Technische
Universität
Braunschweig

Master's Thesis

Extension of a CFD-CSM Toolchain for Simulating the Wing of a Regional Aircraft with Distributed Propulsion Employing an Actuator Disk Model

David Siegmund

September 5, 2024

DLR Institute of Aerodynamics and Flow Technology
Prof. Dr. Stefan Görtz

Institute of Aircraft Design and Lightweight Structures
Prof. Dr.-Ing. Ingo Staack

Supervisor:
Dr.-Ing. Lars Reimer

Abstract

The present thesis investigates fluid-structure interaction (FSI) and propeller interference effects for a wing of a multi-propeller regional aircraft by conducting high-fidelity multidisciplinary simulations employing the actuator disk model. For that, a software extension is implemented which integrates the existing actuator disk model of DLR's (German Aerospace Center) flow solver *TAU* into a coupled simulation involving fluid dynamics and structural mechanics solvers (CFD-CSM).

With the transition towards a more climate-compatible aviation, the aerospace industry faces a major challenge in the upcoming decades. Confronting this challenge, the EU funds large research projects such as HERA. HERA's goal is to develop and assess novel architectures for propeller aircraft with hybrid propulsion. One of its projected tasks is to advance current modeling capabilities for these types of aircraft. Reynolds-averaged Navier Stokes (RANS) analyses are a well-established simulation technique. In combination with the actuator disk model they provide an accurate and cost-effective solution for simulating propeller-wing aerodynamics. The use of lightweight materials and improved manufacturing techniques enable more slender and flexible wing designs, making them more aerodynamic but also more susceptible to FSI effects. This challenges the limits of monodisciplinary Computational Fluid Dynamics (CFD) analysis; coupled CFD-CSM simulations are, however, able to take FSI effects into account. The *FlowSimulator* suite has been developed in a joint effort by *Airbus*, *ONERA*, and DLR to create a software framework for multi-disciplinary simulations. While the flow solver *TAU* includes an AD model, it has not been integrated and applied in *FlowSimulator*-based CFD-CSM simulations.

Therefore, this thesis implemented a software module which creates an appropriate interface for the AD model, allowing it to be used in multi-disciplinary simulations for the first time. With that, a series of coupled CFD-CSM analyses of a HERA use case are conducted. The analyses investigate FSI and propeller interference effects on the wing and assess the influence of several wing and propeller parameters on aerodynamic performance on the basis of multiple design variations. The results show the significance of wing flexibility and propeller-wing interaction effects which depend on the combined influence of various interrelated design parameters. The CFD meshes needed for the different designs are created using an automated mesh generation process based on a parametric CAD model. In preparation for a future application in multidisciplinary optimisation (MDO) scenarios, the AD-integrated process is tested under high loads and extreme parameters, confirming its robustness for large deformations and a wide range of parameter inputs.

With this thesis, an important capability for the multidisciplinary analysis of propeller aircraft is established. Advanced modeling techniques, which can take both the fluid and the structural domain into account, are key in identifying more efficient but also feasible designs. In the future, the developed workflow can be expanded and used for trade-off studies or integrated into MDO toolchains, enhancing the development of novel propeller aircraft architectures.

Zusammenfassung

Die vorliegende Arbeit untersucht Fluid-Struktur-Interaktion (FSI) und Propellerinterferenzeffekte an einem Regionalflugzeug mit verteiltem Propellerantrieb mit Hilfe hochauflösender multidisziplinärer Simulationen unter Verwendung eines Wirkscheibenmodells. Dazu wird eine Softwareerweiterung implementiert, die das bestehende Wirkscheibenmodell des Strömungslösers *TAU* des DLR (Deutsches Zentrum für Luft- und Raumfahrt) in eine Fluid-Struktur-gekoppelte Simulation (CFD-CSM) integriert.

Mit dem Übergang zu einer klimaverträglicheren Luftfahrt steht die Luftfahrtindustrie in den kommenden Jahrzehnten vor einer großen Herausforderung. Um dieser Herausforderung zu begegnen, finanziert die EU umfassende Forschungsprojekte wie beispielsweise HERA. Im Rahmen des HERA-Projektes sollen neuartige Architekturen für Propellerflugzeuge mit Hybridantrieb erforscht werden. Eine der geplanten Aufgaben besteht darin, die derzeitigen Fähigkeiten zur Simulation solcher Flugzeugkonfigurationen zu verbessern. Reynolds-gemittelte Navier-Stokes-Analysen (RANS) sind eine etablierte Methode, die in Kombination mit dem Wirkscheibenmodells für angemessen genaue und kostengünstige Option zur Simulationen von Propellerkonfigurationen bietet. Der Einsatz von Leichtbaumaterialien und verbesserten Fertigungsverfahren ermöglicht gestrecktere und flexiblere Flügelkonstruktionen, die zwar aerodynamischer, aber auch anfälliger gegenüber FSI-Effekten sind. Während monodisziplinäre CFD-Simulationen (Computational Fluid Dynamics) dadurch an ihre Grenzen geraten, sind gekoppelte CFD-CSM Simulationen im Gegensatz dazu in der Lage FSI-Effekte abzubilden. Die *FlowSimulator*-Suite ist ein Software-Framework, das gemeinschaftlich von *Airbus*, *ONERA* und dem *DLR* entwickelt wurde, um die Durchführung multidisziplinärer Simulationen zu vereinfachen. Obwohl der DLR-Strömungslöser *TAU* bereits ein Wirkscheibenmodell enthält, wurde es bisher noch nicht in *FlowSimulator*-basierten CFD-CSM-Simulationen integriert und angewendet.

Deshalb wurde in dieser Arbeit ein Softwaremodul implementiert, das eine geeignete Schnittstelle für das Wirkscheibenmodell schafft, sodass es erstmalig in multidisziplinären Simulationen eingesetzt werden kann. Mit Hilfe dieser neugeschaffenen Fähigkeit wurden eine Reihe von gekoppelten CFD-CSM-Analysen am Beispiel eines konkreten Anwendungsfalls aus dem HERA-Projekt durchgeführt. Mit diesen Analysen wurde der Einfluss der FSI-Effekte und der Propellerströmung auf den Flügel untersucht und der Effekt verschiedener Flügel- und Propellerparameter auf die Flugleistungen anhand mehrerer Designvariationen bewertet. Die Ergebnisse machen die Signifikanz der Flügelflexibilität und der Propeller-Flügel-Interaktionseffekte deutlich, die durch die Wechselwirkung zahlreicher Flügel- und Propellerparameter entstehen. Die für die Variationen benögtig-

ten CFD-Netze wurden mit Hilfe eines automatisierten Netzgenerierungsprozesses auf der Grundlage eines parametrischen CAD-Modells erstellt. In Hinblick auf eine künftige Anwendung in Optimierungsszenarien wurde der entwickelte Simulationsprozess unter hohen Lasten und extremen Parametern getestet, wodurch seine Robustheit gegenüber großen Verformungszuständen und einem weiten Spektrum an Parametereingaben bestätigt werden konnte.

