non-recursive nodes are identified here through apex the $()^0$. The DOFs associated to the i-th non-recursive node are stored in the vector $\boldsymbol{x}_i^0$.
Defining a single blade nodes number as N_{bl}, it is then possible to group each recursive node DOFs in the vector $\boldsymbol{x}_j^k$ where $k = 1 : N_b$ and $j = 1 : N_{bl}$.
The number of non-recursive structural nodes N_{nr} can be computed as:

$$N_{nr} = N_{tot} - \underbrace{N_{bl} * N_b}_{=N_r} = N_{tot} - N_r \tag{3.14}$$

Where N_r is the number of recursive nodes.

These definitions allows to rewrite 3.13 as:

$$\boldsymbol{x} = \begin{Bmatrix} \boldsymbol{x}^0 \\ \boldsymbol{x}^r \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{x}_1^0 \\ \boldsymbol{x}_2^0 \\ \vdots \\ \boldsymbol{x}_{N_{nr}}^0 \\ \boldsymbol{x}_1^1 \\ \boldsymbol{x}_2^1 \\ \vdots \\ \boldsymbol{x}_{N_{bl-1}}^{N_b} \\ \boldsymbol{x}_{N_{bl}}^{N_b} \end{Bmatrix} \tag{3.15}$$

Please do note that the order in which recursive or non-recursive nodes DOFs are defined does not follow specific rules. In equation 3.15, non-recursive nodes DOFs are defined earlier with respect to the others. This is just a convention used to better introduce the nodes classification.

3.2.3 MBC for multibody DOFs

To explain the way the multiblade transformation works for a large number of DOFs, it is first important to understand how it works in a simple case.

1 DOF example

A three bladed rotor with rigid blade elements is defined and a single DOF s_k is assigned to the k-th one of them (so, $k = 1 : 3$). This means that only 3 DOFs describe the entire rotor motion.
In this simple case, the recursive nodes degrees of freedom $\boldsymbol{x}_j^k$ (with $j = 1$) are scalars since:

$$\boldsymbol{x}_1^k = s_k \tag{3.16}$$

According to equation 3.15 and assuming non-recursive DOFs are not present, the system DOFs $\boldsymbol{x}$ can be written as:

$$\boldsymbol{x} = \begin{Bmatrix} \boldsymbol{x}_1^1 \\ \boldsymbol{x}_1^2 \\ \boldsymbol{x}_1^3 \end{Bmatrix} = \begin{Bmatrix} s_1 \\ s_2 \\ s_3 \end{Bmatrix} = \boldsymbol{s} \tag{3.17}$$

The transformation to MBC is defined according to equation 3.3 as:

$$\boldsymbol{s} = \mathbf{T}(\psi)\boldsymbol{s}_{mb} \tag{3.18}$$

where $()_{mb}$ indicate parameters transformed in the non-rotating frame.

2 DOF example

When a new DOF p_k (with $k = 1 : 3$) is added to each model blade, the transformation to MBC requires more attention. In the new configuration, each blade node structural DOFs are defined as $\boldsymbol{x}_1^k = \{s_k \ p_k\}^T$.
The entire rotor DOFs are then assembled as:

$$\boldsymbol{x} = \begin{Bmatrix} \boldsymbol{x}_1^1 \\ \boldsymbol{x}_1^2 \\ \boldsymbol{x}_1^3 \end{Bmatrix} = \begin{Bmatrix} s_1 \\ p_1 \\ s_2 \\ p_2 \\ s_3 \\ p_3 \end{Bmatrix} \tag{3.19}$$

Where once again non-recursive node are assumed not to be present. MBC transformation is effective only if consistent DOFs are transformed to the non rotating frame independently. This happens if:

$$\boldsymbol{s} = \mathbf{T}(\psi)\boldsymbol{s}_{mb}, \quad \boldsymbol{p} = \mathbf{T}(\psi)\boldsymbol{p}_{mb} \tag{3.20}$$

where $\boldsymbol{p} = \{p_1 \ p_2 \ p_3\}^T$ and $\mathbf{T}(\psi)$ is the transformation matrix defined in equation 3.3.
To correctly apply the transformation, the system coordinates in equation 3.19 needs to be reordered. So, a DOFs permutation matrix $\mathbf{P}^{\boldsymbol{x}}$ is introduced such that:

$$\boldsymbol{x} = \begin{Bmatrix} s_1 \\ p_1 \\ s_2 \\ p_2 \\ s_3 \\ p_3 \end{Bmatrix} = \mathbf{P}^{\boldsymbol{x}} \begin{Bmatrix} s_1 \\ s_2 \\ s_3 \\ p_1 \\ p_2 \\ p_3 \end{Bmatrix} = \mathbf{P}^{\boldsymbol{x}} \begin{Bmatrix} \boldsymbol{s} \\ \boldsymbol{p} \end{Bmatrix} = \mathbf{P}^{\boldsymbol{x}} \boldsymbol{x}_P \tag{3.21}$$

Where $()_P$ indicates the permuted degrees of freedom vector. $\mathbf{P}^{\boldsymbol{x}}$ is a square matrix whose dimensions L_P are exactly equal to the number of recursive nodes DOFs. For this reason, it is very important to correctiily assess how many DOFs are related to each node to effectively implement the multiblade transformation.

In MBDyn, for instance, each $\boldsymbol{x}_j^k$ vector can store different numbers of degrees of freedom according to the way the associated node is defined.

There are two options.
Static nodes have 6 DOFs associated (3 displacements and 3 rotations) while **dynamic** nodes 12 (3 displacements, 3 rotations, 3 linear momenta and 3 momenta moments).

The permutation allows the system transformation multiblade to MBC. The matrix $\mathbf{T}^{\boldsymbol{x}}(\psi)$ is introduced as follows:

$$\boldsymbol{x}_P = \begin{Bmatrix} \boldsymbol{s} \\ \boldsymbol{p} \end{Bmatrix} = \begin{bmatrix} \mathbf{T}(\psi) & \mathbf{0} \\ \mathbf{0} & \mathbf{T}(\psi) \end{bmatrix} \begin{Bmatrix} \boldsymbol{s}_{mb} \\ \boldsymbol{p}_{mb} \end{Bmatrix} = \mathbf{T}^{\boldsymbol{x}}(\psi)\ \boldsymbol{x}_{mb} \tag{3.22}$$

Constraints

The generalized MBDyn state vector defined in equation 3.11 stores information about the system constraint equations. How is the $\boldsymbol{\lambda}$ vector transformed to MBC?
Constraints can be either enforced on recursive or non-recursive nodes. This means that $\boldsymbol{\lambda}$ elements as well can be grouped in recursive and non recursive quantities according equation 3.15.

$$\boldsymbol{\lambda} = \begin{Bmatrix} \boldsymbol{\lambda}^0 \\ \boldsymbol{\lambda}^r \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{\lambda}^0_1 \\ \boldsymbol{\lambda}^0_2 \\ \vdots \\ \boldsymbol{\lambda}^0_{N_{cnr}} \\ \boldsymbol{\lambda}^1_1 \\ \boldsymbol{\lambda}^1_2 \\ \vdots \\ \boldsymbol{\lambda}^{N_c}_{N_{bl}} \end{Bmatrix} \tag{3.23}$$

Where N_c is the number of constraint enforced on each blade and N_{cnr} is the number of non-recursive joints.
To apply the transformation to MBC, the process follows equation 3.21 and 3.22, giving:

$$\boldsymbol{\lambda} = \mathbf{P}^{\boldsymbol{\lambda}} \boldsymbol{\lambda}_P \tag{3.24}$$

$$\boldsymbol{\lambda}_P = \mathbf{T}^{\boldsymbol{\lambda}}(\psi)\ \boldsymbol{\lambda}_{mb} \tag{3.25}$$

Please do note that each recursive joint has specific number of DOFs. This must be carefully considered when generating the permutation ($\mathbf{P}^{\boldsymbol{\lambda}}$) and MBC ($\mathbf{T}^{\boldsymbol{\lambda}}(\psi)$) transformation matrices for the $\boldsymbol{\lambda}$ vector.

3.2.4 Local VS global reference frames

As already mentioned in the previous section, MBC transformation works only in case consistent quantities are combined together. To make sure this is the case, it is important to identify how the given solver generates and describes the model.
MBDyn, for instance, writes the equation of motions of each element with respect to a single global reference frame. This, if not correctly addressed, can be a problem for the transformation.

To explain the concept, a four bladed rotor is presented here as example. The geometry is defined such that the first blade is aligned with the direction of the positive global MBDyn X axis. The specific set-up can be described with a null azimuth angle ($\psi = 0$). Consequently, the k-th blade azimuthal orientation is assessed by:

$$\psi_k = (k-1)\frac{2\pi}{4} \tag{3.26}$$

(where $k = 1 : 4$) which is an adaptation of equation 3.1 to this case. As consequence, the second blade is collinear to the positive direction of the global Y axis. The rotor considered configuration is depicted in fig. 3.2. In red is represented MBDyn global reference frame.

Assuming we want to focus only on each blade first node, the entire system structural DOFs can be written as:

$$\boldsymbol{x} = \left\{ \boldsymbol{x}_1^0 \dots \boldsymbol{x}_{N_{nr}}^0 \quad \boldsymbol{x}_1^1 \dots \boldsymbol{x}_1^2 \dots \boldsymbol{x}_1^3 \dots \boldsymbol{x}_1^4 \dots \right\}^T \tag{3.27}$$

Where $\boldsymbol{x}_1^k$ for $k = 1 : 4$ is the vector storing the k-th blade first node DOFs. According to MBDyn formulation each $\boldsymbol{x}_i^k$ is defined as follows:

$$\boldsymbol{x}_1^k = \left\{ p_{1x}^k \quad p_{1y}^k \quad p_{1z}^k \quad r_{1x}^k \quad r_{1y}^k \dots \right\}^T \tag{3.28}$$

Where p_{1u}^k is k-th blade first node position DOF in the u-th direction (with $u = x, y, z$) and r_{1u}^k is the same node orientation about u-th axis. All the $\boldsymbol{x}_1^k$ vectors are defined in the global MBDyn reference frame.

Four local reference frames are now introduced. Each one of them is placed at a specific blade root position with the X axis aligned with the element longest dimension (positive from root to blade tip) and the Z axis pointing upwards. Using such reference frames, it is possible to introduce $\widetilde{\boldsymbol{x}}_1^k$: a new representation of each blade first node DOFs. Maintaining the

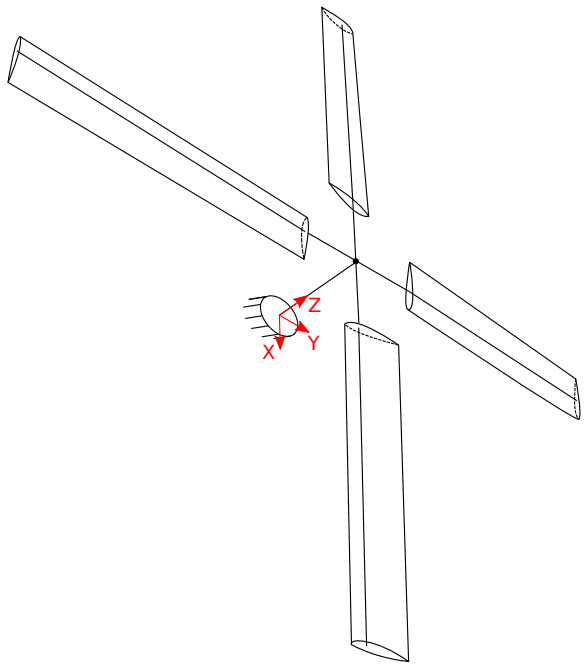

Figure 3.2: Four bladed rotor with $\psi = 0$

same formulation used in 3.28, the vectors are so defined:

$$\widetilde{\boldsymbol{x}}_1^k = \left\{ \widetilde{p}_{1x}^k \quad \widetilde{p}_{1y}^k \quad \widetilde{p}_{1z}^k \quad \widetilde{r}_{1x}^k \quad \widetilde{r}_{1y}^k \ldots \right\}^T \tag{3.29}$$

The information stored by $\widetilde{\boldsymbol{x}}_1^k$ and $\boldsymbol{x}_1^k$ is the same. What differs is the way it is presented and this can be a crucial factor for the correct implementation of the multiblade transformation.

The use of $\boldsymbol{x}_1^k$ is not a reliable method to set the transformation. That is because, going back to the four bladed rotor example introduced above, the first terms of $\boldsymbol{x}_1^1$ and $\boldsymbol{x}_1^2$ (p_{1x}^1 and p_{1x}^2 respectively) carry different information since the problem geometry makes the two DOFs describe different blades local behaviours. This can be proven thanks to the visualization of the local represented blades DOFs. The local reference frames describes each blade motion as if it was recorded by four observers sitting at the elements roots. For the k-th observer, $\widetilde{p}_{1x}^k$ represent by how far the first node is displaced along the k-th blade X axis. So, $\widetilde{p}_{1x}^k$ provides consistent information about the rotor overall behaviour.

What does p_{1x}^k represents instead? How is it compared to the information carried by $\widetilde{p}_{1x}^k$? For this specific rotor geometry and orientation:

$$p_{1x}^1 = \widetilde{p}_{1x}^1, \; p_{1x}^2 = -\widetilde{p}_{1y}^2 \tag{3.30}$$

So, applying the transformation defined in eq. 3.3 to the p_{1x}^k variables does not satisfy the consistency requirements among the transformed quantities. That is because, in this simple case, the operation would combine DOFs assessing different blades local behaviours (see eq. 3.30). The situation can be worse if $\psi \neq 0$ since the information coming from $\boldsymbol{x}_1^k$ is even more shuffled.

As example, the information carried by p_{1x}^k and by $\widetilde{p}_{1x}^k$ are plotted in fig. 3.3. The figure on the left, represents a visualization of the DOFs stored in p_{1x}^k. The global MBDyn reference frame is represented in red. The four blue cirles represents the blade nodes and the arrows represent the direction along which p_{1x}^k DOFs describe the motion. The arrow heads are filled with red since they describe quantities assessed with respect to the global reference frame. It is obvious how such DOFs represent different individual components of the blade behaviour. On the right, instead, the blade local reference frames are depicted in green (actually only two of them are presented to make the picture more readable). The arrow heads are now filled with green to indicate that they are representing the local DOFs described by $\widetilde{p}_{1x}^k$. It is clear from the visualization how the information carried about the blades behaviour is consistent. So, to transform first nodes X position DOFs to the non-rotating frame the use of $\widetilde{p}_{1x}^k$ variables is recommended.

The formulation can be extended to the entire range of DOFs simply suggesting the use of $\widetilde{\boldsymbol{x}}_j^k$ instead of $\boldsymbol{x}_j^k$ as set of recursive nodes coordinates for the MBC transformation.

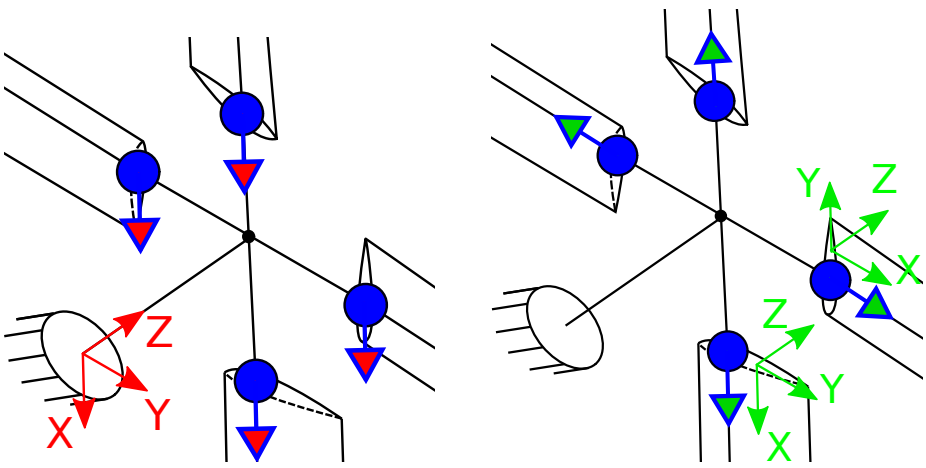

Figure 3.3: Blades node 1 first DOF described in global VS local reference frames

In case the solvers provide equations written in a global reference frame only, the DOFs can be brought to local reference frames thanks to an azimuthal rotation matrix. In the current example, the azimuth position of the k-th blade is assessed by equation 3.26.

To move to the local representation, each blade equations must be rotated with respect to the correct ψ_k angle. The rotation matrix used, depends on the specific axis about which the rotor is spinning. For instance, in case the rotor spins about Z axis (as in fig. 3.2), $\mathbf{R}_z^k$ is used to rotate the k-th blade equations.

$$\mathbf{R}_{\boldsymbol{z}}^{\boldsymbol{k}} = \begin{bmatrix} cos(\psi_k) & -sin(\psi_k) & 0 \\ sin(\psi_k) & cos(\psi_k) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.31}$$

It is important to point out that in some cases, rotations about other axis needs to be accounted as well in this process. If a gimbal hinge is present in the hub, for instance, the rotating disk may be tilted about other axis. In such cases, the use of a complete 3D rotation matrix is suggested.

Some further considerations are required about the definition of local blade coordinated. Bringing multibody equations into new reference frames, in general, requires both the rotation of the structural nodes and of the algebraic constraint equations. This process is though affected by:

- how the specific software writes the assembly "free-free" equations
- how it enforces constraints

MBDyn, as already mentioned, writes the nodes equations with respect to a global reference frame. This means a rotation to local blade frames is needed to obtain a consistent multiblade transformation. This process can have some complications. Due to the way structural nodes equations are written in MBDyn, momentum and momenta moment definition can introduce extra rotations components in case relative motion is present. Allowing some relative motions between different part of the model makes it much harder to impose the correct transformation to local reference frames since extra terms appear in the formulation. To overcome this, the definition a rotor clamped to the ground is recommended. An explanation of this is presented in [25]. If such conditions are satisfied, instead, it is possible to rotate structural DOFs equations just by pre and post-multiplying the azimuthal rotation matrix (see eq. 3.34). Such matrix transformation must act on all DOFs with the correct angular azimuthal orientation.