Im Rahmen dieser Arbeit konnte eine wichtige Fähigkeit für die multidisziplinäre Analyse von Propellerflugzeugen geschaffen werden. Hochentwickelte Modellierungstechniken, die sowohl die Fluid- als auch die Strukturmechanik miteinbeziehen, sind für die Identifizierung effizienterer Flugzeug-Designs, die gleichzeitig technisch realisierbar sein sollen, entscheidend. In Zukunft kann die entwickelte Arbeitsablauf erweitert und in Vergleichsstudien und multidisziplinären Optimierungs-Toolkette eingesetzt werden, um so die Entwicklung neuartiger Propellerflugzeugarchitekturen voranzutreiben.

Task

Extension of a CFD-CSM Toolchain for Simulating the Wing of a Regional Aircraft with Distributed Propulsion Employing an Actuator Disk Model

Background

In support of the transformation of aviation towards climate-neutrality, intensive research is going on in the definition and design of alternative emission-free/reduced emission aircraft architectures, especially for the regional aircraft market. According to the current state of research, promising configurations are characterized by a number of purely electric or hybrid-electric propeller drives distributed across the wingspan. Optimal design solutions that are compatible under realistic operational conditions can only be achieved by taking into account all the interaction effects and constraints between the disciplines involved. This suggests the use of multidisciplinary optimization (MDO) technologies. The currently ongoing EU project HERA (Hybrid-Electric Regional Aircraft), which aims to define feasible hybrid-electric regional aircraft concepts with the objective of entry-into service by 2035, operates against this background and with the above objective. In this context, the further development and application of digital enablers such as the necessary multidisciplinary analysis and optimization capabilities based on high-fidelity methods for aerodynamics (CFD) and structure (CSM) are to be advanced. The planned master's thesis will make a decisive contribution on the way to MDO for hybrid-electric regional aircraft. The aim of the master's thesis is the multidisciplinary (here: CFD-CSM-coupled) analysis (MDA) of aircraft designs originating from the HERA project on the basis of high-fidelity methods as a basic prerequisite for the MDO applications later planned in HERA. The aircraft concepts pursued in HERA are characterized by the presence of propellers as a challenging novelty with regard to the MDA simulation capabilities available at DLR and others. Although their efficient steady-state averaged modeling in purely monodisciplinary CFD analysis using actuator disk technologies is state of the art, their inclusion in the MDA context and the associated challenges have not yet been considered. With regard to this shortcoming and the previous lack of know-how about the influence of aero structure interaction effects on the aero performance of configurations with actuator disk-modeled propellers, the master's thesis is intended to provide a remedy.

Task definition

Against the above background, the objective of the master's thesis is threefold:

1. The existing CFD-CSM-coupled analysis capabilities at DLR based on the CFD code TAU are to be suitably extended with regard to the inclusion of actuator-disk modeled propulsion systems and to evaluate and demonstrate the robust usability of

the extended process in application to a simplified aircraft configuration borrowed from the HERA project. This includes the appropriate setup of the aero-structure coupled simulation problem as well as the configuration of the simulation settings to handle the interaction aspects (CFD-CSM load/deformation transfer, CFD mesh deformation problem, etc).

2. In preparation for the MDO scenarios projected in HERA, a crucial element for the targeted gradient-based optimizations is the sensitivity calculation. In terms of parameter variation, it is necessary to check whether the existing codes and the CFD-CSM process chain, which is to be expanded to include the active disk modeling, can be used to robustly calculate sensitivities with regard to the position and geometry of the active disks and - optionally - changes to the wing geometry.

3. Evaluation of the aero-structure interaction influence on the aero performance by means of CFD-CSM coupled simulations for a selection of actuator disk or propeller designs (propeller position, operating conditions, optional number of propellers) and optional wing planform variations in comparison to corresponding results based on mono-disciplinary aero analyses.

The following steps are essentially required:

- Literature research on the theory of actuator disk modeling and the state of the art of CFD CSM coupling with high-fidelity numerical methods

- Familiarization with the existing *FlowSimulator*-based CFD-CSM coupling processes at DLR and the methods and procedures used therein, including actuator disk modeling. The process variant with use of the CFD solver *TAU* and CSM solver *NASTRAN* should be used (optionally, if available, alternatively the CSM solver and optimization framework *LAGRANGE* from *Airbus Defense & Space* to be used in the MDO following this work)

- Set-up of a parametric CAD model for the aeroshape (wing) including parametric actuator disk models, derived from the model data provided by the HERA project
    - Parametric modeling of the actuator discs taking into account geometric boundary conditions
    - Parametric modeling of the wing (optional)

- Automated creation of CFD meshes and identification of suitable configuration-specific meshing parameters; verification of common mesh quality criteria and validation runs with *TAU*

- Setting up a parametric structural model (beam model) that changes with the positioning of the actuator disks based on the model data provided by the HERA project

- Set up a parametric structural model that changes with the aeroshape (beam model, possibly more realistic GFEM) based on the model data provided by the HERA project (optional)

- Set-up and integration of the actuator disk modeling into the simulation with generic propeller parameters, possibly with propeller specifications and operating conditions provided by the HERA project (if available)

- Evaluation and selection of suitable projection methods for the load and deformation transfer step in the CFD-CSM coupling problem

- Evaluation of the CFD mesh deformation problem with special consideration of the aspects and, if applicable, constraints due to the presence of the actuator disks in the CFD mesh, if necessary special treatments for e.g. planarity preservation, ensuring CFD mesh quality and process robustness within the expected deformation range

- Set up the CFD-CSM coupled simulation chain according to objective (1) and test and demonstrate its usability for the baseline configuration. Exemplary selection and definition of suitable evaluation criteria for the aero performance analyses and parameter studies.

- Execution of parameter studies in the sense of a sensitivity analysis with regard to the robustness of the process chain and in the sense of objective (2) for selected propeller design variations (and optionally aero shape changes)

- Execution of the aero- and aero-structure-coupled analyses according to objective (3) together with corresponding post-processing and evaluation of the role of the consideration of the structural response on the aero-performance behavior

Contents

List of Figures

Nomenclature

Latin Terminology

b	wing span [m]
c	chord length [m]
c	sonic speed [m/s]
C_L	lift coefficient
C_D	drag coefficient
C_p	pressure coefficient
D	propeller diameter [m]
D	drag force [N]
F	force [N]
L	lift force [N]
m	mass [kg]
Ma	Mach number
P	power [W]
Q	torque [Nm]
q	dynamic pressure [Pa]
r	radius, radial coordinate [m]
Re	Reynolds number
S_{ref}	reference area [m²]
t	time [s]
T	thrust [N]
u	flow velocity [m/s]
w	induced flow velocity [m/s]
x, y, z	spatial coordinates [m]
y^+	dimensionless wall distance

Greek Terminology

α	angle of attack [°]
β	blade pitch angle [°]
Δ	difference
ϵ	propeller angle of incidence [°]
η	efficiency

ϕ	helix angle [°]
μ	dynamic viscosity [Pa·s]
ρ	density [kg/m³]
Ω	angular velocity [rad/s]

Indices

cr	cruise
ind	induced
∞	onflow condition
loc	local
prop	propeller / propulsive

Abbreviations

acc.	according
approx.	approximately
AD	actuator disk
AUSM	Advection upstream splitting method
BET	Blade-Element Theory
CAD	computer-aided design
CFD	computational fluid dynamics
CSM	computational structural mechanics
DEP	distributed electric propulsion
DLR	German Aerospace Center (*ger.:* Deutsches Zentrum für Luft- und Raumfahrt)
DNS	direct numerical simulation
e. g.	for example
Eq.	equation
EU	European Union
(e)VTOL	(electric) vertical take-off and landing
Fig.	figure
FS	FlowSimulator
FSDM	FSDataManager
GDP	gross domestic product
GHG	greenhouse gas
HERA	Hybrid-Electric Regional Architecture
HPC	high-performance computing
ICAO	International Civil Aviation Organization

IEA	International Energy Agency
IPCC	Intergovernmental Panel on Climate Change
MDA/O	multi-disciplinary analysis and optimisation
min.	minute(s)
MTOW	maximum take-off weight
NACA	National Advisory Committee for Aeronautics
NASA	National Aeronautics and Space Administration
prop.	propeller
RANS	Reynolds-averaged Navier-Stokes
RBF	radial basis funtion
res.	resulting
rpm	revolutions per minute
SA	Spalart-Allmaras turbulence model
SAF	sustainable aviation fuel
SARC	SA model with rotation and curvature correction
Sec.	section
spec.	specified
SRIA	Strategic Research and Innovation Agenda
TLAR	top-level aircraft requirements
WTP	wing-tip mounted propellers

1. Introduction

1.1. Motivation

It is scientific consensus that the green-house gas emissions from the onset of industrialisation have initiated a global warming trend. Without intervention, this will cause significant disruptions to earth's ecosystems. As of 2021, climate change had already caused an increase of of 1.1 °C in the global average temperature. A further increase to 2°C could already trigger irreversible climatic effects and lead to serious repercussions for societies around the globe [7]. Consequently, 197 countries committed to reducing their green-house gas (GHG) emissions in the 2015 Paris-agreement to limit global warming to well below 2°C. Of these, the major historical emitters pledged to achieve net-zero carbon emissions by mid-century. According to the 2022 IPCC[1] report, to stay below a 2°C rise, an emission budget of 1,150 Gt CO_2 remains – assuming a certainty of 66% [8]. With an average annual emission of around 56 Gt CO_2-equivalent between 2010 and 2019, about 20 years at current levels are left to reach net zero. Although emissions continue to rise, the pace has slowed recently, possibly indicating a forthcoming trend reversal. However, the short time frame available highlights the need for immediate and concerted action across all sectors.

In the public discourse about climate change, air travel often finds itself as a target of criticism for its high greenhouse gas (GHG) emissions. Yet, surprisingly, aviation is merely responsible for around 3% of the annual CO_2 emissions, placing it at lower end of the list of emitters in the transport sector. For context, road traffic (15%) and shipping (11%) are the biggest contributors to the CO_2 emissions this sector by a large margin [9]. However, these relatively low numbers draw a misleading picture for two main reasons. Firstly, aviation's climate impact is not solely caused by CO_2 emissions, but also involves other aerosols like nitric oxides, water vapour and the subsequent formation of contrail cirrus, which have a significant cumulative impact [1]. Including these non-CO_2 sources, which are likely responsible for most of the short-term warming effect, aviation's actual global warming impact amount to roughly 5% [1]. Secondly, air travel has a disproportionately high impact when measured per individual. The reason for the comparatively low global effect is that only a minority of the world's population, predominantly from high-GDP countries, frequently flies [10]. However, in their 2020 report on future energy systems and technology, the *Internation Energy Agency* (IEA) expects that demand for air travel and cargo will triple by 2070 [10]. This is in agreement with market forecasts of the two largest aircraft manufacturers, *Airbus* and *Boeing*, whose current order backlog amounts to roughly 12 years of their combined production capacity [11]. With the major-

[1] Intergovernmental Panel on Climate Change

ity of these orders originating from South-East Asia or the Middle East, aviation's market growth is primarily generated in regions whose inhabitants as yet fly only rarely, if at all. Even if future demand grows less rapidly than what the IEA's forecast claims, additional demand is always going to thwart the goal of lowering aviation's GHG emissions, since every increment of market growth directly counteracts any gradual improvement in aircraft efficiency. Future growth is not unlikely, given that it would simply continue the historic trend. Between 1960 and 2020, CO_2 emissions caused by aviation grew almost persistently despite remarkable gains in transport efficiency which were achieved during the same time period (see Fig. 1.1). The shown diagrams underscore how all improvements in terms of transport efficiency, which is a combination of aircraft efficiency and passenger load factors, are outpaced by the extra demand. Thus, it is reasonable to expect aviation's share in global GHG emissions to rise in the coming decades; especially so, as decarbonisation of various other GHG intensive activities, like road transport or electrical energy production, is already underway.