The entire system structural coordinates defined with respect to MBDyn global reference frame ($\boldsymbol{x}$) can be grouped as:

$$\boldsymbol{x} = \begin{Bmatrix} \boldsymbol{x}^0 \\ \boldsymbol{x}^1_{bl} \\ \boldsymbol{x}^2_{bl} \\ \boldsymbol{x}^3_{bl} \\ \boldsymbol{x}^4_{bl} \end{Bmatrix} \tag{3.32}$$

So, the transformation to the local DOFs is defined as:

$$\begin{Bmatrix} \boldsymbol{x}^0 \\ \boldsymbol{x}^1_{bl} \\ \boldsymbol{x}^2_{bl} \\ \boldsymbol{x}^3_{bl} \\ \boldsymbol{x}^4_{bl} \end{Bmatrix} = \begin{bmatrix} \begin{bmatrix}\mathbf{I}\end{bmatrix} & & & & \\ & \begin{bmatrix}\mathbf{R}_{\boldsymbol{bl}}(\psi_1)\end{bmatrix} & & & \\ & & \begin{bmatrix}\mathbf{R}_{\boldsymbol{bl}}(\psi_2)\end{bmatrix} & & \\ & & & \begin{bmatrix}\mathbf{R}_{\boldsymbol{bl}}(\psi_3)\end{bmatrix} & \\ & & & & \begin{bmatrix}\mathbf{R}_{\boldsymbol{bl}}(\psi_4)\end{bmatrix} \end{bmatrix} \begin{Bmatrix} \boldsymbol{x}^0 \\ \widetilde{\boldsymbol{x}}^1_{bl} \\ \widetilde{\boldsymbol{x}}^2_{bl} \\ \widetilde{\boldsymbol{x}}^3_{bl} \\ \widetilde{\boldsymbol{x}}^4_{bl} \end{Bmatrix} \tag{3.33}$$

where $\boldsymbol{x}^k_{bl}$ and $\widetilde{\boldsymbol{x}}^k_{bl}$ are vectors storing all DOFs related to the k-th blade defined with respect to global and local reference frames. The formulation can be compressed in:

$$\boldsymbol{x} = \mathbf{R}^{\boldsymbol{x}}(\psi)\ \widetilde{\boldsymbol{x}} \tag{3.34}$$

in which $\widetilde{\boldsymbol{x}}$ stores local reference frames DOFs and $\mathbf{R}^{\boldsymbol{x}}(\psi)$ applies the transformation. $\mathbf{R}_{\boldsymbol{bl}}(\psi_k)$ matrices are square matrices whose dimensions are equal to the number of each blade DOFs. In MBDyn, this one depends on the type of structural node implemented in the formulation. The matrices are block diagonal where each one of them is a 3 by 3 rotation matrix ($\mathbf{R}^{\boldsymbol{k}}_{\boldsymbol{z}}$ for instance, see eq. 3.31) whose coefficients depends on the angle (ψ_k) defining the specific blade azimuthal position.
Please note that in equation 3.33 the description of non recursive nodes DOFs $\boldsymbol{x}^0$ is not rewritten in a local reference frame since:

$$\boldsymbol{x}^0 = \mathbf{I}\ \boldsymbol{x}^0 \tag{3.35}$$

This is a consequence that such DOFs are not touched by the MBC transformation and consequently do not need to be described with respect to new reference frames.

The way constraint are enforced in MBDyn really depends on many parameters. In the software, it is possible to implement in different manners the same joints. This is mostly due to the implementation of the "total

joint" in most recent versions of MBDyn. This element allows the definition of various types of joints simply enabling or not specific position and orientation constraints DOFs. The "total joint" is enforced directly in the local reference frames of each node. This means that no extra rotation is required to correctly apply the multiblade transformation. Implementing the same constraint through *ad hoc* MBDyn joints may results instead into partially local or global reference frames joint definitions. This, if not accounted, may induce errors in the transformation. So, for MBDyn users, a good hint would be to use as much as possible "total joint" elements.

From the previous considerations, when MBDyn "total joints" are used:

$$\boldsymbol{\lambda} = \mathbf{R}^{\boldsymbol{\lambda}}(\psi)\ \widetilde{\boldsymbol{\lambda}} = \mathbf{I}\ \widetilde{\boldsymbol{\lambda}} \tag{3.36}$$

Where $\widetilde{\boldsymbol{\lambda}}$ stores the "locally" assessed constraint DOFs.

So, the MBDyn state vector $\boldsymbol{q}$ is described with respect to local blades reference frame through:

$$\boldsymbol{q} = \begin{Bmatrix} \boldsymbol{x} \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{bmatrix} \mathbf{R}^{\boldsymbol{x}}(\psi) & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{Bmatrix} \widetilde{\boldsymbol{x}} \\ \widetilde{\boldsymbol{\lambda}} \end{Bmatrix} = \mathbf{R}_{\boldsymbol{tot}}(\psi)\ \widetilde{\boldsymbol{q}} \tag{3.37}$$

Where $\mathbf{R}_{\boldsymbol{tot}}$ is defined from equations 3.34 and 3.36.

3.2.5 The transformation

In the previous sections, all the steps required to transform MBDyn DOFs to the non-rotating frame have been presented. Here, the entire process is showed.

VWP

To ease the formulation, MBDyn equations are assumed to be derived through a VWP (Virtual Work Principle) formulation. This helps understanding how the MBC transformation is applied to both "sides" of the problem matrices. Thanks to the VWP, MBDyn equations can be written as:

$$\delta \boldsymbol{q}^T (\mathbf{E}\ \dot{\boldsymbol{q}} - \mathbf{A}\ \boldsymbol{q}) = \mathbf{0} \tag{3.38}$$

Since the transformation does not apply to all MBDyn states, the correct identification of the system recursive and non-recursive nodes ($\boldsymbol{x}^r$, $\boldsymbol{x}^0$) and joints DOFs ($\boldsymbol{\lambda}^r$, $\boldsymbol{\lambda}^0$) is assumed.

Global to local

The first step of the transformation requires the blades multibody equations description through their local reference frames. The process is explained and detailed in section 3.2.4.
The different components present in eq. 3.38 are processed differently. The vector $\boldsymbol{q}$ is transformed as in equation 3.37 while:

$$\dot{\boldsymbol{q}} = \mathbf{R}_{\boldsymbol{tot}}(\psi)\ \dot{\widetilde{\boldsymbol{q}}} \tag{3.39}$$

$$\delta \boldsymbol{q}^T = \delta \widetilde{\boldsymbol{q}}^T\ \mathbf{R}_{\boldsymbol{tot}}^T(\psi) \tag{3.40}$$

DOFs permutation

Local recursive degrees of freedom $\widetilde{\boldsymbol{x}}^r$ and $\widetilde{\boldsymbol{\lambda}}^r$ must now be permuted (see sec. 3.2.3) in preparation of the MBC transformation. Equations 3.21 and 3.24 becomes then:

$$\widetilde{\boldsymbol{x}}^r = \mathbf{P}^{\boldsymbol{x}}\ \widetilde{\boldsymbol{x}}_P^r \tag{3.41}$$

$$\widetilde{\boldsymbol{\lambda}}^r = \mathbf{P}^{\boldsymbol{\lambda}}\ \widetilde{\boldsymbol{\lambda}}_P^r \tag{3.42}$$

Since only recursive DOFs are permuted,

$$\widetilde{\boldsymbol{x}} = \begin{Bmatrix} \widetilde{\boldsymbol{x}}^0 \\ \widetilde{\boldsymbol{x}}^r \end{Bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}^{\boldsymbol{x}} \end{bmatrix} \begin{Bmatrix} \widetilde{\boldsymbol{x}}_P^0 \\ \widetilde{\boldsymbol{x}}_P^r \end{Bmatrix} = \mathbf{P}_{tot}^{\boldsymbol{x}}\ \widetilde{\boldsymbol{x}}_P \tag{3.43}$$

and

$$\widetilde{\boldsymbol{\lambda}} = \begin{Bmatrix} \widetilde{\boldsymbol{\lambda}}^0 \\ \widetilde{\boldsymbol{\lambda}}^r \end{Bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}^{\boldsymbol{\lambda}} \end{bmatrix} \begin{Bmatrix} \widetilde{\boldsymbol{\lambda}}_P^0 \\ \widetilde{\boldsymbol{\lambda}}_P^r \end{Bmatrix} = \mathbf{P}_{tot}^{\boldsymbol{\lambda}}\ \widetilde{\boldsymbol{\lambda}}_P \tag{3.44}$$

The global system DOFs permutation is defined by matrix $\mathbf{P}$ as:

$$\widetilde{\boldsymbol{q}} = \begin{Bmatrix} \widetilde{\boldsymbol{x}} \\ \widetilde{\boldsymbol{\lambda}} \end{Bmatrix} = \begin{bmatrix} \mathbf{P}_{tot}^{\boldsymbol{x}} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_{tot}^{\boldsymbol{\lambda}} \end{bmatrix} \begin{Bmatrix} \widetilde{\boldsymbol{x}}_P \\ \widetilde{\boldsymbol{\lambda}}_P \end{Bmatrix} = \mathbf{P}\ \widetilde{\boldsymbol{q}}_P \tag{3.45}$$

Which results in:

$$\dot{\widetilde{\boldsymbol{q}}} = \mathbf{P}\ \dot{\widetilde{\boldsymbol{q}}}_P \tag{3.46}$$

$$\delta \widetilde{\boldsymbol{q}}^T = \delta \widetilde{\boldsymbol{q}}_P^T\ \mathbf{P}^T \tag{3.47}$$

MBC transformation

It is finally possible to apply the transformation to MBC. Once again, only recursive and non-recursive DOFs $\widetilde{\boldsymbol{x}}_P^r$ and $\widetilde{\boldsymbol{\lambda}}_P^r$ are affected by the process. Equations 3.22 and 3.25 becomes:

$$\widetilde{\boldsymbol{x}}_P^r = \mathbf{T}^{\boldsymbol{x}}(\psi)\ \boldsymbol{x}_{mb}^r \tag{3.48}$$

$$\widetilde{\boldsymbol{\lambda}}_P^r = \mathbf{T}^{\boldsymbol{\lambda}}(\psi)\ \boldsymbol{\lambda}_{mb}^r \tag{3.49}$$

The transformation is so decomposed:

$$\widetilde{\boldsymbol{x}}_P = \begin{Bmatrix} \widetilde{\boldsymbol{x}}_P^0 \\ \widetilde{\boldsymbol{x}}_P^r \end{Bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{T}^{\boldsymbol{x}}(\psi) \end{bmatrix} \begin{Bmatrix} \boldsymbol{x}_{mb}^0 \\ \boldsymbol{x}_{mb}^r \end{Bmatrix} = \mathbf{T}_{\boldsymbol{tot}}^{\boldsymbol{x}}(\psi)\ \boldsymbol{x}_{mb} \tag{3.50}$$

and

$$\widetilde{\boldsymbol{\lambda}}_P = \begin{Bmatrix} \widetilde{\boldsymbol{\lambda}}_P^0 \\ \widetilde{\boldsymbol{\lambda}}_P^r \end{Bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{T}^{\boldsymbol{\lambda}}(\psi) \end{bmatrix} \begin{Bmatrix} \boldsymbol{\lambda}_{mb}^0 \\ \boldsymbol{\lambda}_{mb}^r \end{Bmatrix} = \mathbf{T}_{\boldsymbol{tot}}^{\boldsymbol{\lambda}}(\psi)\ \boldsymbol{\lambda}_{mb} \tag{3.51}$$

From which is possible to define the system multiblade transformation matrix as:

$$\widetilde{\boldsymbol{q}}_P = \begin{Bmatrix} \widetilde{\boldsymbol{x}}_P \\ \widetilde{\boldsymbol{\lambda}}_P \end{Bmatrix} = \begin{bmatrix} \mathbf{T}_{\boldsymbol{tot}}^{\boldsymbol{x}}(\psi) & \mathbf{0} \\ \mathbf{0} & \mathbf{T}_{\boldsymbol{tot}}^{\boldsymbol{\lambda}}(\psi) \end{bmatrix} \begin{Bmatrix} \boldsymbol{x}_{mb} \\ \boldsymbol{\lambda}_{mb} \end{Bmatrix} = \mathbf{T}_{\boldsymbol{mb}}(\psi)\ \boldsymbol{q}_{mb} \tag{3.52}$$

Which results in:

$$\dot{\widetilde{\boldsymbol{q}}}_P = \mathbf{T}_{\boldsymbol{mb}}(\psi)\ \dot{\boldsymbol{q}}_{mb} + \dot{\mathbf{T}}_{\boldsymbol{mb}}(\psi)\ \boldsymbol{q}_{mb} \tag{3.53}$$

$$\delta\widetilde{\boldsymbol{q}}_P^T = \delta\boldsymbol{q}_{mb}^T\ \mathbf{T}_{\boldsymbol{mb}}^T(\psi) \tag{3.54}$$

Overall process

To compact the formulation, combining equations 3.37, 3.45, and 3.52 its possible to define the entire process transformation matrix $\mathbf{M}$ as:

$$\boldsymbol{q} = \big(\mathbf{R}_{\boldsymbol{tot}}\ \mathbf{P}\ \mathbf{T}_{\boldsymbol{mb}}\big)\boldsymbol{q}_{mb} = \mathbf{M}\ \boldsymbol{q}_{mb} \tag{3.55}$$

and consequently

$$\delta\boldsymbol{q}^T = \delta\boldsymbol{q}_{mb}^T\ \mathbf{M}^T \tag{3.56}$$

Thanks to to equations 3.39, 3.46 and 3.53, $\dot{\boldsymbol{q}}$ is connected to $\boldsymbol{q}_{mb}$ and $\dot{\boldsymbol{q}}_{mb}$ by:

$$\dot{\boldsymbol{q}} = \mathbf{M}\ \dot{\boldsymbol{q}}_{mb} + \dot{\mathbf{M}}\ \boldsymbol{q}_{mb} \tag{3.57}$$

Where

$$\dot{\mathbf{M}} = \mathbf{R}_{\boldsymbol{tot}}\ \mathbf{P}\ \dot{\mathbf{T}}_{\boldsymbol{mb}} \tag{3.58}$$

MBDyn equations can eventually be rewritten in the non-rotating frame as:

$$\delta \boldsymbol{q}_{mb}^T \Big(\underbrace{\mathbf{M}^T \mathbf{E} \mathbf{M}}_{\mathbf{E}_{mb}}\ \dot{\boldsymbol{q}}_{mb} - \underbrace{(\mathbf{M}^T \mathbf{A} \mathbf{M}\ - \mathbf{M}^T \mathbf{E} \dot{\mathbf{M}})}_{\mathbf{A}_{mb}}\ \boldsymbol{q}_{mb} \Big) = \mathbf{0} \tag{3.59}$$

Which corresponds to:

$$\delta \boldsymbol{q}_{mb}^T \big(\mathbf{E}_{mb}\ \dot{\boldsymbol{q}}_{mb} - \mathbf{A}_{mb}\ \boldsymbol{q}_{mb} \big) = \mathbf{0} \tag{3.60}$$

3.3 Transformation results

As already explained in section 3.1.2, writing the rotor equations in the non-rotating frame changes the way the system is described. This can be easily seen since the new set of coordinates affects extensively the way the system matrices elements are defined and organized.

This section briefly explains how the MBC transformation affects the description of a simple rotor multibody model generated with the general purpose software MBDyn. It focuses first on how the system matrices are reorganized to eventually validate the process with some reference results.

3.3.1 Matrices structure

To easily present the transformation effects, a simple four bladed gimballed rotor model is used here.

The mast and hub structure is modelled with a simple scheme. From bottom to top, the first node is grounded and connected to the lower of the two swashplate nodes. The latter are connected with a dedicated joint. The upper of the two is eventually connected to the hub node. All the just mentioned node and joint elements are this model equivalent of the so called **non-recursive** nodes ($\boldsymbol{x}^0$) and joints ($\boldsymbol{\lambda}^0$) introduced in section 3.2.2.

Each blade element is composed of five nodes connected by two MBDyn beam elements. Their first node is attached to the rotor hub node through a pitch hinge and it is connected to the swashplate through a pitch link.

This simple model, of course, does not represent a realistic rotor system. This scheme is designed to include simple examples of **recursive** nodes ($\boldsymbol{x}^r$) and joints ($\boldsymbol{\lambda}^r$) as, respectively, blades nodes and pitch hinges and links.

Please note that, to obtain a valid transformation, the system is rotating. No aerodynamical elements are used in the formulation.

Figure 3.4 compare the elements populating the simple rotor associated MBDyn matrices before (in red) and after (in blue) the application of the MBC transformation. The visualization are generated with the MATLAB "spy" command. On the left, the comparison between $\mathbf{A}_{mb}$ and $\mathbf{A}$ is presented while, on the right, $\mathbf{E}_{mb}$ and $\mathbf{E}$ are overlapped.

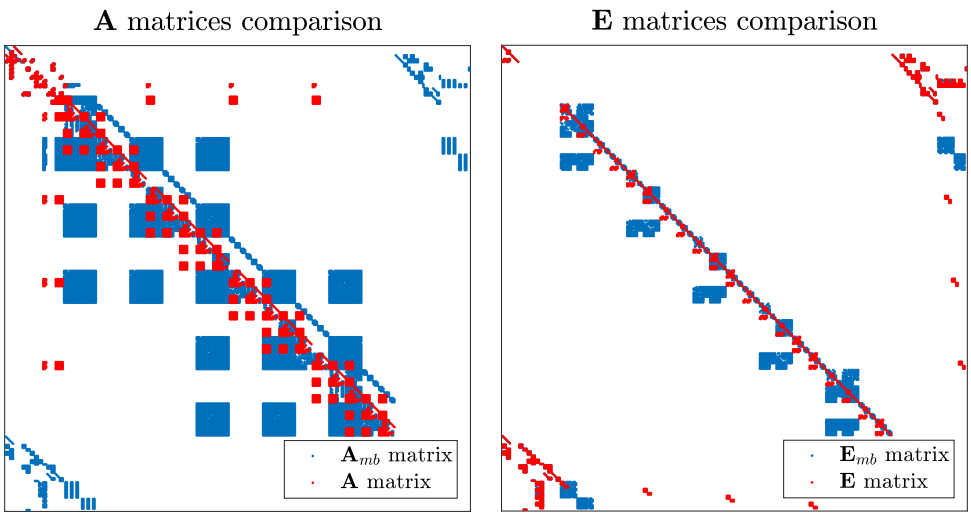

Figure 3.4: $\mathbf{A}_{mb}/\mathbf{A}$ and $\mathbf{E}_{mb}/\mathbf{E}$ comparisons

It is easy to visualize that the transformation drastically manipulates the matrices structure. The actual modifications are broken down one by one in following lines.

Figure 5.3 shows a side to side comparison of $\mathbf{E}_{mb}$ and $\mathbf{E}$ matrices. A grid is introduced to ease the comparison between different sections of the matrices. The boxes named 1.1 and 2.1 refers, in both matrices, to the terms storing information about the non-recursive nodes and joints respectively. The 1.2 and 2.2 rectangles, instead, contain the nodal and joint properties associated to recursive nodes. The terms not included in the grid, which appear on the bottom of each figure, store the information about the joints enforcements but with a slightly different description compared to the one found in boxes 2.1 and 2.2. For this reason, the constraining terms are analyzed only in those two.