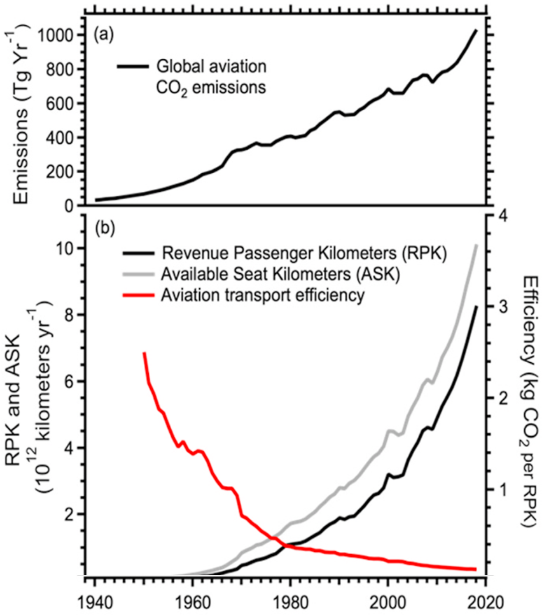


Figure 1.1.: Historic development of aviation's CO_2 emissions, capacity and transport efficiency; by Lee et al. (2021) [1]

For the aviation industry in particular, the transition to a climate-compatible future may be more difficult to achieve than for other industries. [12]. Apart from the obvious economic aspects of transforming an already capital-intensive industry, there are several fundamental technological issues which have to be resolved before wide-scale commercial adoption of aircraft with net-zero emissions becomes even feasible. A major challenge for the design of eco-friendly aircraft is finding a viable alternative for its current energy carrier. Kerosene, the most widely used aviation fuel today, has a specific energy of around 12,000 Wh/kg [13], whereas state of the art battery technology peaks at more than an order of magnitude less [14]. This weight disparity limits battery deployment to short-range and low payload applications. With further advancements of this technology through the intensive ongoing research of electro-chemical energy storage, implementation of batteries is likely to increase successively, not least since they are well suited for hybrid-electric architectures [15]. Another potential substitute for kerosene is hydrogen, which, despite it bringing its own technical challenges, is believed to be the most realistic future candidate, especially for long-range applications [16]. Hydrogen does provide a specific energy content roughly equivalent to kerosene but falls short in terms of volumetric energy density. For it to provide substantial merit for aviation, hydrogen would likely have to be liquefied and stored in cryogenic tanks or kept under extremely high pressures elsewise [17]. This does not only add technical complexity, but also diminishes the fuel's overall energy efficiency. Lastly, a large-scale employment of hydrogen in aviation might be impeded by insufficient supply caused by a continuing shortage of large-scale green hydrogen production capacities and rising competitive demand from other decarbonising industries [18]. A different possible approach is using so-called sustainable aviation fuels (SAF) which are chemically similar to regular kerosene but, unlike their fossil counterpart, artificially synthesised from organic base material. SAF's great advantage is that they cause little to no technological disruption, as they are already compatible with the jet engines currently in use. In fact, EU regulations require a progressively increasing blending of regular kerosene with SAF in the upcoming decades [19]. A big issue related to SAF is that their production process is so energy-consuming that, according to Braun et al., the scalability of this technology seems economically questionable [20]. Therefore, the authors conclude that even in the long-term SAF will continue to be more expensive than kerosene today. In summary, replacing kerosene as the primary energy carrier for aviation will be technically and economically challenging. Since there is not one obvious substitute, the mid- to long-term future of aviation remains uncertain. Nevertheless, one possible benefit of this situation is that it could stimulate innovation by encouraging the industry to explore and implement more unconventional concepts.

A fundamental problem with any form of combustion powered propulsion, which has not been addressed so far, is that all of these concepts are not actually climate-neutral. Even under the assumption that all of the CO_2 created during operation can be offset in the production process, the combustion of any fuel alternative would still emit various other green-house gases, especially water. As mentioned earlier, a substantial fraction if not a majority of aviation's warming effect comes from these non-CO_2 sources. Researchers from the German Aerospace Center (DLR) estimate that in a scenario of wide-spread future operation of aircraft burning green fuel alternatives, the climate effect by contrail cirrus would be higher than today [21]. If true, this would put turbine engines in general at a big disadvantage for being a considerable choice of propulsion system in future climate-friendly aircraft. A remaining option, which does not rely on combustion, are electrically driven propellers powered either by batteries or hydrogen fuel cells.

Hybrid aircraft architectures present an opportunity for a progressive transition from conventional to renewable propulsion systems. As part of its *Green New Deal* strategy, the European Union (EU) funds large public-private partnerships with the long-term goal to achieve net GHG-neutrality by 2050 [22]. These research and innovation (R&I) programmes are intended to focus efforts and expertise from private and public contributors across the European continent. The *Clean Aviation* programme for example, involving over 40 different members [23], is set out to cooperatively develop proof-of-concepts for a number of hybrid-electric aircraft designs for the regional and short-range market. One of the several work packages defined in the programme roadmap (SRIA[1]) is HERA: **H**ybrid-**E**lectric **R**egional **A**rchitecture. The project's aim is to develop a viable hybrid aircraft design (see Fig. 1.2) and to evaluate the feasibility of different architectures with respect to their performance and ramifications for other systems [24]. The ambitious target metrics defined for the HERA project are: fuel-burn reduction of 50% and emission reduction of 90%, relative to a comparable state-of-the-art aircraft from 2020 [25].

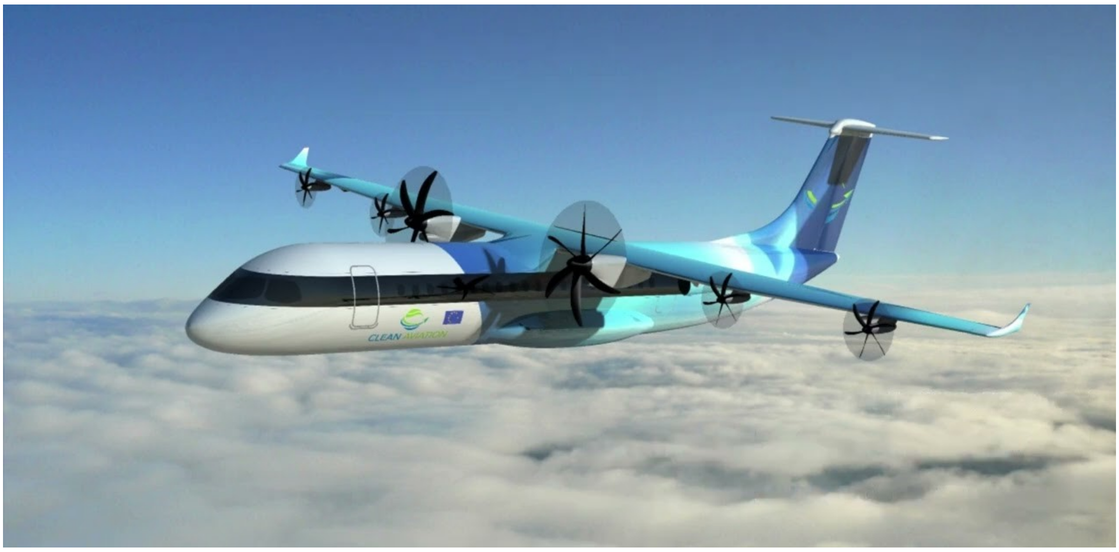


Figure 1.2.: Visualisation of a conceptual aircraft investigated in the EU-project HERA; from project-hera.eu [2]

The advent of electrified propulsion could have a profound impact on the design of future aircraft. Electric engines and propellers are both known for their high efficiencies and thus present a well-suited option for modern propulsion architectures. This is primarily because electric propulsion systems are, generally speaking, scale-independent [26]. The efficiency of a thermal engine, on the other side, is closely coupled to its nominal sizing due to fundamental thermodynamic reasons. Additionally, electric power can usually

[1] Strategic Research and Innovation Agenda

be distributed more easily than mechanical power. These features combined give rise to an unprecedented level of design flexibility with respect to the allocation of propulsors across the aircraft. Potential benefits of this concept known as *Distributed Electric Propulsion* (DEP) are a higher maximum wing lift and increased lift generation at low air speeds as well as a reduced need for control surfaces [26]. However, DEP comes with a few disadvantages as well. Apart from increased complexity on aircraft system level, propeller-wing interaction can have an adverse effect on overall aircraft efficiency. Propeller interference can also exacerbate turbulent wing flow, reducing the potential for laminar drag savings. To summarise, more electrified, multi-propeller aircraft designs could hold a lot of unused potential for the future of aviation. To evaluate their true merit, intensive research involving a holistic, multi-disciplinary approach is needed.

1.2. Literature Review

1.2.1. Propeller-Wing Interaction

The aerodynamic interference between wings and propellers is governed by a multitude of flow phenomena, emerging from the inherently unsteady interaction between the high-energy slipstream and the flow around the wing, involving viscous and compressibility effects. Due to its complex nature, propeller-wing interaction has been a subject matter of countless publications. One of the earliest experimental investigations was carried out by Prandtl in 1921, in which he described basic mechanisms of this interaction [3]. An important finding of his was that the interaction between propellers and wings can have a positive impact on the overall aerodynamic performance (e.g. augmented lift), but also negative ones, like higher wing drag and reduced propeller thrust. These findings have since been reproduced and more accurately quantified by numerous publications among which are the works by Veldhuis.

Veldhuis has been a prolific researcher on the topic of propeller-wing interaction. In his 2005 dissertation, Veldhuis conducted a comprehensive investigation into this topic by conducting a series of wind-tunnel experiments to refine low-fidelity modeling techniques [4]. In a subsequent analysis he assessed the quality of his low-fidelity models by comparing them with high-fidelity numerical simulations, referencing both against the experimental results. Using his insights, Veldhuis also performed a wing design optimisation based on his findings. Providing a theoretical foundation for his work, Veldhuis begins by laying out a detailed description of the involved flow phenomena and the intricacies of the interaction mechanisms at play. A primary mode of interference he describes is the induced flow of the propeller slipstream affecting the aerodynamic wing forces. The spinning propeller blades create a wake of accelerated and rotating fluid surrounded with helical shaped vortices originating from their tips. As a consequence of the accelerated flow in the slipstream, the flow tube contracts with increasing distance to the propeller plane due to reduced pressure, as shown in Fig. 1.3. Because the contraction creates a radial velocity directed at the center of the tube, the local flow direction also varies depending on the position relative to the tube's axis. To characterise the interference effect, the flow of the slipstream can be divided into two components, the axial flow and the (rotating) swirl flow, each having a distinct effect on the wing aerodynamics. While being azimuthally symmetrical, the axial component shows a characteristic radial distribution. From the propeller root outwards the load first generally increases until it peaks at around three quarters of the blade length before fading away towards the tip of the blade. As a result of the increased flow velocity, the wing experiences a similarly distributed lift enhancement at the wing section immersed in the wake. The swirl component, on the other hand, does not produce an azimuthally constant velocity distribution, since the swirl flow rotates in unison with the propeller's sense of rotation. The wing section downstream of the up-going blade experiences an increase in local angle of attack, whereas the downwash of the down-going blade causes an opposite effect. The superposition of these two effects is shown qualitatively in the schematic in Fig. 1.4. The figure highlights how the two mechanisms, the accelerated flow effect and the angle of attack effect, either amplify or counteract each other, depending on the propeller's sense of rotation. Veldhuis found that a propeller revolving in the inboard-up direction leads to an increase in lift-over-drag ratio [27]. Additionally, it can be seen that the propeller also affects a wider area of the wing (denoted by W-I and W-IV in Fig. 1.4) than the sections directly located in the path of the propeller slipstream. In his wind tunnel experiments, Veldhuis investigated the sensitivities of different propeller positions typical for a two-engine propeller airplane. He found that the vertical positioning has by far the highest impact on the wing's aerodynamic performance compared to the span- or stream-wise position. This can be attributed to, both, the unequal distribution of the induced velocity over the propeller disk and the strong relation between the vertical propeller location and the area of the wing surface covered by the slipstream flow, Veldhuis reasons. In summary, the correct alignment of a propeller can be exploited to increase the air flow over the wing, reliably providing additional lift even in low-speed flight or under high angles of attack [27].