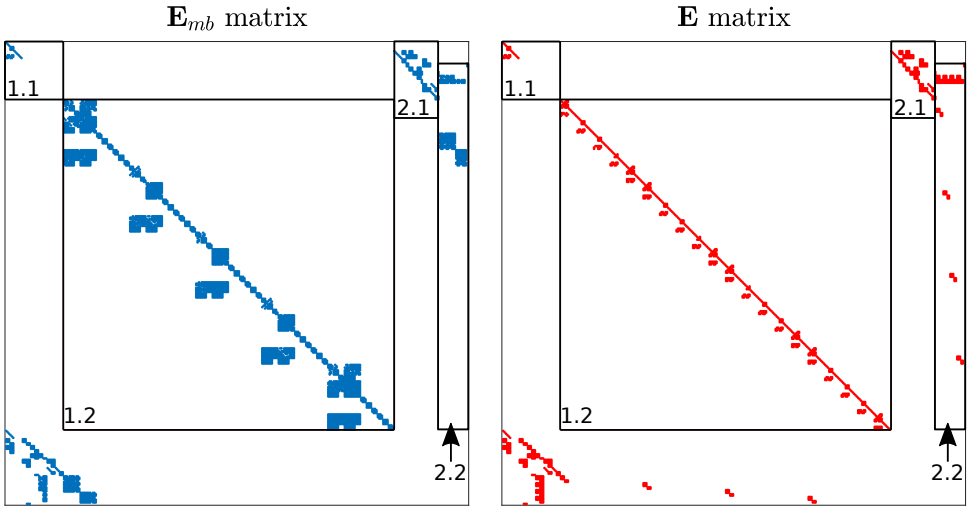

Figure 3.5: Comparison between $\mathbf{E}_{mb}$ and $\mathbf{E}$

The exact method MBDyn uses to write the system equations and consequently the model representing matrices is not explained here. For further information, please refer to [24].

From Figure 5.3 it is clear that, as expected, the quantities related to non-recursive nodes and joints are not affected by the formulation. This is true since the terms stored in boxes 1.1 and 2.1 maintain the same disposition in both $\mathbf{E}_{mb}$ and $\mathbf{E}$. Please note that these visual comparisons do not provide any information about the actual terms magnitude. According to the formulation presented in 3.2, though, only the recursive quantities are affected by the MBC transformation. So, matrices elements associated to non-recursive DOFs maintain the same description both in terms of terms placement and magnitude.

The recursive quantities, instead, are effectively manipulated by the process. In matrix $\mathbf{E}$, box 1.2 contains information about the blade nodes properties. Inside such box, the red plot show a recursive pattern which repeats itself four times: once per blade. With a careful look, the shape of the pattern describing a single blade appears, instead, only once in $\mathbf{E}_{mb}$ matrix 1.2 box. Such shape is extended to the entire nodes DOFs area. This provides a simple visual representation of how each individual blade description is reassembled into an overall description of the rotor. The same process is clear when comparing the information coming from the recursive blade joints enforcement. While in $\mathbf{E}$ matrix four red enforcing constraints blocks appear in 2.2, in $\mathbf{E}_{mb}$ matrix a single larger element contains the joints description. This, once again, shows easily how the multiblade transformation groups

each blade recursive information into an overall rotor description.

Figure 3.6, instead, shows a side to side comparison of $\mathbf{A}_{mb}$ and $\mathbf{A}$ matrices.

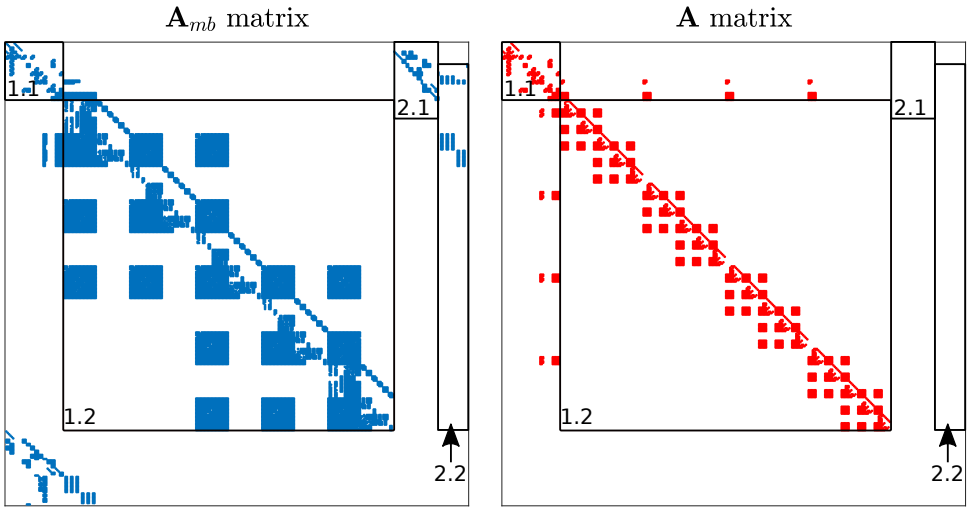

Figure 3.6: Comparison between $\mathbf{A}_{mb}$ and $\mathbf{A}$

The interpretation of these visualization requires a little more attention. The reason behind this is the fact that $\mathbf{A}_{mb}$ contains information coming from both $\mathbf{A}$ and $\mathbf{E}$ matrices as explained in equation 3.59. This is evident when comparing $\mathbf{A}_{mb}$ and $\mathbf{A}$ 2.1 and 2.2 blocks in Figure 3.6. $\mathbf{A}$, according MBDyn formulation, does not contain any information about the joint enforcements. They are present, though, in $\mathbf{A}_{mb}$ due to the $\mathbf{E}$ matrix influence on the matrix. Please note that in area 2.2, the joints are grouped matching the MBC transformation description of the recursive quantities.

Each matrix boxes 1.1 and 1.2 show a similar trend as the one presented in Figure 5.3. This time, though, boxes 1.1 maintain only a similar organization in both $\mathbf{A}$ and $\mathbf{A}_{mb}$ since in the latter the influence of $\mathbf{E}$ matrix corresponding box is present. The red elements present in $\mathbf{A}$ matrix 1.2 block show, once again, the repetition of the same pattern describing each blade elements. This pattern is "expanded" and presented only once in $\mathbf{A}_{mb}$ 1.2 block. Here as well, the influence of $\mathbf{E}$ matrix is visible.

In Figure 5.3 some terms appears in the rectangular boxes placed above and to the left of each matrix 1.2 block. These represent the forces introduced by the pitch links on both the upper swashplate node and each blade first node. In matrix $\mathbf{A}$, these elements repeat themselves with an asymmetrical pattern that is grouped, once again, in matrix $\mathbf{A}_{mb}$ by the MBC DOFs.

3.3.2 New description

To prove the transformation is acting properly on the system, the new system of coordinates must assess the rotor behaviour while respecting the theoretical formulation presented in section 3.1.2. The effects of the transformation can be easily assessed when computing the eigenvalues and eigenvectors of associated to both $\mathbf{A}_{mb}/\mathbf{E}_{mb}$ and $\mathbf{A}/\mathbf{E}$ matrices. They represent the dynamical behaviour of the system both in non-rotating and rotating reference frames.

Since a four bladed gimballed rotor is used here, $n = 1$ from equation 3.2. Consequently, the q^k coordinates describing the k-th blade q-th DOF are recombined through the collective (q_0), the cyclic (q_{1c} and q_{1s}) and the reactionless (q_1) non-rotating frame DOFs.

According to the theoretical development presented in 3.1.2 some system eigenfrequencies remains the same while others are shifter by $\pm 1/rev$ after the MBC transformation. Those remaining constant after the transformation are associated to system modes whose eigenshapes are dominated by the collective and reactionless coordinates. The opposite happens for the eigenvectors associated modes dominated by the cyclic coordinates activation. Table 3.1 shows an example of collective and cyclic dominated modes eigenfrequencies. The chart compares their associated adimensional eigenfrequencies described in the rotating and non-rotating frame with a fixed angular speed.

Mode name	**Rotating frame freq.** [1/rev]	**Non-rotating frame freq.** [1/rev]
Gimbal	1	0
	1	2
Cone	5.39	5.39

Table 3.1: Cyclic and collective modes description examples

The gimbal appears twice since both the regressive and progressive modes derive from the system eigenananlysis. Since these motions are dominated by cyclic coordinates, we can see in table 3.1 how their associated frequencies are shifted by the expected $\pm 1/rev$ amount when described with MBC coordinates.

The cone mode, instead, being dominated by collective coordinates, is represented by the same eigenvalue in both formulations.

Modes dominated by cyclic degrees of freedom also (for instance the first two gimbal modes) are characterized by the theoretical phase lag between

each couple of cyclic coorinates as defined in the theoretical frame. An interesting outcome of the extended MBC multibody formulation is that such behavior is evident in the Lagrange multipliers associated DOFs as well. Still using as reference the four bladed rotor presented above, the concept is now proven.

Among each blade first node DOFs, five of them are locked by the corresponding revolute hinge. Table 3.2 shows the activation of the Lagrange multipliers DOFs associated to the constraints each joint enforces in the Z direction of such nodes. The degrees of freedom presented there are extracted from the eigenvector associated to the progressive gimbal mode. Within each column, the values showed are normalized with respect to the norm of the larger complex number. In the model, the Z axis is placed with respect to the rotor as visualized in Figure 3.2. The k-th blade DOF is called $\lambda^k_{F_z}$ and the MBC corresponding ones are called $\lambda^{F_z}_0$, $\lambda^{F_z}_{1c}$, $\lambda^{F_z}_{1s}$ and $\lambda^{F_z}_1$.

DOFs	**DOFs activation**	**DOFs activation**	**MBC DOFs**
$\lambda^1_{F_z}$	$0.0834 - 0.5032i$	$0 + 0i$	$\lambda^{F_z}_0$
$\lambda^2_{F_z}$	$-0.1813 + 0.9834i$	$0.9951 + 0.0988i$	$\lambda^{F_z}_{1c}$
$\lambda^3_{F_z}$	$-0.0834 + 0.5032i$	$0.0988 - 0.9951i$	$\lambda^{F_z}_{1s}$
$\lambda^4_{F_z}$	$0.1813 - 0.9834i$	$0 + 0i$	$\lambda^{F_z}_1$

Table 3.2: Progressive gimbal mode Lagrange multipliers activation

Table 3.2 provides a simple representation of how the analyzed cyclic mode activates differently the regular MBDyn DOFs and their corresponding MBC version. The regular DOFs show a pattern that describes the periodic oscillation of the four bladed rotor. Please note that, despite the blade elements are modeled without structural damping, the eigenvectors are complex since the system is rotating. This, in MBDyn formulation, introduces some levels of damping due to the presence of Coriolis forces.

The multiblade DOFs behaviour, instead, matches the theoretical analysis presented in section 3.1.2. The collective and hingeless DOFs are not active while the cyclic ones are. Computing:

$$\lambda^{F_z}_{1c}/\lambda^{F_z}_{1s} = i \tag{3.61}$$

Equation 3.61 proves the existence of the expected $\pi/2$ phase distance between $\lambda^{F_z}_{1c}$ and $\lambda^{F_z}_{1s}$ and that the former precedes the latter. Since the rotor is spinning about the positive Z axis, this easily proves the progressive nature of the investigated eigenmode.

Chapter 4

TiPa

TiPa, which stands for *Tiltrotor Parametric model generator*, is a MATLAB environment based software developed at Politecnico di Milano. It represents a key element in the formulation of a tiltrotor generic configuration aeroelastic numerical predictor with uncertainty propagation capabilities. The development of the software started within this thesis project alongside the current whirl flutter research frame.

TiPa is conceived to complete different tasks.
First, it is designed to provide a semi-automatic model generator for an arbitrary user defined tiltrotor wind tunnel configuration. The process is based on three software input cards that can store information about the system features and the required simulation properties. The software is designed to handle different amounts of input data in order to match the specific level of knowledge about the model configuration. Such information is used by *TiPa* to prepare input files for the multibody software MBDyn.
Second, *TiPa* directly provides the abovementioned input files to MBDyn. This happens inside the current *TiPa* simulation in order to automatize the complete aeroelastic assessment.
Third, *TiPa* acts as a posprocessor for the response variables provided by MBDyn analysis.

Such tasks are designed to make *TiPa* simulation flexibles and agile in order to be coupled with the external Uncertainty Quantification software DAKOTA. This cooperation completes the formulation of the aeroelastic stochastic solver.

This chapter starts with a brief introduction to the key ideas behind *TiPa* conception and development. After that, the software simulation capabilities are presented as well as the cooperation between DAKOTA and *TiPa*. In conclusion, the working principles behind the complete aeroelastic solver

with uncertainty propagation capabilities are described.

4.1 Parametric conception

Since the early development stages, *TiPa* has been conceived to model a generic tiltrotor configuration. For this reason, each step of its formulation has been modeled to handle whatever generic data its user may want to provide to simulate a specific wind tunnel system. The software requires the definition of a series of input variables to be manipulated by *TiPa* internal schemes in order to generate the desired geometries. These input parameters are accessible through some input cards (see section 4.2). As consequence, *TiPa* generated models design can be tuned in a really short amount of time just by acting on such variables definitions.

The development of a parametric tiltrotor whirl flutter predictor is meant to satisfy specific investigation requirements. Whirl flutter is an aeroelastic phenomenon affected by a large number of factors. As already explained in section 1.3, the need to understand how each design feature influences a specific system aeroelastic behaviour is crucial in gaining further insights on the origin of instability phenomenon. The easy tunability of *TiPa* models allow this type of investigation.

Today's tiltrotors whirl flutter research is led by TRAST. The project promotes the importance of investigating the behaviour of a generic proprotor configuration to gain better knowledge of the whirl flutter phenomenon (for further details see section 1.3.2). *TiPa* a numerical tool is designed to match TRAST test-bed flexibility to both provide and gain useful information from the research project.

In conclusion, *TiPa* is meant to be a crucial component in the development of a generic tiltrotor wind tunnel model configuration predictor with uncertainty quantification capabilities. This allows the investigation of the system aeroelastic stochastic behaviour with respect to uncertain design features. The "non-intrusive" gPCE based UQ technique has been selected to propagate the random input parameters through the model (see sections 2.3.3 and 2.4.3). The method is introduced in the overall formulation thanks to the software DAKOTA. This does not operate on the internal structure of *TiPa*, but requires easy access to the latter input an output variables (see 2.6). For this reason, *TiPa* simple input parameters system was designed.

4.2 Interfacing with *TiPa*

TiPa does not provide a Graphical User Interface (GUI). The user can interact with the software through specific **input cards**. They are MATLAB files that contain information about both the desired tiltrotor features and the overall simulation. *TiPa* reads such cards in the beginning of each run.

TiPa handles the model generation dividing the tiltrotor entire assembly into two parts (or subsystems): the wing one and the rotor one. This, as explained in section 4.4, provides further flexibility in the modelling process and in *TiPa* analysis selected approach.

The cards definition matches such design scheme. Consequently three different input cards exist.

The **control card** contains a series of variables designed to control the current simulation. For instance they define:

- which assembly component to generate and analyze
- the mode to use in generating each submodel elements (see sec. 4.3)
- the air data for the aeroelastic assessment
- the postprocessing operations *TiPa* must run on MBDyn analysis results

The **wing card** stores the user defined information required to model the wing subsystem. It contains the model geometrical, aerodynamical and structural features. The variables *TiPa* extracts from this card are processed by the software algorithm and transformed into a MBDyn input file.

The **rotor card**, instead, contains information about the rotor geometrical, aerodynamical and structural definition. It includes the blade elements characterization and the description of the rotor/hub command chain. This information as well is processed and transformed to a MBDyn input file.

4.3 The modes

The details of the information stored in both wing and rotor cards can be different. *TiPa* is conceived to work in two modes:

- the *generation* mode
- the *import* mode

The differences between the two formulations justify the different amount of information the user may have to generate each submodel beam elements.

In *generation* mode, the user provided data is as little as possible, hence, the code computes all the required properties to generate the full model. This mode is mostly used to test configurations where the user completely lacks information about inertia and stiffness properties of the elements in the simulation. *TiPa* estimates all the missing properties from the little data provided. For details about this mode see section 4.3.1.

In *import* mode, instead, the user may desire to test components whose characterization comes from an experimental campaign. In this situation, the data available to generate the model is assumed to be more detailed and can be "directly" fed to the software. To read more about this mode see section 4.3.2.

It is important to underline, though, that the every single assembly submodel can be defined either in *generation* or *import* mode without limiting to a specific formulation the definition of the other component.

The following subsections aims at briefly presenting the two different modes functioning. There, only an introduction to the different input variables required by each specific formulation is presented. For further information about *TiPa* internal developlment, the software is provided with a dedicated User's guide. The file can be found in the software dedicated GitLab repository at ref. [14].

4.3.1 The *generation* mode

This mode allows the user to generate each submodel beam elements in case very little details about the components mechanical properties are known.

The software requires to define:

- the type of material used
- the geometrical shape and dimensions of the section
- the mass of the entire beam element
- the X,Y,Z position of each beam element begin/end
- the number of beam elements
- the beam nodes twist angles

Such information is used by *TiPa* to generate each beam elements mass, inertia and stiffness properties to complete the characterization of the com-

ponent. The algorithms *TiPa* uses in this mode are explained in details in the User's guide (ref. [14]).

4.3.2 The *import* mode

In case the user wants to model subcomponents whose beam elements (wing, blade...) mechanical properties have been previously characterized experimentally, much more details about their definition are available. The software allows dealing with this situation thanks to the *import* mode.

This second procedure allows the definition a component by importing the latter section properties (2D) at specific positions along the span. These properties are then expanded to the 3D model through the *import* mode formulation.

The inputs *TiPa* requires are:

- *ETA*: the adimensional position along the component beam axis (X) of the i-th section whose properties are known
- m^i_{2D}: the mass per unit span of the i-th section
- y^i_{CG}, z^i_{CG}: the Y and Z coordinates of the i-th section CG position
- $I^i_{xx\_2D}$, $I^i_{yy\_2D}$,$I^i_{zz\_2D}$: the i-th section moments of inertia
- y^i_{SC}, z^i_{SC}: the Y and Z coordinates of the i-th section shear center (SC) position
- EA^i: the i-th section axial stiffness measure
- EJ^i_y, EJ^i_z: the i-th section bending stiffness about the section Y and Z axis
- GJ^i: the i-th section torsional stiffness measure
- GA^i_y, GA^i_z: the i-th section shear stiffness about the section Y and Z axis
- ξ^i: the *damping factor* of the i-th section
- X^i, Y^i, Z^i: the position in space of each section "center" point
- θ^i: the twist angle of the i-th section

The entire mode formulation is once again presented in ref. [14].

4.4 *TiPa* analysis

TiPa is designed to execute a different range of analysis. It is possible, for instance, to execute analysis on each submodel. This allows the individual characterization of each one of them. The numerical subsystem dynamical behaviour can consequently be tuned to match the real model dynamical behaviour. *TiPa* provides an internal parametric design investigation tool to easily assess how specific parameters influence the mechanical behaviour. A possible outcome of such investigation is showed in Figure 4.1 where, on the left, the system eigenfrequencies are plotted for three different values of a user defined input parameter (called "para"). On the right, instead, a percentage error esteem of the differences between two consecutive simulations assessments is reported.