A central aspect of propeller-wing interference is that it is a reciprocal relationship, meaning that the influence of the wing flow on the propeller is not negligible [4]. This is true for tractor and pusher propellers alike and, according to Veldhuis, driven by two main mechanisms. First, the wing circulation causes a slight upwash in front and a downwash behind the wing. In case of the tractor configuration investigated by Veldhuis, this then has the same effect as if the propeller disk would be at an positive angle of incidence. Because of the geometry of a rotating blade, a non-zero incidence angle causes the local angle of attack to oscillate depending on the momentary azimuthal position of the blade, with the minimum angle being located at the upgoing position (P-IV) and the maximum at the opposite side (P-II). This, in turn, causes the blade load to oscillate over a revolution as well, producing a small yawing moment. Second, the presence of the wing downstream of the propeller partially reduces the angular momentum in the wake similar to a stator vane. This effect, known as swirl recovery, improves propulsive efficiency. Swirl recovery

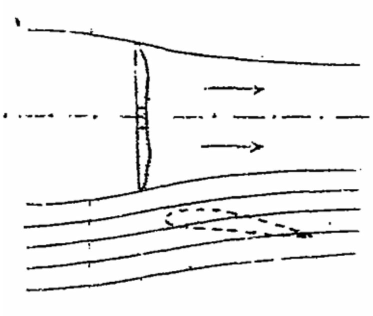


Figure 1.3.: Illustration of a contracting propeller wake; by Prandtl (1921) [3]

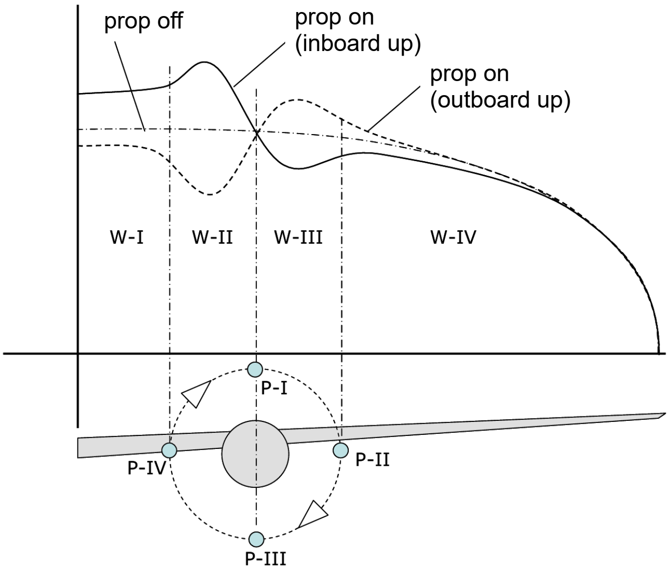


Figure 1.4.: Characterisation of wing load distribution affected by propeller interference; after Veldhuis (2005) [4] (labels corrected by the author of this thesis)

has been described by other authors like Draper et al. as well who stated that it is sensitive to, again, the propeller position and its angle of attack [28]. Veldhuis later found it to be affected by other factors such as the propeller's power setting and wing loading, too [29]

Because of the many interdependent flow phenomena at play, designing an appropriate low-fidelity model for propeller-wing interference is no trivial task. A common approach is to apply the vortex-lattice-method, emulating the wing circulation, and to superimpose a (semi)-empirically derived propeller-wake model which produces a combined flow field solution. Using a combined experimental and a low-fidelity computational approach, Witkowski et al. developed one of the first propeller-wing interaction models in 1989. Their setup involved a untapered half-wing with a Purdue propeller model in a tractor configuration at a Reynolds number of 470,000 and a Mach number of 0.1. According to their results, the authors were able to predict total lift and propeller performance with reasonable accuracy. Their total drag prediction, however, deviated by up to 12% from their experimental results. Veldhuis expanded this modeling approach by implementing a so-called full-interaction model in which the flow disturbance by the wing also affects the inflow conditions of the propeller model, better accounting for the real-world interference conditions mentioned above. With that, Veldhuis was able to predict the total lift and lift distributions as well as propeller performance metrics for a wider set of propeller-wing arrangements. Prediction accuracy deteriorated, though, at the extremes of the parameter ranges investigated. The quality of the drag predictions was generally worse.

With continuously increasing computational power, it becomes progressively easier to employ high-fidelity CFD (computational fluid dynamics) methods for a wider range of applications. For the problem of investigating propeller-wing interference, a RANS (Reynolds-averaged Navier-Stokes) simulation provides a few key advantages over low-fidelity approaches. Apart from a presumably higher accuracy of lift and drag distributions, a RANS solution would be able to resolve the chord-wise distribution of loads and therefore pitching or, respectively, torsion moments. Furthermore, a RANS approach inherently models swirl recovery, which low-fidelity models not necessarily do, as well as viscous interactions, compressibility and other interference effects between propeller and wing, nacelle, or even between multiple propellers. Unfortunately, modeling moving propeller blades in a high-fidelity CFD analysis adds a great amount of complexity. The additional computational effort and expertise needed would generally be disproportionate to the surplus value provided by the results. Therefore, several scientific publications related to this topic investigate the efficacy of the momentum-theory based actuator disk (AD) model as an alternative for a fully-resolved propeller. With an actuator disk, all the forces exerted on the flow by the propeller are averaged and reduced to a boundary condition acting on a 2D disk surface, thus eliminating the need for a time-resolving computation. This allows the AD model to be employed in (steady-state) RANS simulations greatly reducing the associated computational cost. Despite its inability to resolve transient flow phenomena, the resulting induction flow produced by the AD model is still globally similar to that of a propeller (more on actuator disk theory in Sec. 2.3).

One of the publications mentioned is the research from Stokkermans et al., who investigated the aerodynamic performance of wing-tip mounted propellers with a particular interest in the capabilities of actuator models for this purpose. In order to assess the models, the authors used wind-tunnel measurements as well as CFD results of a fully-resolved propeller to compare and validate the two employed actuator models: the actuator disk and the actuator line. In contrast to the disk model, an actuator line model is also able to reproduce transient flow features, representing a middle ground between the AD model and a direct numerical simulation (DNS) in terms of computational requirement. The authors' computational setup consists of a single tractor propeller attached to wing with a nacelle at a fixed distance from its leading edge. The experimental results are obtained in a low turbulence wind-tunnel at an onflow velocity of 40 m/s. For their CFD setup, Stokkermans et al. constructed two domains, one to test the isolated propeller and one for the complete setup which featured so-called sliding mesh interfaces. For each domain five meshes of varying sizes were generated in order to conduct a grid independence study. The mesh sizes for the complete setup ranged from 21 million cells for the coarsest one up to 107 million cells for the finest mesh. Following the results from Kim and Ree [30], who compared various turbulence models for modeling wingtip vortices, the Spalart-Allmaras (SA) turbulence model was selected as a primary choice. It was found to provide the best accuracy among the one- and two-equation models investigated. In an effort to reproduce the findings from Kim and Ree, the k-ω model was chosen as a secondary choice.