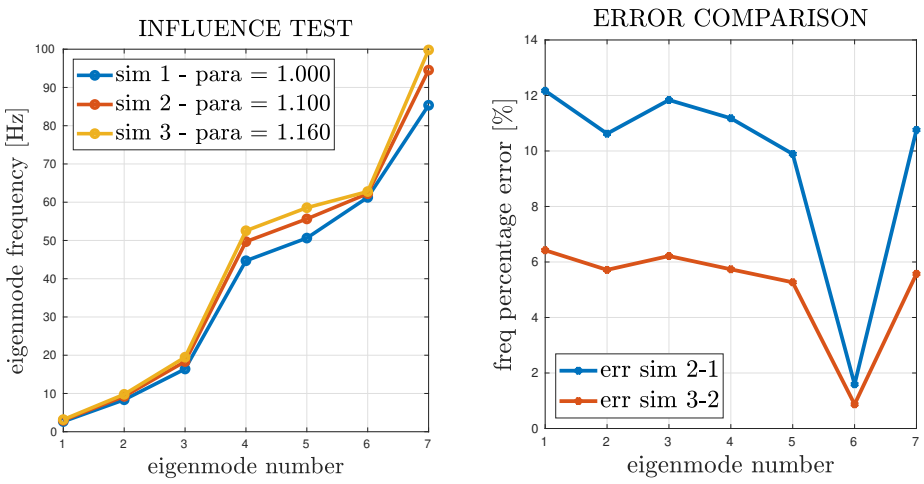

Figure 4.1: *TiPa* parametric design investigation tool outputs

For what concerns complete *TiPa* tiltrotor investigations, two different strategies can be adopted. The basic aeroelastic predictor is based on the definition of the complete tiltrotor MBDyn model starting from the definition of the wing and rotor properties. This approach already provides the possibility to execute complete stochastic parametric aeroelastic assessments. For further information about this process, see section 4.4.3. The use of this modelling approach, though, has some fundamental limitations when complete stochastic aeroelastic assessments are executed on it. This concept is addressed in details in Chapter 5.

The alternative approach, instead, is based on the tiltrotor models development starting from the definition and analysis of the two submodels: one representing the wing element, the other describing the rotor+pylon

subsystem. This second *TiPa* formulation is based on the definition of both the submodels with a node clamped to the ground. They are eventually assembled together to describe the entire proprotor aeroelastic behaviour. This modelling strategy was implemented with some precise reasons behind.

First, dividing the tiltrotor assembly into two separated submodels reduces the number of MBDyn degrees of freedom in each model generation. This, despite the need of two separate assessments, cuts down the each *TiPa* analysis computation time hence speeding up the entire aeroelastic investigation. This happens since in a tiltrotor model, the rotor element plays an important role in defining the convergence of the solution. The imposed angular speed as well as the inputs applied to the control chain require to be introduced progressively in the model motion to be handled properly by the solver. This, inevitably, extend the overall simulation time. If the tiltrotor is divided into two submodels, though, each one of them can be investigated with different simulation times according to their specific requirements. The wing analysis can be much shorter compared to the other system's. As consequence, despite two simulations are executed, with this approach the overall *TiPa* simulation time is cut down due to the reduced number of degrees of freedom.

Second, if rotor is modeled with a node clamped to the ground, it is possible to apply the MBC transformation to such subsystem matrices. The reason behind this requirement is presented in section 3.2 as well as the transformation theoretical formulation. The MBC transformation has the fundamental benefit to provide a better point of view about the rotor aeroelastic behaviour. The transformation can as well be very important in order to obtain as much as possble information about the complete system. This happens since it removes the intrinsic periodicity of this kind of rotating systems providing complete information about the rotor dynamics. Section 4.4.2 presents how the MBC transformation is introduced in *TiPa* workflow.

That said, this formulation was introduced in *TiPa* to improve the overall stochastic predictor investigation capabilities. The use of complete MBDyn models as basis for the sensitivity analysis has some limitations. The main reason behind this is that the use of the conventional modelling approach is strongly dependent on the convergence of each tiltrotor MBDyn analysis. This is not always granted when random input are provided to the stochastic predictor. Executing individual subparts assessments, instead, provides a higher level of controllability over the entire process. The concept is addressed with more details in Chapter 5.

The following sections briefly explains how each tiltrotor component can be individually modelled and analyzed to eventually describes the two dif-

ferent options for complete systems definitions and assessments.

4.4.1 Wing subsystem modelling and analysis

As explained in section 4.2, the wing subsystem definition and analysis is regulated by both wing and control cards. These contain the user defined information necessary to a single wing dynamics or aeroelastic assessment.

A complete wing submodel analysis is now presented. The process is explained graphically in Figure 4.2 on the left flowchart.

Before the assessment begins, the user defines the wing and simulation features through the appropriated cards. Please note that in the figure, the direct interaction between the user and the solver elements is represented with dashed lines. On the other hand, all the other actions are represented by continuous arrows. This convention is maintained along the entire chapter writing.

When *TiPa* analysis starts, the wing and control cards variables are imported by the software to be provided to *TiPa* **preprocessor**. This part of the software has two main purposes:

1. transforming the user defined data into information compatible with MBDyn modelling schemes

2. printing such information into a MBDyn input file

MBDyn is a Command Line Interface (CLI) tool. For this reason, each software simulation is controlled by an input text file. The latter contains the wing model desired geometry (provided by the wing card) and the definition of the simulation analysis including the wind tunnel air properties (provided by the control card). Such information is written according to MBDyn syntax.

The input file is now provided to MBDyn to execute the user defined requested simulation. *TiPa* makes extensive use of MBDyn eigenanalysis since its output files store information about the modelled multibody system equations. The theoretical frame behind MBDyn eigenanalysis is presented in ref. [24]. Starting from such data, thanks to a **postprocessor**, it is possible to reconstruct the MBDyn wing system matrices. These are really important in *TiPa* aeroelastic assessment. Since they allows the computation of the system eigenfrequencies, they are used to investigate the system aeroelastic stability. The postprocessing operations are controlled by the variables stored in the control card.

Once the system matrices and eigenfrequencies are computed, the simulation is complete.

4.4.2 Rotor subsystem modelling and analysis

The rotor subsystem modelling and simulation process is now presented.

The operation main frame is similar to the one presented in the previous section. Once again, the input cards are imported by *TiPa* to be transformed into a MBDyn input file by a preprocessor block. MBDyn simulation is executed and the useful data is extracted by a postprocessor routine. This time, though, the process may not stop here.

With specific control card parameters, the user can enable the **MBC transformer** algorithm. The latter, when turned on, applies to the rotor subsystem matrices the multiblade coordinate transformation (MBC). Once again, the usefulness of such operation is extensively discussed in Chapter 3 as well as the theoretical description of the transformation applied to multibody equations.

As results, the MBDyn rotor subsystem is now described through the non-rotating frames DOFs.

In figure 4.2, a visual representation of the entire process is presented on the right.

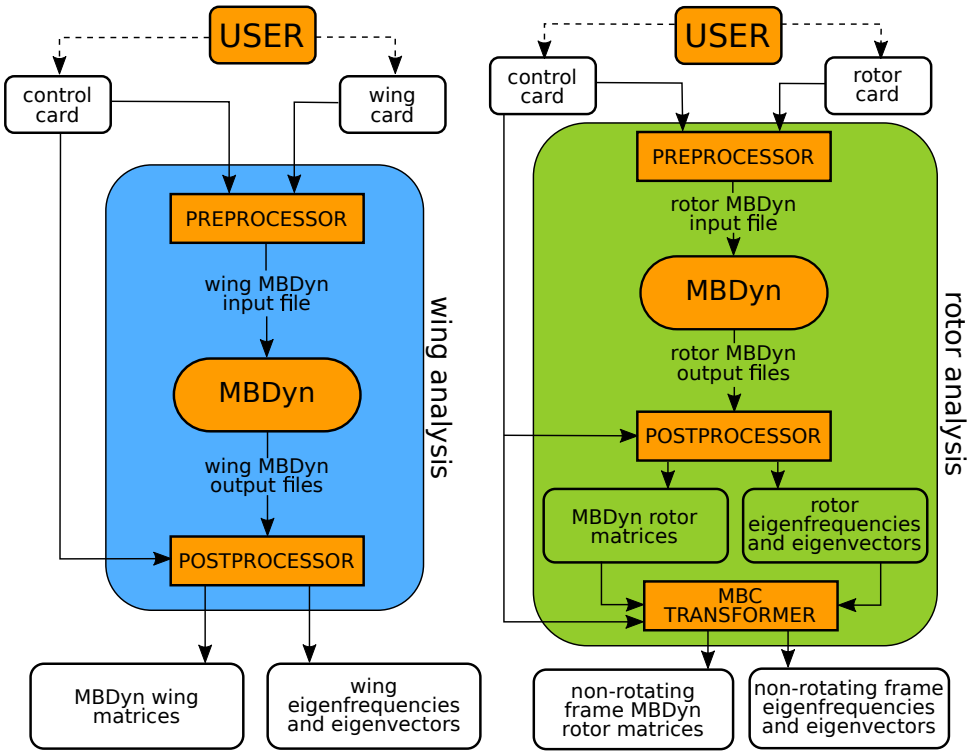

Figure 4.2: *TiPa* wing and rotor submodels analyses comparison

4.4.3 Tiltrotor conventional modelling and analysis

This section presents *TiPa* workflow of the conventional approach for a complete tiltrotor assessment.

Within the procedure, the steps followed are very similar to those executed during an individual wing investigation. This time, though, all the input cards are read by the preprocessor in order to generate a complete tiltrotor MBDyn input file. After the MBDyn analysis is complete, the postprocessor extract the system related matrices as well as the associated eigenfrequencies and eigenvectors to evince information about the system behaviour.

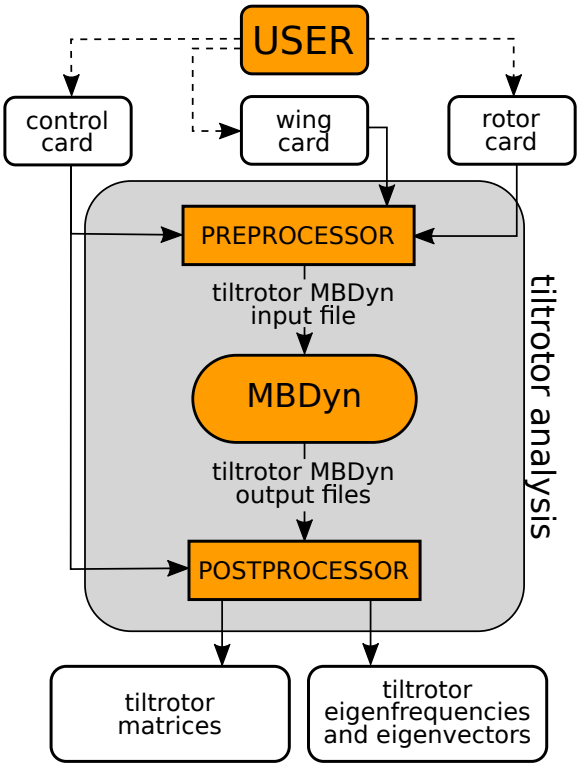

Figure 4.3: *TiPa* conventional tiltrotor model generation and analysis

4.4.4 Tiltrotor alternative modelling and analysis

As mentioned in the beginning of this section, *TiPa* tiltrotor models can be assembled as well starting from the wing and rotor+pylon subsystems. The process relies on a combination of the two models individual dynam-

ical behaviour assessments to evince the entire system representation. A visualization of the formulation used is showed in figure 4.4.

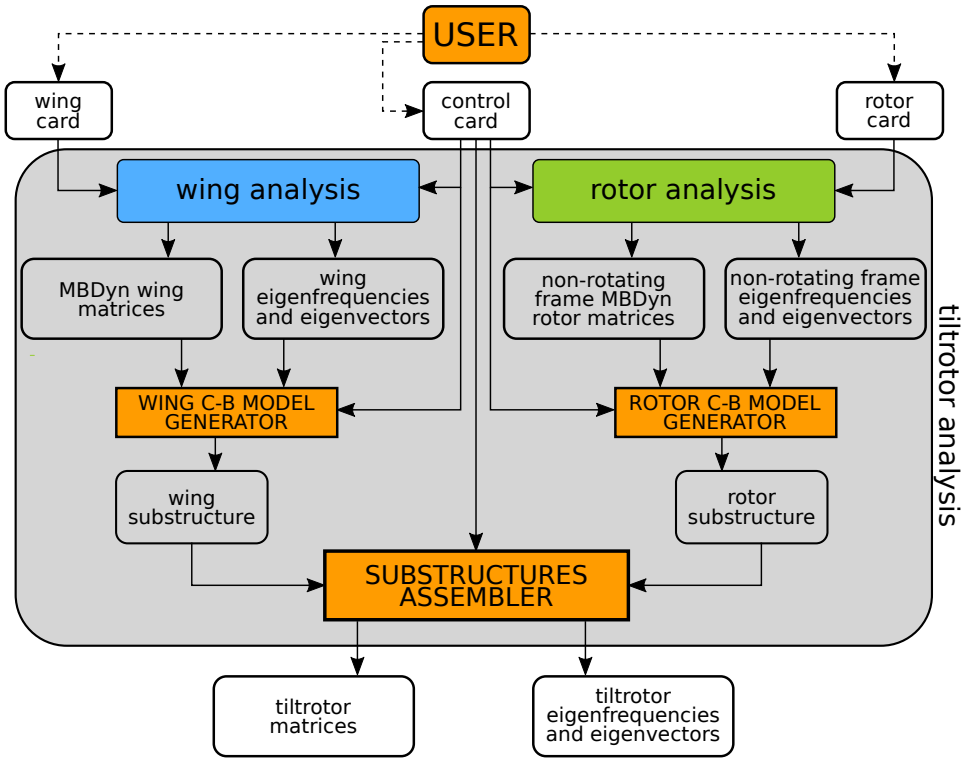

Figure 4.4: *TiPa* alternative tiltrotor model generation and analysis

The process starts as usual with the user definition of the input cards. Once again, all three of them are used.

After that, each single submodel assessment is executed. In figure 4.4, the coloured blocks called **wing analysis** and **rotor analysis** contain the corresponding subsystems simulation processes. What happens inside such blocks matches the formulation showed in figure 4.2 inside the rectangles with matching colours. The wing and rotor analysis outputs are the same compared to those generated in each subsystem individual analysis. Please note that, in this complete model definition, each submodel input card does not affect the analysis of the other one.

The MBDyn wing and rotor models are now combined through a dynamical substructuring process. Within *TiPa*, the technique is based on a generalization of the Craig-Bampton (C-B) approach (ref. [5]). This one is based on the definition and analysis of a complex structure as an assembly of different and simpler substructures. The latter are generated with the separation of the starting model in different regions whose individual description

is generated from a transformation of their systems coordinates using specific matrices. This operation both reduces the number of the system DOFs and provide an intuitive representation of the regions. The procedure differs from the standard modal reduction process since the C-B approach maintains the physical representation of each substructure "boundary nodes" DOFs. These are used to enforce the connections among the individual systems to reproduce the complex original system behaviour.

The generation of the wing and rotor substructures is assigned to two independent algorithms. In figure 4.4, the routines are represented by the blocks called **wing** and **rotor C-B model generator** respectively. The definition of individual algorithms derives from the different approaches necessary to formulate each component substructure. A brief presentation of the implemented techniques is presented in appendix B.

The wing and rotor substructures are eventually assembled by the **substructures assembler** algorithm. This provides the entire tiltrotor system matrices as well as its eigenfrequencies and eigenvectors. Please note that in this formulation the tiltrotor entire system physical degrees of freedom are not direcly accessible since the C-B transformation describes the system mostly trough the subsystems modal coordinates. The system mode shapes, though, can be easily reconstructed using the C-B transformation matrices. All the substructuring algorithms are controlled by *TiPa* control card.

4.4.5 Some concluding remarks

This section contains some concluding remarks about the just explained *TiPa* investigation methods. These considerations are useful to explain the development of the complete solver with uncertainty quantification capabilities.

TiPa investigation process explained in section 4.4 is completely automatic. This means that once the three input cards contain the user defined properties and the software is started, all the tasks presented in the section are executed in sequence by *TiPa*. This continuous workflow is fundamental for the development of the complete aeroelastic predictor with uncertainty quantification.

Both the subsystems and the tiltrotor analyses presented in sections 4.4.1, 4.4.2, 4.4.3 and 4.4.4, though, do non represent complete aeroelastic investigations. In standard *TiPa* analyses, the user can define a single airspeed to test the model with. This, though, does not provide any information about the system behaviour at other speed. So, with this approach, a series of successive analysis is required to complete the assessment.

The problem lies in the fact that it is not possible to access MBDyn aerodynamics formulation outside of the software environment. As consequence, to evince the aeroelastic behaviour of the system for different ranges of speed, MBDyn allows to run a single simulation where the wind airspeed is raised in time and consecutive eigenanalysis are executed. The information evinced from this type of analysis provides a detailed description of the aeroelastic behaviour of the system. Once again, this method is implemented in *TiPa* and can be used as a standard assessment tool, but it is not the one used in the complete stochastic solver. The reason behind this lies in the uncertainty quantification adopted strategy and it is explained in the following section.

4.5 The complete flutter/whirl flutter stochastic investigation

The discussion presented in Chapter 2 proved that the "non-intrusive" gPCE based Uncertainty Quantification method is optimal to be combined with MBDyn multibody dynamics software in order to develop a stochastic general configuration tiltrotor aeroelastic solver. The technique is introduced in the *TiPa*/MBDyn formulation thanks to the external software DAKOTA. In section 2.6, a quick introduction to the tool working principles is presented.

Here, the actual combination of DAKOTA and *TiPa* is presented. In order to understand how the two software are combined to run complete stochastic flutter and whirl flutter investigations, it is first important to show the *TiPa* and DAKOTA basic interaction.

4.5.1 DAKOTA and *TiPa* combination

Since DAKOTA is a Command Line Interface software, the interaction with the tool occurs through the Operative System (OS) terminal. Given that the DAKOTA and *TiPa* combination has been developed on a UNIX-like OS, this chapter explains this interaction using such systems environment commands as reference.

As already mentioned in section 2.6, to allow DAKOTA cooperating with another software an interface has to be established. The latter actual implementation strategy is not reported in this section, for reference see ref. [1]. So, assuming a working interface has already been designed, the basic steps defining a single DAKOTA/*TiPa* analysis are now presented.

The first important element is the way the user can interact with the overall process. The operation is initiated with the execution of DAKOTA

form command line with this syntax.

```
>> dakota -i (input_file_name) -o (output_file_name)
```

Where the ">>" sign is used to indicate command line inputs along the entire chapter writing. The command is straightforward and it is so interpreted. The user is telling DAKOTA which text input file to use to run the analysis and the name of the file to store the process results in. The input file represents the single method the user has to interact with the entire process. This file stores the information about which type of analysis DAKOTA must run and how it is supposed to interact with the user provided software. This means that, during a DAKOTA/*TiPa* coupled analysis, it is not possible to directly access the tiltrotor model definition through *TiPa* input cards. For this reason, both the simulation properties and geometry must be tuned before the assessment starts.