In agreement to prior research, the results from Stokkermans et al. confirmed the superiority of the SA-model as it could most accurately predict total pressure in the wake while the other investigated models suffered more severely from numerical diffusion. The SA-model was also able to predict flow separation more reliably than the k-ω model. Still, the comparison of the fully-resolved propeller CFD simulation with wind-tunnel data revealed discrepancies in the flow quantities: In the case of the pressure coefficient an error of 3.1% for the maximum value and 0.8% deviation for the integral value was found. The authors' results also underscored the actuator models' capabilities. Between the actuator line model and the fully-resolved propeller there was almost no detectable deviation. The results from the actuator disk model, on the other hand, showed a lower pressure on the upper side of the wing leading to a slight overprediction of lift of 3.9%. The authors attribute this effect to the differences in the interaction of wing and wake caused by the time-averaging simplification of the disk. Given the savings in numerical costs of 85% versus the fully-resolved propeller setup, the actuator disk presents a viable alternative with an acceptable accuracy, the authors conclude. Multiple other publications report similar findings regarding the actuator disk's modeling error and performance [31, 32, 33].

A recent study by Schollenberger et al. investigated the effect of different distributed propeller configurations on aerodynamic performance using a common 19-seater regional aircraft for reference [34]. In particular, the research examined three types of configurations with a varying number of propellers: partial distributed propulsion (partial DP), full distributed propulsion (full DP), and wingtip mounted propellers (WTP). The number of propellers ranged between one for the WTP and seven for a full DP configuration. Regarding partial DP case, the positions were concentrated towards the outer wing, starting at the wing's tip. To obtain their results, the authors conducted a CFD analysis using DLR's flow solver *TAU*. Their setup involved multiple RANS simulations employing *TAU's* actuator disk model. A previous trimming iteration established the force equilibrium needed for steady boundary conditions. For the analysis a hybrid grid was used with cell numbers varying between 16 and 24 million cells, depending on the number of propellers involved. In terms of geometry, their grid consisted of a half-sphere domain with a symmetry plane, which was divided in a number of subdomains. For example, the propeller wake regions were discretised with a structured subgrid to limit numerical diffusion as much as possible. The wing was resolved with 218 points in the spanwise direction, and the cell size in the quasi-structured grid was set to 2% of the chord length c, close to the vortex core it was set to 0.75% × c.

The authors' results revealed that a full DP configuration only achieves minimal efficiency improvements [34]. The small efficiency advantage gained from the augmented lift caused by the wake flow is diminished by the higher viscous drag on the wing caused by the same effect. Fewer, but larger propellers proved to be overall more efficient, especially when they are placed in strategic locations like the wingtip. The biggest gains were achieved with the three propeller configuration (per half-span), with an additional 5% in aerodynamic efficiency and a reduction in required power of 2.9%. In general, the results demonstrated the effectiveness of wingtip mounted propellers in lowering induced drag by rotating in a direction opposite to the tip vortex (inboard-up). In an addition to their study, Schollenberger et al. refined their WTP-only configuration by optimising the propeller position and angle so that were able to achieve an efficiency increase of 5.6%, marking their highest measured gain.

Similar efficiency improvements were also found by Veldhuis who used the previously introduced VLM-model to optimise the wing of a Fokker 50 under consideration of propeller-wing interference. Among other parameters, the author optimised for the vertical propeller position, propeller angle of incidence as well as camber and twist distribution of the wing [4]. Veldhuis measured improvements in aerodynamic efficiency of up to 10 drag counts for cruise conditions. He states that a low-propeller position and a negative incidence angle, which reduces inflow distortions, are particularly beneficial. However, the source remains inconclusive about the isolated effect of the wing shape optimisation. A 2020 study by Chauhan and Martins, which specifically investigated aerodynamic shape optimisation of a propeller-wing configuration, found that the presence of the propeller had an insignificant effect on the optimisation result [35]. The authors used a gradient-based optimisation technique to optimise a total of 223 geometric parameters, controlling twist and shape of the wing, in conjunction with RANS simulations employing the actuator disk to model the propeller flow. However, their approach did not involve a concurrent optimisation of any propeller parameters.

Research publications about propeller-wing optimisation with the actuator disk using a multi-disciplinary approach (CFD-CSM) are not known to the author.

1.3. Research Objective

The overarching objective of this research is to expand the capabilities of the CFD-CSM (Computational Fluid Dynamics - Computational Structural Mechanics) coupled analysis framework existing at the German Aerospace Center (DLR) through an integration of the actuator disk (AD) modeling technology. The AD model is a well-established technique in CFD for efficiently simulating propeller aerodynamics. While it has already been implemented in DLR's CFD solver *TAU*, the proprietary *FlowSimulator* software suite, a framework for multi-disciplinary analysis and optimisation (MDA/O), has so far lacked an appropriate interface for *TAU's* AD model, barring ADs from being used in coupled CFD-CSM analyses. In the face of upcoming challenges related to a transition from fossil fuels to renewable energy sources, the development of highly-efficient aircraft designs will become ever more important, and capable and robust numerical tools play a central role in the search for these new designs. Therefore, by integrating the AD model this work establishes the basic functionality needed for a future expansion of the MDA/O toolchain, enhancing DLR's capabilities for analysis and optimisation of propeller aircraft.

The need for reducing aviation's greenhouse gas emissions has laid ever more emphasis on aircraft efficiency considering the significantly lower energy densities of renewable fuel alternatives as compared to conventional kerosene. This is why, in recent years, propeller aircraft have seen rising attention as propellers are generally able to provide high propulsive efficiencies. Additionally, the advent of electrification in aviation has opened up new possibilities for the design of emission-free aircraft as electrically-driven propellers can be employed in novel propulsion architectures, i.e. distributed electric propulsion (DEP). DEP promises efficiency gains emerging from a synergistic interaction between the wing and the propeller flow. However, propeller-wing interaction is a complex phenomenon, as prior research has shown (see Section 1.2). The literature review indicates both the potential performance benefits as well as the challenges for the design of modern propeller aircraft.