The uncertainty propagation scheme is based on the theoretical frame presented in section 2.3.3 and 2.4.3. The process, in this specific case, requires the communication between two different environments. *TiPa* operates entirely in MATLAB environment. DAKOTA, instead, is assumed for simplicity to be working in the directory where the software is executed. That said, the before-mentioned "interface" refers to whatever exchange of information occur between the two different environments.

The overall DAKOTA/*TiPa* assessment is divided into two phases:

1. the definition of *TiPa* analysis surrogate model

2. the assessment of *TiPa* response stochastic content

The generation of a surrogate model of *TiPa* random input-output relationship has the purpose of providing a process representation much simpler to handle with respect to the original one. If the real system response is called $X(\theta)$, the gPCE formulation provides an approximation of the such output $\widetilde{X}(\theta)$ as an expansion of polynomial basis $\Phi^j(\xi(\theta))$ whose definition depends on the probabilistic definition of the input random variables $\xi(\theta)$. The deterministic coefficients c^j are used to tune the representation. The equation is the following:

$$\widetilde{X}(\theta) = \sum_{j=0}^{S} c^j \Phi^j(\xi(\theta)) \approx X(\theta) \tag{4.1}$$

The detailed theoretical frame behind this entire formulation is explained in section 2.3.3.

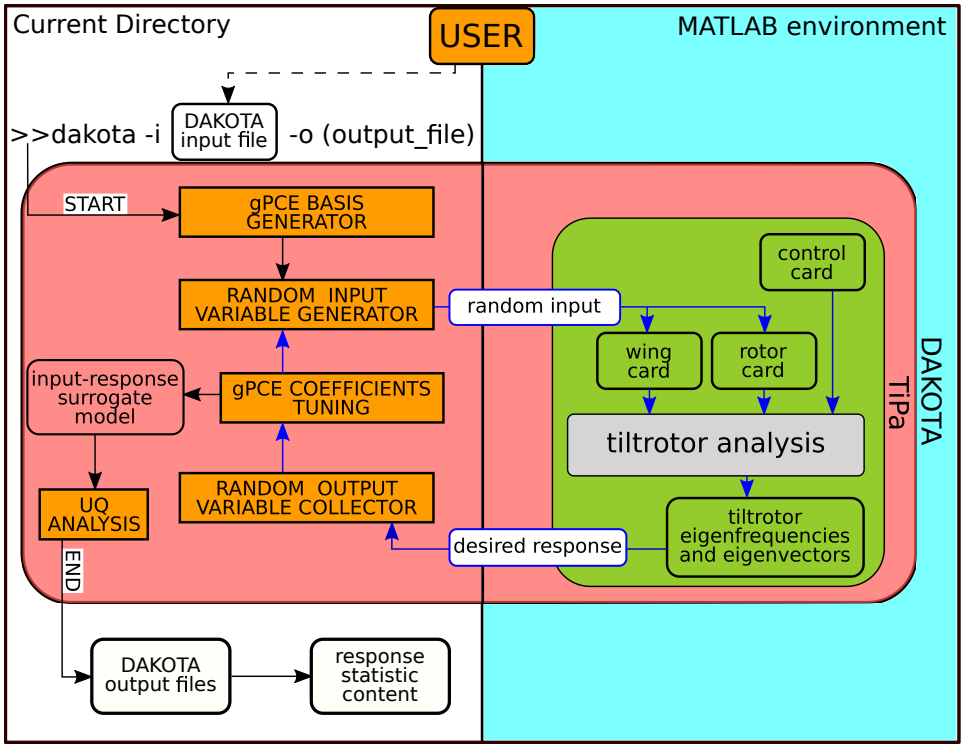

Figure 4.5: Schematization of a DAKOTA/*TiPa* interaction

A schematic representation of the DAKOTA/*TiPa* interaction is presented in Figure 4.5.

The process begins when DAKOTA is started with the command line input presented above in this section. The software, as first step, generates the polynomial basis $\Phi^j(\xi(\theta))$. The task, that can be regulated through DAKOTA input file, is completed by a **gPCE basis generator** algorithm according to the user provided information about the nature of the process random input(s) $\xi(\theta)$.

The coefficients c^j are consequently tuned by DAKOTA with the iterative execution of *TiPa*. The process, that happens entirely inside DAKOTA run, can be easily visualized in figure 4.5. There, DAKOTA environment is represented by the bigger reddish box, while *TiPa* appears on the right side of the flowchart in the green rectangle. The picture is divided vertically in half to represent the two different environments in which the simulation is executed. A DAKOTA **random input variable generator** algorithm evinces a deterministic value of $\xi(\theta)$ according to the user defined input probabilistic distribution. The parameter is consequently provided to *TiPa*. This process represents the first interface interaction between the two soft-

ware. *TiPa* stores the received variable either in the wing or rotor card according to which subsystem it is referred to and execute the complete tiltrotor analysis. The **tiltrotor analysis** block contains one between the two processes presented in Figure 4.3 and Figure 4.4. Please note that since $\xi(\theta)$ affects the tiltrotor geometry or properties definition, each time such variable is provided to *TiPa*, the complete generation of a new model is required.

Once *TiPa* analysis is complete, the second interaction occurs. From the tiltrotor simulation some output variables are provided back to DAKOTA. These parameters represent the j-th loop responses and are collected by a **random output variable collector** algorithm. The responses the user wants DAKOTA to collect are defined through the software input file (see ref. [1]). In this application, they usually store information about the whirl flutter critical modes frequency esteems and damping factors. In case multiple response variables are considered, different response functions $\widetilde{X}(\theta)$ are computed.

The j-th output variables are provided to a **gPCE coefficients tuning** scheme in order to identify the values of the expansion coefficients c^j. The process started with the random input variable generation is repeated iteratively according to DAKOTA internal design as long as all the coefficients are computed. This loop is presented in Figure 4.5 with blue arrows.

Once the tuning is complete, the cycle ends and the surrogate model is entirely defined. This new model is used to compute the system output stochastic behaviour since it represents the relationship between the random input parameters $\xi(\theta)$ and an approximation of the random response $\widetilde{X}(\theta)$ through a polynomial representation. The approximated definition is very convenient because, on one side, the expansion coefficients c^j already provides some statistic information about the real uncertain response $X(\theta)$ while, on the other, the new polynomial description allows the computationally inexpensive application of Monte Carlo-like sampling methods to assess the output probability and cumulative distribution functions as well as the input-output sensitivity indices (see section 2.3.3). This process is represented in Figure 4.5 in the block called **UQ analysis**.

After this operation, the assessment is complete. DAKOTA saves the results in a series of output text files and stops its run. It is important to point out that during the entire process DAKOTA is not aware about the internal structure of *TiPa*. The two interactions happening through the interface between the two software are the only elements necessary to complete the assessment. This is the strong feature of the "non-intrusive" gPCE formulation.

A problem

The process explained in the previous section shows a non-negligible problem. During an UQ assessment, *TiPa* input cards are only accessible by DAKOTA. For this reason, as already mentioned, the user must tune the model design and simulation properties (excluding the random variables provided by DAKOTA) before the UQ software is executed. The problem lies in the fact that the information about the wind tunnel air properties (speed, density,...) are stored in the control card. As results, the user cannot get access to them during DAKOTA execution. This limits each DAKOTA/*TiPa* analysis to a specific airspeed configuration.

A possible solution to the problem is represented by the idea introduced in section 4.4.5. There, the possibility of executing a series of MBDyn eigenanalysis inside each *TiPa* tiltrotor assessment is presented. This, in case the airspeed is gradually increased during the multibody simulation, provides a possible solution of the problem. If the interesting tiltrotor eigenvalues are collected at different speed conditions, a complete picture of the system behaviour is obtained. Since DAKOTA allows the user to define whatever number of response variables desires, all this data can be provided to DAKOTA to obtain a complete stochastic aeroelastic assessment. This approach, though, is not feasible for three reasons.

First, in case the tiltrotor model is generated by *TiPa* with the substructuring process, this introduces some problems. Since the purpose of the investigation is the assessment of the whirl flutter instability, the system is analyzed in a regime horizontal flight condition. The wing and rotor subsystems analysis require different amounts of simulation time to reach such condition. This is mostly due to the more complex nature of the latter with respect to the former one. Since the final purpose is the definition of the complete aeroelastic model, it is crucial to define a routine to extract information about the two systems in the exact same airspeed conditions. This, for the reasons just mentioned, can be extremely complicated end increase exponentially the MBDyn simulation time.

Second, the use of a large number of system responses would increase as well DAKOTA simulation time. This would inevitably affect the overall process efficiency.

Third, DAKOTA output files provides information about the interactions of the random input-output variables during each simulation. This includes the sensitivity evaluation of the random provided parameters influence on the system response. In case a large number of responses is provided to DAKOTA, such evaluation would inevitably lose value. That is true since

the software would cross compare interactions between response variables coming from data obtained at different airspeeds. This, for many reasons, has no physical meaning.

In conclusion, the problem needs a different solution. The selected approach is presented in the following section.

4.5.2 Flutter/whirl flutter investigation

The main purpose of the complete investigation is the definition of the model V-f and V-ξ curves. These functions provide information about the system eigenmodes frequency (f) and damping factor (ξ) with respect to the incoming airspeed values.

To obtain such data, the aeroelastic model needs to be tested at different flight speeds. For this reason, a strategy must be developed to overcome the DAKOTA/*TiPa* limitation to single airspeed per analysis.

The solution implemented uses as fundamental brick the formulation presented in section 4.5.1 and depicted in Figure 4.5. Such workflow is slightly modified in order to obtain a complete whirl flutter assessment. This new version of the DAKOTA/*TiPa* interaction allows the iterative update of the MBDyn simulation airspeed.

The process makes use of a loop stored into a shell script. The procedure is represented in Figure 4.6 where the script is called "DAKOTA_TiPa.sh". There, as in the previous chapter, the ">>" sign is used as convention to precede simple representations of the command line inputs used in the framework. Once again, the process was designed on UNIX-like systems. For this reason, the formulation presents such Operative Systems features.

The operation begins with the user definition of all the N desired airspeed values to test the aeroelastic model with. These variables are stored in the shell script first part. *TiPa* tiltrotor model is still not accessible during the simulation. For this reason, the model features must be tuned before the assessment starts.

DAKOTA is no longer invoked directly. The execution is regulated by the external shell script. The script is initiated with the command:

```
>> ./DAKOTA_TiPa.sh
```

The execution starts with a loop that collects iteratively the i-th airspeed from the user provided values. The variable is printed into a text file called "airspeed.dat" which will be later used by *TiPa*. DAKOTA is now executed with the usual command:

```
>> dakota -i (input_file_name) -o (output_file_name)
```

The DAKOTA/*TiPa* analysis executed in this process is the same one presented in section 4.5.1 and in Figure 4.5 in the block with the reddish colour. Only one detail is different. *TiPa*, before starting its analysis, imports the i-th simulation airspeed data from the text file "airspeed.dat". The process can be easily visualized on the right side of the DAKOTA/*TiPa* block in Figure 4.6. This allows the automatic update of MBDyn simulations airspeed values during external loop progress. Please note that within the i-th loop, the flight speed remains constant. Once the DAKOTA/*TiPa* analysis is complete, the responses stochastic data are stored in the i-th output files.

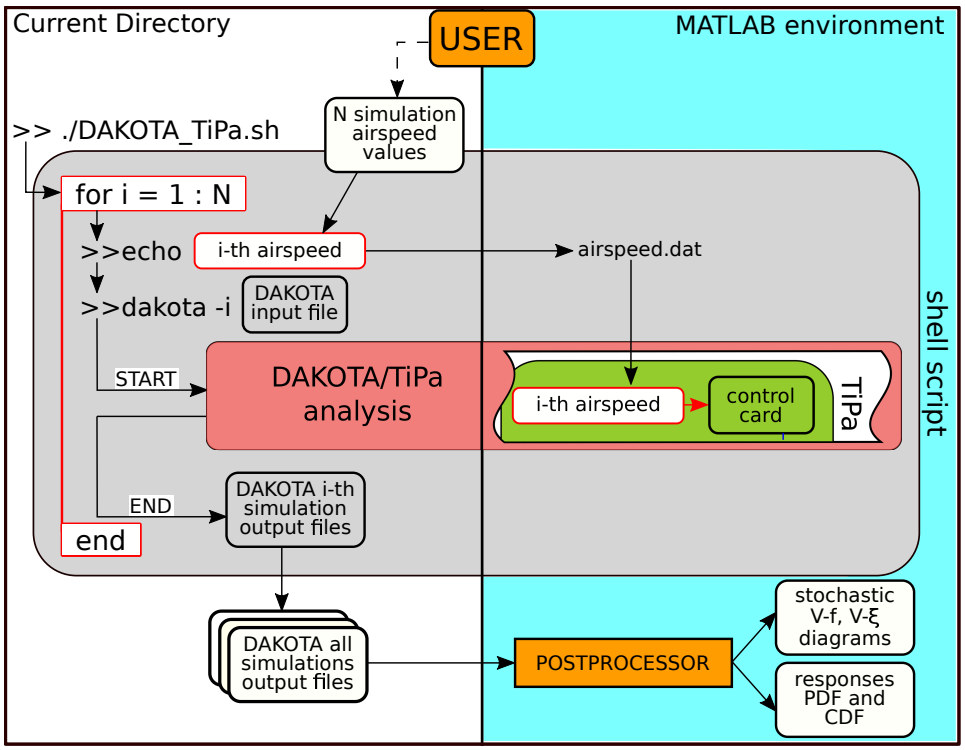

Figure 4.6: Schematization of a complete DAKOTA/*TiPa* aeroelastic assessment

This entire process is repeated N times: once per each defined airspeed. The shell script is over when the loop is completed.

The entire collection of DAKOTA simulations output files is eventually parsed from a MATLAB **postprocessor** which imports the interesting results in the numerical environment.

The typical response variables analyzed are the frequencies (f) and the the damping (ξ) associated to the system modes that may lead to the whirl flutter phenomenon development. In such cases, plotting the i-th loop

DAKOTA responses characterizations over the i-th airspeed values provides a stochastic visualization of the V-f and V-ξ diagrams.

From DAKOTA results files it is possible as well to plot the responses Probability Density Functions (PDF) and the Cumulative Distribution Functions (CDF) curves and to collect the Sobol indices esteem. These provides information about how each input parameter affects a given response function.

Chapter 5

Application

This chapter is conceived to show the most important features of *TiPa* and its combination with DAKOTA. It is divided into two parts.

The first one shows two possible examples of DAKOTA/*TiPa* analysis. There, some considerations about the investigation parameters tuning are presented as well as some examples of the useful indices derived from the analysis. The assessments are executed on a semplification of the three bladed stiff in-plane version of the WRATS model generated entirely with *TiPa*. Please note that such model represents a simpler version of the real WRATS test-bed. For this reason, we do not expect it to match the original system behaviour. The generated numerical model is meant only to provide a working assembly in order to show some typical analysis results. In this part, the tested tiltrotor is generated entirely in MBDyn through the conventional modelling approach (see. section 4.4.3).

The second part, instead, explains some insigths about the alternative modelling approach (see. section 4.4.4). Despite the method shows some interesting and encouraging results, its validation, at this stage of the development, is not complete yet. For this reason, the final part of this chapter explains the state of the art of this different modelling approach. In doing so, it both shows the steps that led to the validation of most of the entire procedure and highlights a possible path to complete the assessment of the process in future research campaigns.

5.1 The model

The tiltrotor model used as reference is the three bladed stiff in-plane version of the WRATS test-bed model. The system is a 1/5 scaled replica of the V-22/JVX aeroelastic model. Further reference on the starting model geometry can be found both in ref. [33] and ref. [26] and in section 1.3.

To obtain the most consistent analysis conditions, the entire range of simulations was executed with the rotor in *windmill* condition: the zero torque trim case. To grant this reference condition, the formulation required the investigation of the collective influence on the rotor torque using the isolated rotor analysis of *TiPa*. These tests led to the introduction of a simple controller capable of maintaining such condition at the different tested airspeeds. The tests were entirely executed with the rotor angular

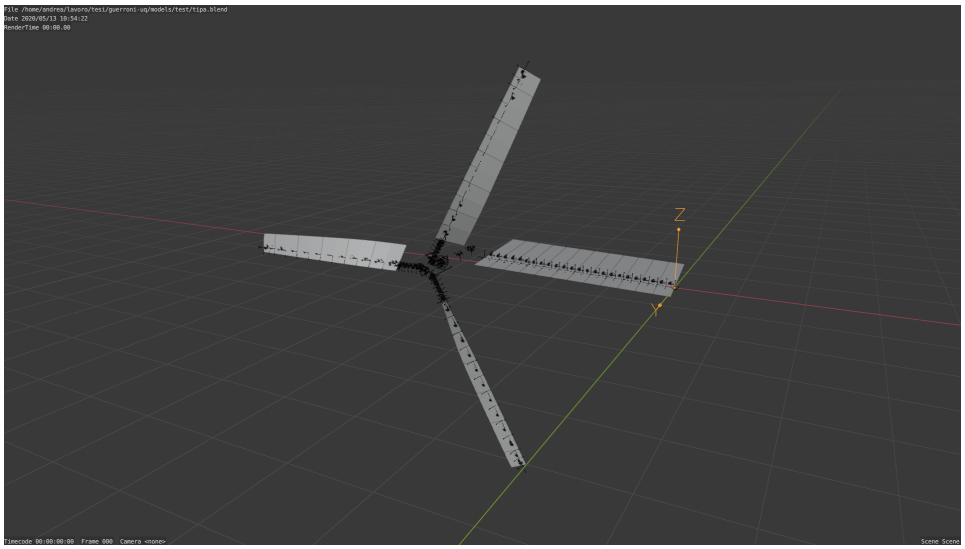

Figure 5.1: The *TiPa* generated WRATS model

speed set to 742 RPM which, according to the data used, represents the reference angular speed in airplane mode for the WRATS test-bed. The model used in the following assessments is presented in Figure 5.1.

5.2 Uncertainty Quantification parameters tuning

The gPCE method relies on the definition of a surrogate model to execute the Uncertainty Quantification assessments (see section 2.3.3). For this reason, the accuracy of the analysis results strongly depend on the quality of the defined approximated model. Its precision is regulated by the order of the

polynomial approximation used to generate it.

For this reason, some investigations were executed to identify the optimal number of polynomial coefficients to use in the definition of each iteration surrogate response models.

To do so, a series of analysis were run to understand the stochastic variation of the WRATS isolated wing model first bending mode frequency under an uncertain mass distribution. This random parameter was chosen since it would have extensively affected the system response value. The tests were executed increasing the order of the Gaussian quadrature rule used to compute the multidimensional integration necessary evaluate the gPCE coefficient values. The number of evaluated gPCE coefficients matches, in this simple case, the quadrature order.

PCE coeff.	**Mean value**	**Variance**	**Time**
1	19.694080904	0.0	25 sec
2	19.712230207	0.48856253450	55 sec
3	19.712295477	0.49074529832	1 min 17 sec
4	19.712295871	0.49076493460	1 min 43 sec
5	19.712305684	0.49078412682	2 min 23 sec

Table 5.1: Effects of gPCE order on reference analysis results

Table 5.1 shows how the gPCE order mostly affects the variance esteem of the random response rather than its mean value. Despite this, the average and variance evaluations converge rapidly as the polynomial order is increased.

It is now important to point out that DAKOTA does not rely on the gPCE coefficients values to compute the local and global sensitivity indices during each analysis. They are evaluated from the sampling methods assessments executed on the surrogate model. For this reason, no extra gPCE coefficients are required to obtain quality sensitivity indices.

So, for the entire sets of analysis presented in the following sections, a fourth order polynomial expansion has been used during the assessments. This was considered to be an optimal solution both in terms of accuracy and in terms of efficiency since, in the following analysis, a limited number of random input and response variables was considered. In case a larger set of parameters are investigated, the use of a lower order polynomial expansion is suggested to maintain a reasonable execution time.

5.3 DAKOTA/*TiPa* analysis results

This section presents some examples of DAKOTA/*TiPa* analysis results. The assessments were executed using the conventional *TiPa* tiltrotor whirl flutter modelling approach. This methodology is based on the parametric generation of the entire assembly within a single MBDyn model. The procedure is explained in section 4.4.3.

5.3.1 Single random input parameter propagation

This section shows the simplest possible version of a DAKOTA/*TiPa* whirl flutter stochastic investigation.

In this example, an uncertain distribution of the wing bending stiffness about the model Y axis (aligned with the incoming wind speed direction) is assumed. To model the random condition, the vector storing the reference wing EJ_y stiffness values is pre-multiplied by a normal distribution $\mathcal{N}(\mu, \sigma^2)$ with:

- mean value $\mu = 1$
- standard deviation $\sigma = 0.05$

Such distribution allows the investigation of the system response when the bending stiffness properties are extracted from a cluster of values close to the nominal ones.

The analysis features and results are now presented.

Analysis time

Table 5.2 shows a detail of the execution times of the different parts of the predictor.

***TiPa* execution**	**DAKOTA execution**	**Total time**
55 min 11 sec	20 sec	58 min 46 sec

Table 5.2: Single random input propagation simulation times

where "DAKOTA execution" refers to the actual time in which the software generates the polynomial basis, tunes the expansion coefficients and executes the UQ analysis. This is pointed out since, as explained in section 4.5.1, most of the entire process actually happens inside DAKOTA environment. Please note that some time seems to be missing in Table 5.2 since the execution

times of both software does not sum up to provide the total analysis time. This is due to the fact that each time a *TiPa* simulation is called, MATLAB is started as well and the latter requires some seconds to become operative at each iteration.

Table 5.2 is extremely powerful in showing the effectiveness of the g-PCE methods in UQ assessments. Here is why.

For each investigated speed, a surrogate model of the random input/output relationship is developed. A single complete *TiPa*/MBDyn tiltrotor assessment requires about two minutes. For this reason, with the used fourth order approximation, four successive runs need to be executed to complete the approximated system representation. Once each surrogate model is ready, DAKOTA, draws 10000 samples from the stochastic input definition and executes a LHS (Latin Hypercube Sampling) UQ assessment (see ref. 2.3.1) on the polynomial representation of the system. This allows the evaluation of the response pdf, CDF and the sensitivity indices of the analysis. This final assessment is extremely fast. By comparison, the same uncertainty assessment could be executed on the original *TiPa* model using the conventional LHS. This, in this case, would require to executes 10000 times a 2 minutes simulation just to obtain the same level of information about the system stochastic response. DAKOTA does this and the gPCE tuning in about 20 seconds.

Of course, there is a price to pay when the gPCE method is used: the responses stochastic content is not evaluated using the real model but by means of an approximation of it. This can affects the results reliability, but it is easy to avoid with the selection of the proper polynomial expansion order.

Stochastic V-f and V-ξ diagrams

The first outcomes of the analysis are the stochastic V-ξ and V-f diagrams. They provide a visual representation of how the system random responses (the beam mode frequencies and damping) are assessed assuming the uncertain input variables. The collection and investigation of the first bending mode properties as response variables is due to the fact that, in the original WRATS model, it is possible to investigate the aeroelastic stability of the system through the definition of this mode stability margin. Figure 5.2 shows the abovementioned diagrams.

In both plots, the red squares are placed at the responses mean values computed at each airspeed condition. The error bars, instead, are a visual representation of the responses standard deviation. Please note that the

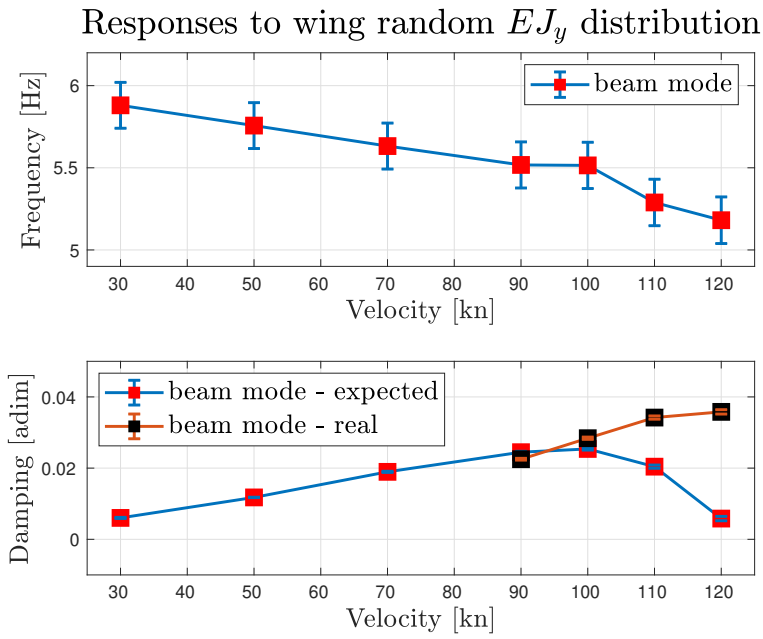

Figure 5.2: Single input stochastic V-ξ and V-f diagrams

minimum tested airspeed has been selected at 30 kn since the simple implemented windmill controller adopted in the model is not effective enough at lower speeds.

Moreover, in Figure 5.2 two different damping trend are presented for speeds higher than 90 kn. The blue line represent the ideal trend we would expect from a perfect definition of the WRATS model. The orange and black one, instead, represents the stability margin derived from its simplification used in the assessments. The real data is not presented for airspeeds lower than 90 kn since the two trends are almost identical. The representation of both curves, though, has the purpose of underlying once again the fact that the analysis presented in this chapter do not try to get any specific conclusion on the real WRATS model design. They are simply presented to show the outcomes of the stochastic predictor developed in this thesis.

In this example, it is clear that the random input parameter is mostly affecting the beam mode eigenfrequency while leaving almost untouched the mode damping quantification. This can be said since the eigenfrequency values shows are more spread around their mean value when compared to the other response ones. The latter parameter actually shows an increasing trend in the variability of the response as the airspeed is increasing. This, despite not being easy to identify through figure 5.2, it can be assessed using

the analysis sensitivity indices.

Local sensitivity

Local sensitivity assessments provide information about how a small perturbation in the input variable affects the system responses. The basic idea behind type of assessment relies on the use of two couples of parameters: a reference input value and its associated system response, and a second input value computed as an increment of the reference one with the related system response. These four variables allows to compute the steepness of the response vs input curve. Such parameter provides the definition of the response variable local sensitivity to the input variation ref. [30].

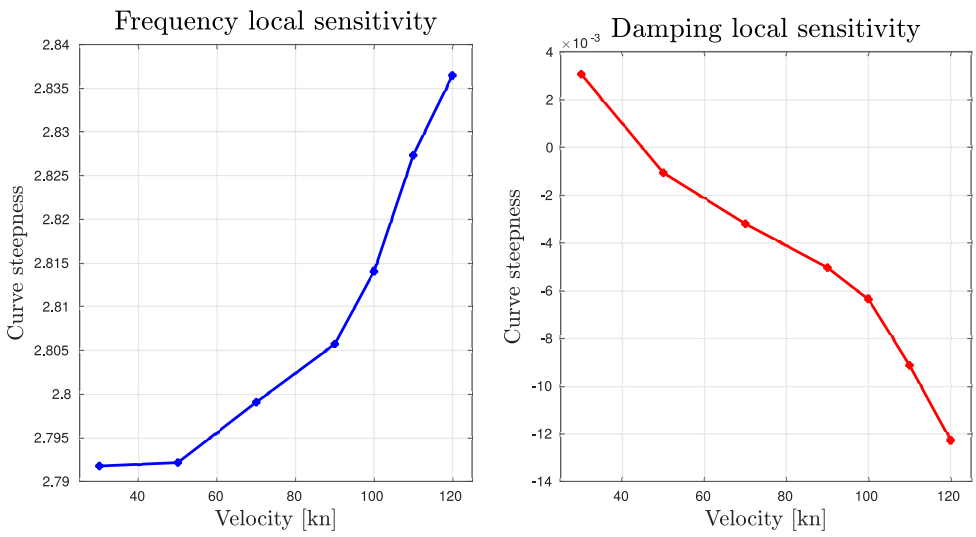

Figure 5.3: Responses local sensitivity to random EJ_y

Figure 5.3 shows the responses local sensitivity indices to the random wing EJ_y distribution. On the left, it is evident that there is an increase in the sensitivity of the beam mode eigenfrequency to the random input value as the airspeed increases. Despite this happens by a very small amount, it means that as the incoming wind speed rises, if the same perturbation in the system input is introduced, the evinced eigenfrequency value derived from the perturbed input tents to increase by a larger amount.

In Figure 5.3 on the right, instead, it is possible to see the beam mode damping associated local sensitivity indices trend. Here, as already clear from Figure 5.2, it is possible to assess how little influence the tested random input parameter has on the mode damping value since the local sensitivity indices have a very low order of magnitude when compared to the other

response indices. The interesting feature of the plot, though, is represented by the change of sign of such sensitivity indices as the airspeed increases. This means that for the same positive increase of the input variable, the beam mode damping value is either increased (at about 30 kn) or decreased (at all the other tested speeds). This provides a very useful insight about the "direction" of the system response at different speeds.

The responses pdf and CDF curves

The last interesting feature provided by a DAKOTA/*TiPa* analysis is the assessment and identification of the responses stochastic curves. An example of such outpus is presented in Figure 5.4 which shows the pdf and CDF curves representing the stochastic characterization of the beam mode eigenfrequency at 30 knots. The pdf curve representation is superimposed with the response mean value (dashed line) and its standard deviation (the two diamonds).

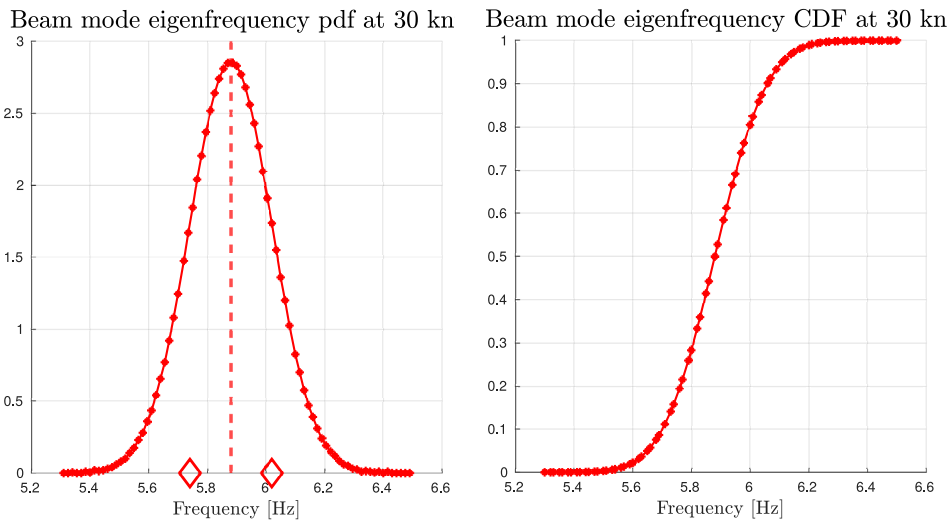

Figure 5.4: Beam mode frequency pdf and CDF curves with random wing EJ_y

This response associated pdf curve maintains the input normal distribution shape with associated:

- Skewness ≈ 0
- Kurtosis $= -2$

For further reference about Skewness and Kurtosis, see section 2.3.3.

5.3.2 Two random input parameters propagation

This section shows some reference outcomes of a DAKOTA/*TiPa* stochastic investigation when two random input parameters are assumed in the system modelling process.

In order to provide a reasonable comparison with the assessment presented in section 5.3.1, the wing beam EJ_y stiffness randomness is used in this investigation as well. The parameter maintains the same stochastic definition of the previous analysis.

As new non-deterministic parameter, a random definition of the pylon mass value M_p is introduced in the formulation. In order to model a possible variability in its definition, its stochastic definition has been generated by pre-multiplying the nominal value of the parameter times a uniform distribution defined in between the values 0.8 and 1.2. This provides a set of input variables placed close to the reference value of M_p. The use of the uniform distribution shape has no particular physical meaning. It was introduced in the analysis in order to provide an example of the possibility to execute analysis with different shapes of the input variables pdf curves.

Analysis time

The DAKOTA/*TiPa* execution time is, in this analysis, extensively affected by the presence of the second random input variable. Table 5.3 shows the details about the simulation times.

***TiPa* execution**	**DAKOTA execution**	**Total time**
142 min 5 sec	52 sec	148 min 58 sec

Table 5.3: Two random inputs propagation simulation times

The analysis time is increased by both the fact that a larger number of *TiPa* executions is required (in order to tune the gPCE coefficients) and by the increased number of random input variables according to equation 2.5 introduced in section 2.3.3.

Stochastic V-f and V-ξ diagrams

The stochastic V-f and V-ξ associated to this assessment are presented in Figure 5.5. Please note that the same concepts discussed in the description of Figure 5.2 beam mode damping trend are valid here. The model used in the analysis does not try to exactly emulate the WRATS behaviour, but it

is meant to show the possible outcomes of some reference DAKOTA/*TiPa* analysis.

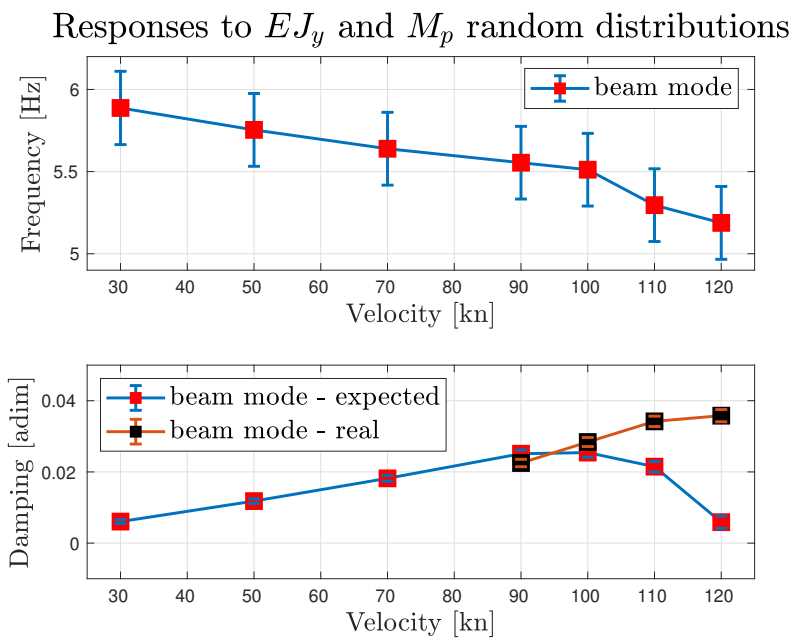

Figure 5.5: Two inputs stochastic V-ξ and V-f diagrams

The curves shows a similar trend compared to those presented in Figure 5.2. When the second random variable is introduced, though, the responses standard deviation tends to increase with respect to the case in which only one appears. It is interesting to notice how, in this second analysis, the beam mode damping standard deviation increment as the airspeed rises is more evident compared to the previous case.

Local sensitivity

Figure 5.6 shows the local sensitivity indices of the system responses to the variability in the M_p value. The represented trends shows that both responses tent to decrease due to an increment in the mass value since the local sensitivity indices always have a negative value. This trend is maintained through all the tested airspeeds.

The comparison between the damping local sensitivity indices represented in Figure 5.3 and Figure 5.6 shows that this parameter is more influenced by an increment in the M_p with respect to a variation of the EJ_y definition. This can be said since the sensitivity indices to the mass pylon

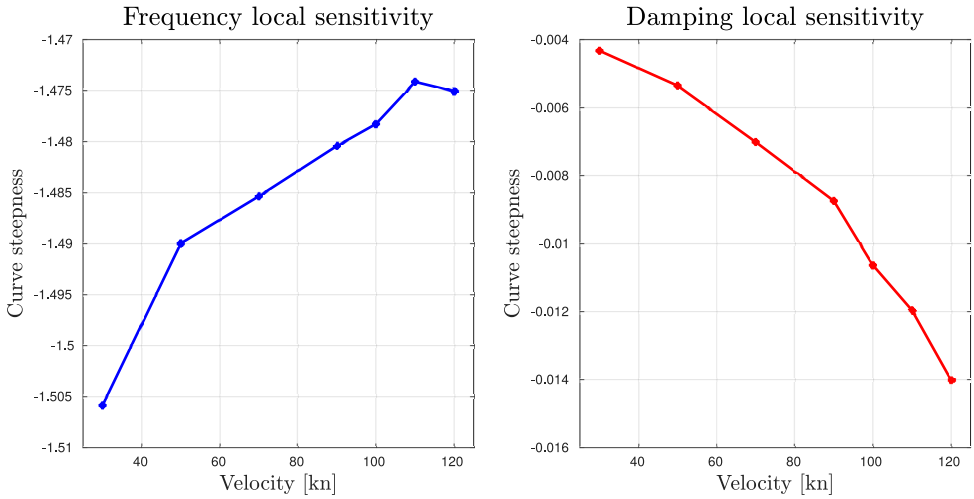

Figure 5.6: Responses local sensitivity to random M_p

value are one order of magnitude larger with respect to those related to the bending stiffness.

Global sensitivity

One of the most interesting aspects of the DAKOTA/*TiPa* investigation when multiple random input parameters are present is the computation of the Sobol indices.

They are used to execute global sensitivity analysis since they provide a quantification of each random input variable contribution to the responses variance (ref [30, 40]). This gives a deep insight about the actual role of each non-deterministic parameter in the system overall behaviour.

Each Sobol index value can be defined in between 0 and 1 and the sum of all the indices associated to a given analysis is always equal to one. In Figure 5.7, two plots are reported. The one on the left shows the values of the Sobol indices indicating the influence of each random parameter input on the variance of the beam mode eigenfrequency at the tested airspeeds. The one on the right, instead, shows the way the damping quantification is affected. Please note that a third index is always estimated when two random variables are introduced in a system. This is meant to assess the contribution of the "interaction" between the two variables to the responses variance. In the presented analysis, very little interaction appears in the results and for this reason it is not reported in Figure 5.7.

These indices can be very helpful in understanding which input parameters are most influential on the system analyzed response. During an aeroe-

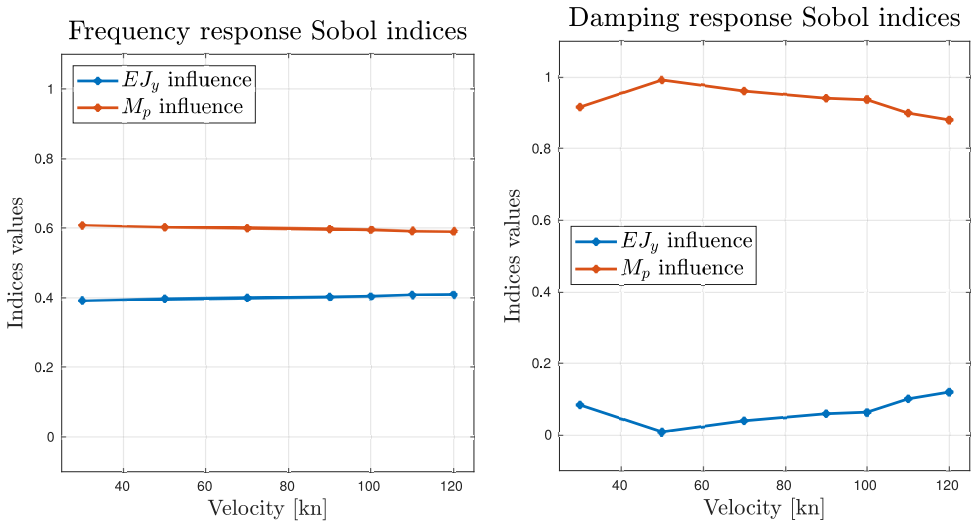

Figure 5.7: Sobol indices associated to the system responses

lastic analysis, for instance, the main purpose could be the investigation of the system parameters whose value affects the most the stability margin. From the right plot in Figure 5.7, it is very clear that randomness of the pylon mass has a larger influence on the beam mode damping variability with respect to the wing stiffness value. This, in this simple test, may happen due to the two different and arbitrary definitions provided to the input variables. In a more realistic case, though, when two (or more) comparable sources of randomness are introduced in the model, the information provided by the Sobol indices can be very powerful and intuitive.

The responses pdf and CDF curves

Figure 5.8 shows the pdf and CDF distributions of the beam mode eigenfrequency evaluated, once again, at 30 kn. By comparison with Figure 5.4 pdf distribution, it is possible to identify how the new parameter inevitably affects the system stochastic response in the same tested airspeed condition.

The change in the pdf properties can be assessed as well in through its shape defining parameters:

- Skewness = 0.04195
- Kurtosis = −1.0461

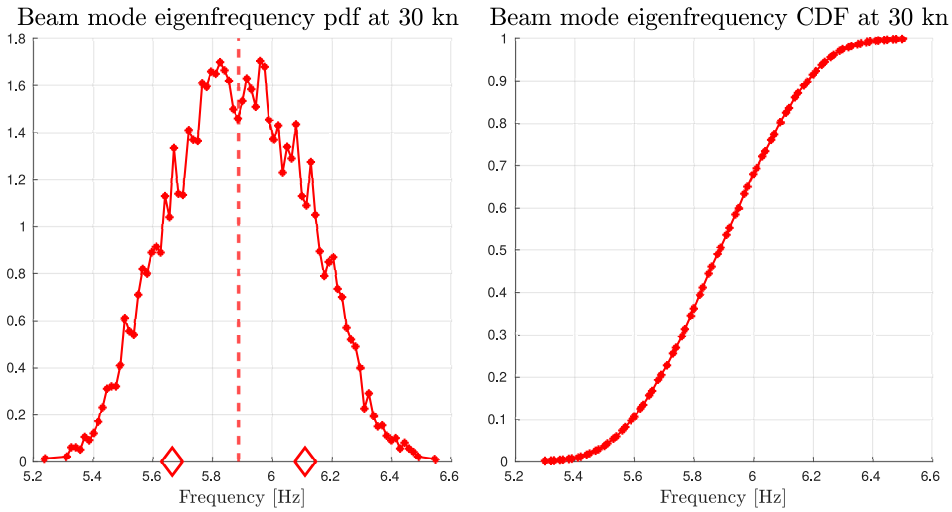

Figure 5.8: Beam mode frequency pdf and CDF curves with random wing EJ_y and M_p

5.3.3 Concluding remarks

Section 5.3 presents some reference analysis obtained from the DAKOTA/*TiPa* collaboration. As already mentioned, such assessments were designed in order to present the most interesting outcomes of this type of analysis. This final section, though, contains some final remarks about the presented assessments.

First, in the reported analysis, the maximum number of random input variables used is two. This does not mean that the stochastic predictor is capable of handling only a maximum of two non-deterministic inputs at a time. The user can choose to introduce in the system as many variables as desired. The use of a large number of variables inevitably affects the speed of the tool as explained in section 2.3.3. Two random inputs were introduced in the second analysis (see section 5.3.2) in order to introduce and present the results of a global sensitivity assessment (which requires multiple stochastic input variables) while maintaining a relatively short simulation time.

Second, the pdf distributions the user can associate to each random input variables are not limited to the normal and uniform ones used in this chapter. Through the gPCE it is possible to model all the distributions which are part of the Wiener-Askey scheme reported in Table 2.1.

Third, the use of the complete tiltrotor model as reference for the investigation analysis is not optimal. This is true since the eigenvalues associated to the system matrices are affected by the periodicity of the rotating system.

As results, the information extracted from those matrices can be slightly different according to the exact condition of the system (the azimuth of the rotor). These effects should not not have a large impact on the resulting assessment since the rotor, when the aircraft is in airplane mode, is in axial flow condition. Despite this, though, with this procedure the effect is not accounted. This is one of the main reason that led to the formalization of the alternative tiltrotor modelling approach based on the multiblade coordinates transformation.

Fourth and last, the use of MBDyn models of the entire assembly to run complete stochastic evaluations is extremely dependent on the convergence of the multibody software. This can lead to the impossibility to correctly identify the flutter condition since the solution could diverge when the analysis gets close to such airspeed. Since the whirl flutter instability is triggered by the interaction of the wing and rotor systems, their individual assessment at the flutter airspeed increases the chances of both simulations to converge. As consequence, a more reliable stochastic assessment of the whirl flutter condition is expected. This is the second important reason for which the alternative approach has been developed.

5.4 The alternative approach

In order to obtain more robust and reliable formulation, the alternative tiltrotor modelling approach has been introduced in *TiPa* design. The technique has the purpose of:

- obtaining the best possible description of the rotor behaviour
- solving the conventional method convergence problem when the whirl flutter condition is approached

A deeper insight on the formulation is presented in section 4.4.4.

The main elements contributing to this innovative formalism have been introduced alongside this work development. Specifically they are:

1. the multiblade transformation for multibody software
2. the substructuring process for MBDyn multibody software

The multiblade transformation applied to multibody systems has been extensively presented in Chapter 3. There, specifically in section 3.3.1 and 3.3.2, the effectiveness of the introduced formulation has been discussed.

The adopted substructuring approach used to define reduced representations of the wing and rotor+pylon subsystems has been introduced briefly

in Appendix B. There, a simple validation example of the wing subsystem substructuring process is presented as well.

This section is designed to present the early results obtained with this new modelling approach. The procedure, though, is not yet reliable enough to provide a correct representation of the modelled system. Here, the state of the art of the procedure is presented alongside to some possible guidelines to complete the development of the formulation in future studies.

5.4.1 Reference set-up

In order to develop a consistent representation of the whirl flutter phenomenon, during each component substructuring process a precise set of constrained normal modes $\mathbf{\Phi}^L$ (see Appendix B) has been used in the construction of the Craig-Bampton matrices of each subsystem. In Table 5.4 are listed the selected mode shapes used in the definition of the wing and rotor substructuring matrix.

Wing normal modes	**Rotor normal modes**
1st bending OoP (Beam)	Gimbal
1st bending InP	Cone
2nd bending OoP	1st collective/cyclic Beam
2nd bending InP	1st collective/cyclic Lead-Lag
1st torsion	...

Table 5.4: Normal modes used in substructuring matrices

Where "OoP" means "Out of Plane" while "InP" means "In Plane". The "..." are placed in Table 5.4 right column to point out that while the normal modes types and numbers used in the beam substructuring process has been maintained as a constant during the whole process since we expect them to be sufficient to model the motion cause by whirl flutter, the use of different shapes associated to the rotor subsystem have been investigated. Different analysis showed that the resulting tiltrotor eigenvalues are affected by the number of rotor normal modes used. Despite this, though, we were not able to define a precise set of rotor modes to use in order to develop a consistent representation of this system to obtain the best possible description of the whirl flutter phenomenon. This topic will require further investigation.

5.4.2 The rotor substructure

The rotor substructuring process used by *TiPa* is briefly explained in Appendix B. There, the basic steps used to define the rotor C-B matrices is

presented. Despite the effectivness of the process can be easily assessed for the wing (proved in Appendix B), this is not true for the rotor. The validation process is more complex because the rotor substructure describes a free-free element since only a single node is connected to the ground in the original constrained model. These degrees of freedom are "freed" by the introduced rigid body motion shapes necessary to generate the system associated C-B matrix. In order to try the effectiveness of the used rigid body motion shapes, we tried to define an equivalent model generated with a different approach.

The definition of a free-free equivalent rotor in MBDyn, though, is not straightforward. For this reason, to test the validity of the transformation, we first compared the MBDyn original clamped rotor with its free-free corresponding substructured model with a very large pylon mass (M_p) attached to its interface node to "reintroduce" the clamp and make the two systems comparable. This was meant to investigate whether the introduced rigid body motions affected in unpredictable ways the system normal modes as well. Table 5.5 shows comparison between the two system normal modes adimensional eigenfrequencies in column two and three. The model were tested in vaquo at 742 RPM, the regime condition. Please, note that both systems are described in multiblade coordinates. This, as explained in section 3.1.2 affects the mode frequency values. For reference, the MBDyn eigenfrequencies computed in the rotating frame are reported in the rightmost column.

Mode name	**MBDyn model** [1/rev]	**Subst. w large M_p** [1/rev]	**MBDyn rotating** [1/rev]
Gimbal	≈ 0	≈ 0	1
	1.98	1.99	1
Cone	1.18	1.18	1.18
1st coll L-L	1.63	1.63	1.63
1st cyc L-L	0.62	0.61	1.62
	2.62	2.61	1.62

Table 5.5: Clamped MBDyn and substructured rotor modes comparison

Table 5.5 shows a clear similarities in the two system representations. This proves that the rigid body shapes generated and introduced in the system through the CB matrix do not alter the rotor flexible behavior.

With the exact value of the pylon mass, the rotor substructured model eigenfrequencies slightly changes with respect to the clamped system ones. The modes shapes, as well, adapt to the free interface condition. As example

of this, in Figure 5.9, the coning mode of the rotor substructure is presented. There, it is easy to identify the displacement of the mast and hub nodes with respect their initial position. This matches the expected behavior of the system with free interface. Among the eigenmodes associated to this substructured system some pure rigid body motions appear as well.

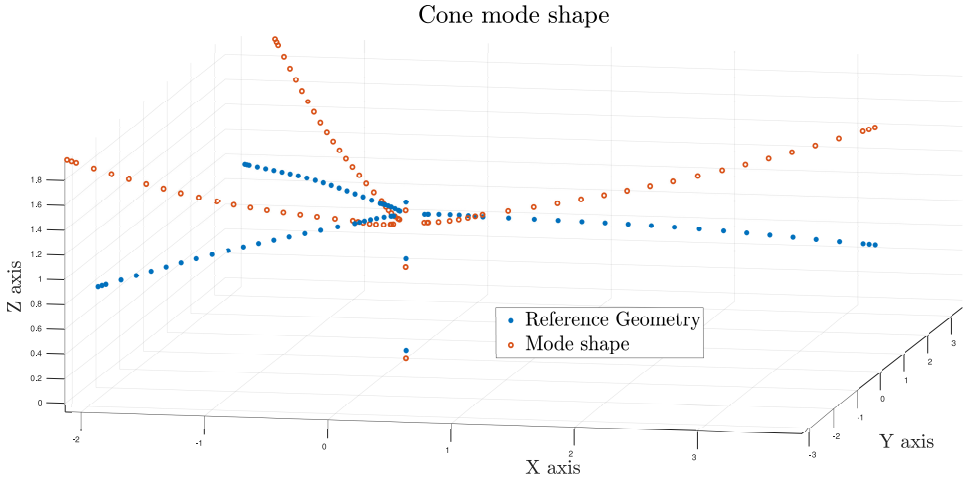

Figure 5.9: Coning mode of the rotor substructure

So, the rotor substructure generated with the current approach, seems to provide a good representation of the subsystem. Despite this, though, the process cannot be entirely validated since we could not generate a different free-free rotor model in order to prove the exact match between the formulations. This step, as well, is left for future investigation.

5.4.3 Tiltrotor model

The fact that we could not validate the abovementioned steps, inevitably reflects on the definition of the tiltrotor model with the adopted strategy.

As results, the dynamical behaviour of the tiltrotor model generated with this approach is not consistent with the expected one. This means that by comparing the tiltrotor models generated with the two different modelling approaches, the resulting descriptions do not match at some level.

The alternative approach proved to be effective in the reconstruction of the proper tiltrotor system mode shapes since they appear to be very similar to those associated to the entire MBDyn tiltrotor model. In figure 5.10, for instance, the beam mode shape of the tiltrotor generated with the C-B approach is represented. The shape was reconstructed starting from the eigenvectors associated to the substructured system matrices.

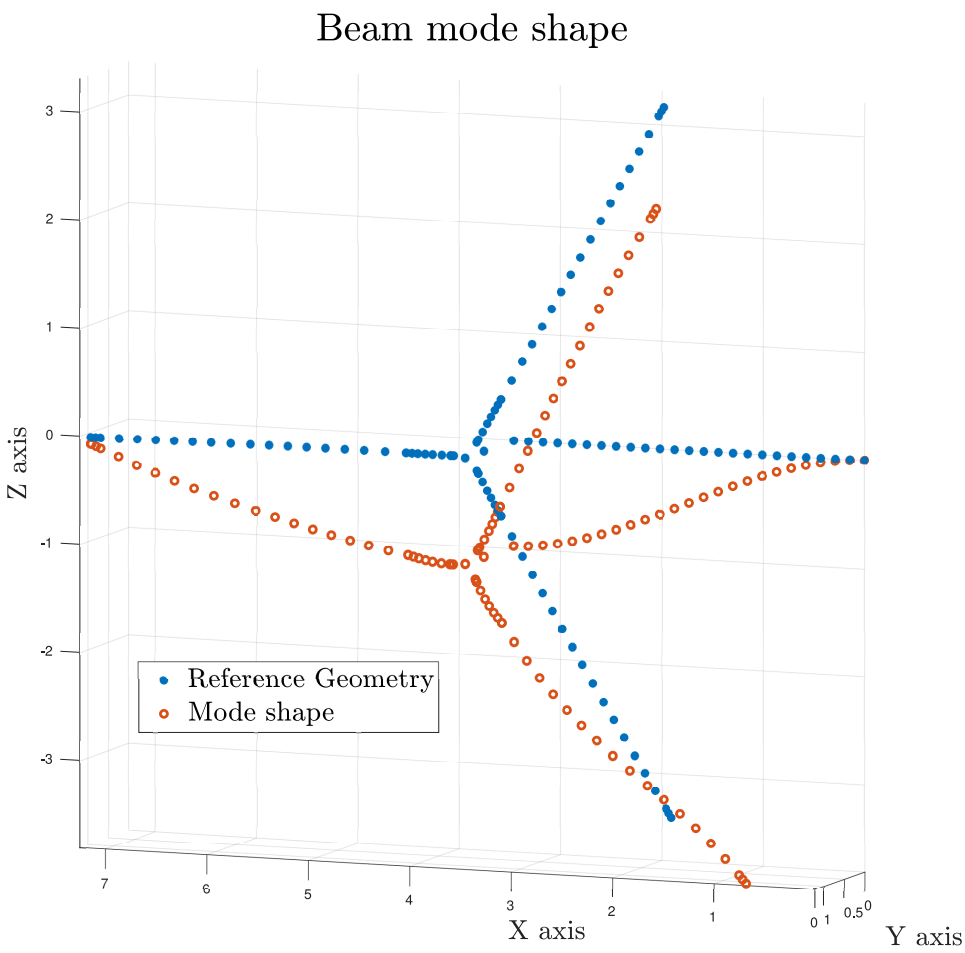

Figure 5.10: Beam mode shape from tiltrotor substructured model

5.4.4 Frequency shift

Despite the system mode shapes proves to be consistent with the expected ones, though, their associated eigenfrequencies do not show yet a consistent representation of the complete system eigenvalues.

This means that the values of the eigenvalues associated to the identified eigenmodes do not match those obtained from the entire MBDyn generated model with the same design features.

Table 5.6 shows a comparison between some of the eigenvalue obtained from the two models, the MBDyn one and the one generated through substructuring. Both models are tested in a 30 kn airspeed condition.

It is immediately clear that the frequency and damping values identified by the two modes have some neat difference. The C-B model, is not capable

Mode name	**MBDyn model**		**C-B model**	
	Freq. [Hz]	damp. [adim]	Freq. [Hz]	damp. [adim]
Beam mode	5.83	0.0083	8.16	0.034
1st bend InP	8.15	0.024	19.54	0.038

Table 5.6: Bending modes comparison between the two models

at this level of the development to match MBDyn titltrotor entire model behaviour and the difference between the two descriptions is not negligible.

At this stage of the development, we can not identify with precision why this occurs. We believe, though, that the not complete validation of the substructuring process for the rotor subsystem (as explained in sections 5.4.1 and 5.4.2) plays an important role in this, since all the other steps have been extensively tested. Despite getting very close to the final results, the complete proof of the process is left to future research.

Chapter 6

Conclusions

In this work, the development of a stochastic parametric tiltrotor "whirl flutter" predictor is presented. Alongside the complete theoretical development, some early results and applications are reported as well.

The tool is based on the interaction of three elements: a parametric tiltrotor model generator called *TiPa*, the multibody general-purpose aeroelastic solver MBDyn and the software DAKOTA. The interaction of the three has been designed to provide an effective investigation tool to support the current research in the field.

TiPa has been developed specifically to provide a flexible interface to the modelling of an arbitrary configuration tiltrotor assembly in order to match the TRAST wind tunnel model flexibility.

The parametric conception of *TiPa* proved to be an essential feature to introduce a forward propagating Uncertainty Quantification (UQ) method in the formulation. A complete discussion about the optimal UQ method to spread random input variables through the multibody formulation is presented in the thesis. This led to the adoption of the non-intrusive generalized Polynomial Chaos Expansions (gPCE) technique as tool to assess the effects of random input variables on the system response. The gPCE has been introduced in the formulation thanks to the software DAKOTA.

The combination of *TiPa* and DAKOTA provides the complete definition of the stochastic parametric aeroelastic predictor. Thanks to the introduction of the UQ techniques, it is possible to execute complete sensitivity analysis of the desired input parameters effects. Moreover, the system non-deterministic response to such variables is estimated through the definition of: the stochastic V-f and V-ξ diagrams and the outputs pdf and CDF curves. For this reason DAKOTA/*TiPa* simulations provide its user a broad range of information in support of the identification of the most crit-

ical parameters in the definition of the analyzed tiltrotor stability margin.

TiPa provides two different modelling approaches to define the complete titltrotor assembly. One option is based on the definition of an entire proprotor MBDyn model, while the other relies on the generation of two individual subsystems to be joined through a substructuring approach. The former modelling option shows some intrinsic limitations due to its strong dependency on the convergence of the multibody solver. This can prevent the tool to provide a stochastic identification of the flutter condition.

The second approach has been designed to overcome this limitation. The method provides a more consistent representation of the rotor dynamics through an innovative adaptation of the multiblade coordinates (MBC) transformation. The thesis reports a complete development and validation of the entire transformation. An original application of the Craig-Bampton substructuring approach has been designed as well for MBDyn multibody matrices.

Despite most of this second modelling approach has been tested and validated, the tiltrotor models generated with this technique do not show the expected dynamical behaviour yet.

The thesis provides some guidelines to complete the validation of the process. This is left for future research.

Appendix A

Orthogonality in polynomials

The concept of orthogonality is more commonly used to define vectors and their reletive orientations in n-dimentional spaces.
Orthogonal vectors are vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ whose **dot product** is equal to zero with $\|\boldsymbol{a}\|$, $\|\boldsymbol{b}\| > 0$. Such operator is a powerful instrument since it allows representing vectors and their respective orientation with scalar variables.

Mathematicians extended this concept to other mathematical elements to have an easy and intuitive tool to define them. Polynomials can then be classified through the concept of orthogonality thanks to the *Hilber spaces* definition an the **inner product** operator.
Given two polinomials $a(x)$, $b(x)$ and the (joint) probability density function of the random variable x, $w(x)$, the inner product is defined as:

$$\langle a, b\rangle = \int a(\xi)b(\xi)w(\xi)d\xi \tag{A.1}$$

Given that, two polinomials Φ^i and Φ^j are called orthogonal if

$$\langle \Phi^i, \Phi^j\rangle = 0 \text{ for } i \neq j.$$

by looking at equation A.1, it's clear that the definition of of the operator is strongly connected to the nature of the weight function $w(x)$. Being the Wiener-Askey polynomials chaoses Φ^j complete basis of the Hilbert spaces, they are orthogonal since:

$$\langle \Phi^i, \Phi^j\rangle = \int \Phi^i(\xi)\Phi^j(\xi)w(\xi)d\xi = 0 \tag{A.2}$$

if $j \neq i$ and $w(\xi)$ is the continous probability density function of the uncertain input variable used to generate Φ^i, Φ^j according to table 2.1. Hence, according to the distribution of the uncertain variables stored in $\boldsymbol{\xi}$, different

familes of polynomials are orthogonal with respect to the variables associated probability density functions.

Appendix B

Substructuring for multibody systems

The Craig-Bampton method [5] is among the most important substructuring techniques currently available in the literature. *TiPa* analysis are based on an adaptation of the Craig-Bampton method to MBDyn multibody system matrices. While the entire DAKOTA/*TiPa* analysis is presented in chapter 4, here only brief disscussion of the key elements adopted to generate the tiltrotor substructures is presented.

B.1 Conventional applications

The Craig-Bampton method has been extensively applied to FEM structural problems to reduce overall model degrees of freedom. The techinque is based on the identification of different subareas splitting a complex model into simpler regions. Each one of them is called "substructure". The DOFs of each area are classified as either **interior** and **boundary** ones. The Craig-Bampton process applies a partial modal reduction to each substructure leaving their boundary DOFs untouched. The technique, which is not presented here, relies on the C-B transformation matrix:

$$\mathbf{C}\boldsymbol{B} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \boldsymbol{\Phi}^R & \boldsymbol{\Phi}^L \end{bmatrix} \tag{B.1}$$

where:

- $\boldsymbol{\Phi}^L$ are the normal modes of each substructure in constrained configuration (fixed boundary DOFs)

- $\mathbf{\Phi}^R$, also called *static shapes*, are the shapes deriving from units displacements of the boundary degrees of freedoms

In a conventional FEM formulation, both $\mathbf{\Phi}^L$ and $\mathbf{\Phi}^R$ can be easily identified from the system mass and stiffness matrices. For further details, the formulation is extensively presented in ref. [5].

B.2 Multibody application

Multibody dynamics equations can we written in different forms. This discussion takes as reference the structure adopted by MBDyn.

MBDyn stores its matrices in the descriptor form as:

$$\mathbf{E}\ \dot{\boldsymbol{q}} = \mathbf{A}\ \boldsymbol{q} \tag{B.2}$$

where $\boldsymbol{q} = \{\boldsymbol{x}\ \boldsymbol{\lambda}\}^T$ is the generalized coordinates vector.

The $\mathbf{A}$ and $\mathbf{E}$ matrices associated to the proper model can be used to easily identify $\mathbf{\Phi}^L$ but not $\mathbf{\Phi}^R$. This happens since the use of a formulation equivalent to the FEM one to generate the $\mathbf{\Phi}^R$ would require inverting matrices with null determinant due to MBDyn matrices structure.

B.2.1 Static shapes

To identify $\mathbf{\Phi}^R$ associated to both the rotor and the wing subsystems, *TiPa* generates them "manually". The term refers to the fact that the software generates such shapes independently from the current analysis. They are defined prior to each analysis and retrieved during *TiPa* assessments.

To correctly generate such shapes, two elements needs to be remembered. First, despite the $\boldsymbol{q}$ vector structure, only the nodal DOFs $\boldsymbol{x}$ must be properly tuned generate the correct $\mathbf{\Phi}^R$ matrix. This is true since the static shapes are defined in an equilibrium condition, hence the contribution provided by the $\boldsymbol{\lambda}$ components is null since they define internal forces.
Second, each MBDyn node has twelve DOFs (3 positions, 3 orientation, 3 momenta and 3 momenta moments). So, since the number of static shapes is associated to the numbers of the boundary nodes DOFs, twelve shapes are needed. This is twice the number of those needed in a conventional FEM approach. This problem was solved by generating the six static shapes associated to unit activations of position and rotation DOFs and consequently reusing them by translating to the left momenta DOFs. This process has proven to be a valid method (see ref. B.2.2).

The approached used to define each subsystem shapes are slightly different. They are now presented.

Wing static shapes

The wing model is very simple. For this reason it is possible and easy to extract the FEM like matrices describing its beam structure from MBDyn system matrices. If no aero panels are present, the extracted mass and stiffness matrices can be used to generate the wing static shapes. The motions describe the positions of the internal DOFs after unit activations of the wing tip ones. These shapes are then redistributed to an MBDyn like twelve DOFs node structures to complete the generation of the wing associated $\mathbf{\Phi}^R$. Once this is done, the wing C-B matrix is complete and the model can be substructured.

Rotor static shapes

The definition of the rotor static shapes is harder due to the more complex nature of the subsystem. For this reason is not possible to define the required shapes starting from equivalent FEM matrices. This time, the boundary node is represented by the node clamping the system to the ground. As consequence, $\mathbf{\Phi}^R$ contains proper rigid body motions. The shapes are this time generated by an algorithm which defines the nodes displacements according to the specific rotor geometry. Once again, the twelve shapes are generated in groups of six. Those associated to the boundary node activations are the first to be generated to be eventually "shifted downwards" to describe the motions associated to the interface momenta DOFs.
Please remember that, since the rotor behaviour is described with multi-blade coordinates, the rotor rigid body motions need to be assessed with such coordinates as well. For this reason, the shapes are generated using the conventional MBDyn DOFs and are eventually transformed to the non-rotating frame coordinates with the MBC transformation matrix $\mathbf{M}$ defined in equation 3.55 in section 3.2.5. The equation is reported once again:

$$\boldsymbol{q} = \big(\mathbf{R}_{\boldsymbol{tot}}\ \mathbf{P}\ \mathbf{T}_{\boldsymbol{mb}}\big)\boldsymbol{q}_{mb} = \mathbf{M}\ \boldsymbol{q}_{mb} \tag{B.3}$$

where $\boldsymbol{q}$ is MBDyn generalized coordinates vector and $\boldsymbol{q}_{mb}$ is its MBC version.
Thanks to $\mathbf{M}$, the rotor rigid body motions can be described in the non-rotating frame $\mathbf{\Phi}^R_{mb}$ with equation B.4.

$$\mathbf{\Phi}^R_{mb} = (\mathbf{M})^{-1}\ \mathbf{\Phi}^R \tag{B.4}$$

The normal modes $\mathbf{\Phi}^L$ needs as well to be described in the non-rotating frame. If they are extracted among the eigenvectors resulting form an eigen-

analysis on the MBC system matrices $\mathbf{A}_{mb}$ and $\mathbf{E}_{mb}$, the desired $\mathbf{\Phi}_{mb}^{L}$ modes are automatically obtained.
It is now possible to create the rotor substucture and to assemble it to the wing one.

B.2.2 Some validation results

To prove the effectiveness of the above-mentioned substructuring process for the multibody systems, here some results are presented.

Table B.1 shows a comparison of eigenfrequencies associated to the first five normal modes of the WRATS isolated wing model. The central column shows the values coming from the substructured model while the left one shows the eigenfrequencies associated to a MBDyn model. The substructured model has been developed according to the procedure presented in this Appendix starting from a MBDyn model of the wing system clamped at both ends.

Mode type	**Subs. freq** [Hz]	**MBDyn freq** [Hz]
1st bend OoP	19.71	19.71
1st bend InP	27.36	27.36
1st torsion	74.52	74.50
2nd bend OoP	157.99	157.80
2nd bend InP	180.39	180.12

Table B.1: WRATS wing modes comparison

The similarities between the two systems dynamical behaviour presented in Table B.1 show the effectiveness of the introduced substructuring approach.

Bibliography

[1] Brian M Adams et al. "DAKOTA, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis: version 6.11 user's manual". In: (2019).

[2] *Bell V-280*. 2020. URL: `https://www.bellflight.com/products/bell-v-280`.

[3] Yakov Ben-Haim. *Info-gap decision theory: decisions under severe uncertainty*. Elsevier, 2006.

[4] Robert P. Coleman. *Theory of Self-Excited Mechanical Oscillations of Hinged Rotor Blades*. ARR 3G29. NACA, 1943.

[5] Roy R. Craig Jr. and Mervyn C. C. Bampton. "Coupling of Substructures for Dynamic Analysis". In: *AIAA Journal* 6.7 (1968), pp. 1313–1319.

[6] Thierry Crestaux, Olivier Le Maître, and Jean-Marc Martinez. "Polynomial chaos expansion for sensitivity analysis". In: *Reliability Engineering & System Safety* 94.7 (2009), pp. 1161–1172.

[7] Yuting Dai and Chao Yang. "Methods and advances in the study of aeroelasticity with uncertainties". In: *Chinese Journal of Aeronautics* 27.3 (2014), pp. 461–474.

[8] F Dempewolff. "Here come the convertiplanes". In: *Popular Mechanics magazine* (1954).

[9] Collins English Dictionary. *Tiltrotor definition*. 2020. URL: `https://www.collinsdictionary.com/dictionary/english/tiltrotor`.

[10] Gaston Floquet. "Sur les équations différentielles linéaires à coefficients périodiques". In: *Annales scientifiques de l'École normale supérieure*. Vol. 12. 1883, pp. 47–88.

[11] Roger Ghanem. "Stochastic finite elements with multiple random non-Gaussian properties". In: *Journal of Engineering Mechanics* 125.1 (1999), pp. 26–40.

[12] Gian Luca Ghiringhelli et al. "Multi-body analysis of a tiltrotor configuration". In: *Nonlinear Dynamics* 19.4 (1999), pp. 333–357.

[13] ST Glusman, Robert A Hyland, and Roger L Marr. "V-22 technical challenges". In: *AGARD Advances in Rotorcraft Technologies Symposium.* 1996.

[14] F Guerroni. *TiPa.* 2020. URL: `https://gitlab.com/fedeguerro/tipa_distro/-/tree/master`.

[15] W Earl Hall. "Prop-Rotor Stability at High Advance Ratios". In: *Journal of the American Helicopter Society* 11.2 (1966), pp. 11–26.

[16] KH Hohenemser and SK Yin. "On the use of first order rotor dynamics in multiblade coordinates". In: *30th Annual National Forum of the American Helicopter Society, Preprint No. 831.* 1974.

[17] Youmin Hu et al. "An Uncertainty Quantification Method Based on Generalized Interval". In: *2013 12th Mexican International Conference on Artificial Intelligence.* IEEE. 2013, pp. 145–150.

[18] Wayne Johnson. "Dynamics of tilting proprotor aircraft in cruise flight". In: (1974).

[19] Wayne Johnson. *Rotorcraft aeromechanics.* Vol. 36. Cambridge University Press, 2013.

[20] Andrew R Kreshock et al. "Development of a New Aeroelastic Tiltrotor Wind Tunnel Testbed". In: *AIAA SciTech Forum* (2019).

[21] Raymond G Kvaternik. "A historical overview of tiltrotor aeroelastic research at Langley Research Center". In: (1992).

[22] Raymond G Kvaternik and Jerome S Kohn. "An experimental and analytical investigation of proprotor whirl flutter". In: (1977).

[23] Martin D Maisel. *The history of the XV-15 tilt rotor research aircraft: from concept to flight.* 17. National Aeronautics, Space Administration, Office of Policy, and Plans ..., 2000.

[24] P. Masarati. "Direct Eigenanalysis of Constrained System Dynamics". In: *Proc. IMechE Part K: J. Multi-body Dynamics* 223.4 (2009). doi:10.1243/14644193JMBD211, pp. 335–342.

[25] Pierangelo Masarati. "MBDyn Theory and Developer's Manual Version 1". In: *X-Devel, Politecnico di Milano, Milan, Italy* (2010).

[26] Pierangelo Masarati et al. "Soft-Inplane Tiltrotor Aeromechanics Investigation Using Two Comprehensive Multibody Solvers". In: *Journal of the American Helicopter Society* 53.2 (2008), pp. 179–192.

[27] Keith McCloskey. *Airwork: a history.* The History Press, 2012.

[28] Nicholas Metropolis and Stanislaw Ulam. "The monte carlo method". In: *Journal of the American statistical association* 44.247 (1949), pp. 335–341.

[29] FABIAN ANDRES LARA MOLINA et al. "Sensitivity Analysis of Flexible Rotor Subjected to Interval Uncertainties". In: (2019).

[30] Jérôme Morio. "Global and local sensitivity analysis methods for a physical system". In: *European journal of physics* 32.6 (2011), p. 1577.

[31] R. Niccoli. *La storia del volo. Dalle macchine volanti di Leonardo da Vinci alla conquista dello spazio. Ediz. illustrata.* White Star, 2013.

[32] Chris L Pettit. "Uncertainty quantification in aeroelasticity: recent results and research challenges". In: *Journal of Aircraft* 41.5 (2004), pp. 1217–1229.

[33] David J Piatak et al. "A wind-tunnel parametric investigation of tiltrotor whirl-flutter stability boundaries". In: (2001).

[34] Wilmer H Reed. *Review of propeller-rotor whirl flutter.* National Aeronautics and Space Administration, 1967.

[35] Helen M Regan et al. "Robust decision-making under severe uncertainty for conservation management". In: *Ecological applications* 15.4 (2005), pp. 1471–1477.

[36] Reuven Y Rubinstein and Dirk P Kroese. *Simulation and the Monte Carlo method.* Vol. 10. John Wiley & Sons, 2016.

[37] Adrian Sandu, Corina Sandu, and Mehdi Ahmadian. "Modeling multibody systems with uncertainties. Part I: Theoretical and computational aspects". In: *Multibody System Dynamics* 15.4 (2006), pp. 369–391.

[38] Corina Sandu, Adrian Sandu, and Mehdi Ahmadian. "Modeling multibody systems with uncertainties. Part II: Numerical applications". In: *Multibody System Dynamics* 15.3 (2006), pp. 241–262.

[39] Pallav Sarma, Jiang Xie, et al. "Efficient and robust uncertainty quantification in reservoir simulation with polynomial chaos expansions and non-intrusive spectral projection". In: *SPE Reservoir Simulation Symposium.* Society of Petroleum Engineers. 2011.

[40] Ilya M Sobol. "Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates". In: *Mathematics and computers in simulation* 55.1-3 (2001), pp. 271–280.

[41] Michael Stein. "Large sample properties of simulations using Latin hypercube sampling". In: *Technometrics* 29.2 (1987), pp. 143–151.

[42] Bruno Sudret, Stefano Marelli, and Joe Wiart. "Surrogate models for uncertainty quantification: An overview". In: *2017 11th European conference on antennas and propagation (EUCAP)*. IEEE. 2017, pp. 793–797.

[43] ES Taylor and KA Browne. "Vibration isolation of aircraft power plants". In: *Journal of the Aeronautical Sciences* 6.2 (1938), pp. 43–49.

[44] F Vogelzang. *Flying-machine*. US1353501A. 1920. URL: `https://patents.google.com/patent/US1353501`.

[45] Norbert Wiener. "The homogeneous chaos". In: *American Journal of Mathematics* 60.4 (1938), pp. 897–936.

[46] Dongbin Xiu and George Em Karniadakis. "The Wiener–Askey polynomial chaos for stochastic differential equations". In: *SIAM journal on scientific computing* 24.2 (2002), pp. 619–644.

[47] Shuxing Yang, Fenfen Xiong, and Fenggang Wang. "Polynomial Chaos Expansion for Probabilistic Uncertainty Propagation". In: *Uncertainty Quantification and Model Calibration* (2017), p. 13.

[48] William T Yeager Jr and Raymond G Kvaternik. "A historical overview of aeroelasticity branch and transonic dynamics tunnel contributions to rotorcraft technology and development". In: (2001).

[49] Hyeonsoo Yeo et al. "Comparison of CAMRAD II and RCAS Predictions of Tiltrotor Aeroelastic Stability". In: *Journal of the American Helicopter Society* 63.2 (2018), pp. 1–13.

[50] Jianhua Zhang, Hao Kang, and Edward C Smith. "Wing Extension Design and Tailoring for a Scaled Tiltrotor Wind Tunnel Model". In: *AIAA Scitech 2019 Forum*. 2019, p. 1368.