High-fidelity Reynolds-Averaged Navier Stokes (RANS) techniques offer an efficient modeling approach for aircraft design, facilitating the identification and analysis of conceptual designs in a cost-effective manner. Advances in materials and manufacturing technologies have enabled the conception of more aerodynamic, slender wings, yet these designs increasingly challenge the limits of monodisciplinary techniques. As more flexible wing designs emerge, the influence of aeroelastic effects grows, causing mono-disciplinary CFD analysis, which is limited to rigid geometries, to produce less realistic results. This is particularly relevant to multi-propeller aircraft, where the propellers' thrust and weight impose additional loads on the wing, increasing its capacity to deflect. This highlights the necessity for advanced simulation and optimisation techniques from the field of MDA/O.

The EU-funded research project HERA seeks to develop a new regional propeller aircraft by 2035, with the specific target of reducing emissions by 50%. In addition to other activities, the project's scope includes the advancement of modeling techniques for propeller aircraft, enabling requisite trade-off studies and design optimisation. The *FlowSimulator* suite represents a versatile, multidisciplinary simulation framework that incorporates a range of effective methods for coupled CFD-CSM analyses. However, the actuator disk model has not been integrated into this framework. Prior research has demonstrated the suitability of the actuator disk model as a tool for emulating propeller flow in high-fidelity CFD simulations. It provides a flexible and cost-effective alternative to fully resolved propeller simulations while offering greater accuracy than low-fidelity approaches. These features make the actuator disk model a useful tool in preliminary design stages and for optimisation problems with large numbers of design variables. Thus, by integrating the actuator disk into the coupled CFD-CSM analysis framework, this thesis contributes to the HERA project and its overarching goals. To achieve this, the methodology employed in this work entails the following tasks:

Literature Review and Theoretical Framework: A review of existing literature on propeller-wing aerodynamics (Section 1.2), CFD-CSM coupled simulations and the theory of actuator disks (Chapter 2) is conducted, providing a theoretical foundation for this work.

Test Case Definition and Toolchain Setup: A defined test case derived from the conceptual aircraft design developed in the HERA project is used to setup a CFD-CSM toolchain, introducing the applied modeling tools required for CAD, CFD mesh generation and CFD-CSM coupling and detailing the automation of this process (see Chapter 3).

Actuator Disk Integration: Employing the actuator disk model on the defined test case, Chapter 4 outlines the emerged issues and discusses the implemented solutions required for integrating the actuator disk technology in coupled CFD-CSM simulations. Additionally, the robustness of the implementation is assessed.

Analysis and Results: In Chapter 5, a parameter variation of important wing geometry and propeller design variables is performed and analysed with high-fidelity CFD-CSM simulations to evaluate the global effects of the parameters and quantify the sensitivities on aerodynamic performance metrics. The simulation results are analysed and assessed using adequate aerodynamic performance metrics. The benefits of the multi-disciplinary approach are evaluated by comparing the results to a reference case obtained from a monodisciplinary CFD simulation of a hypothetically rigid geometry.

In the final chapter (Chapter 6), the study presents a critical assessment of the work conducted, suggests improvements, and outlines a potential future course of action.

2. Theory

This chapter covers the theoretical foundations associated to the research question of this thesis. It details the theoretical background behind coupled simulations that combine Computational Fluid Dynamics and Computational Structural Mechanics (CFD-CSM), the basics of propeller aerodynamics and the theory underlying actuator disk model. In doing so, this chapter also implicitly establishes some of the requirements which have to be fulfilled by the CFD-CSM toolchain.

2.1. CFD-CSM Coupled Simulations

The aerodynamic forces exerted on an aircraft are determined by the shape of the surfaces that are exposed to the flow of air. As with all physical objects, though, an airframe is an inherently flexible structure that will deform under the influence of aerodynamic loads, which in turn changes the flow of the surrounding air. This reciprocal interaction is the subject of study in the field of aeroelasticity. Aeroelasticity is the term used to describe all phenomena that emerge from the interaction of aerodynamic, structural-elastic and, in some cases, inertial effects. It can be classified into two principal categories: static and dynamic. Static aeroelasticity describes phenomena that occur after a static equilibrium state has been reached, at which point all inertial forces have decayed. Static aeroelastic problems include, for example, the variations in wing load distributions resulting from the wing's deflection. In particular, swept wings display a coupled bending-twist behaviour that has a significant impact on the distribution of aerodynamic forces, as the local angle of attack is modified by the wing's deflection. Moreover, excessive wing twist can also result in a reduction of control surface effectiveness due to an insufficient torsional stiffness of the wing. In contrast to static aeroelastic problems, which exclude acceleration-induced forces, dynamic aeroelastic problems additionally involve inertial forces, covering phenomena such as flutter, which is critical to the design of aircraft, as it represents a significant risk for the structural integrity of the vehicle.

Over time, the available margins for improvements of aerodynamic efficiency have become narrower. To further enhance aircraft efficiency and fully exploit the potential of modern lightweight materials, the design of wings is likely to become more slender and flexible, thereby increasing the influence of aeroelastic effects. Computational fluid dynamics (CFD) solvers have long been established tools in aircraft development. However, they are only capable of simulating rigid aerodynamic shapes, thereby neglecting aeroelasticity. In order to account for these effects, multi-disciplinary simulation techniques have been developed, enabling CFD solvers to be coupled with computational structural mechanics (CSM) solvers. In addition to a more realistic representation of the aerodynamic phenomena, coupled CFD-CSM analysis facilitates the identification of technically feasible designs through the integration of the structural mechanics domain. Similarly, a multidisciplinary approach is essential for aerodynamic shape optimisation, as any effective wing design requires a compromise between fluid dynamics and structural mechanics. While coupled CFD-CSM analysis can also be employed to simulate dynamic aeroelasticity, this work only investigates static phenomena, focusing on the wing load distribution. Such phenomena can be analysed with the help of a steady CFD-CSM procedure.

A steady CFD-CSM loop, illustrated in the flowchart in Fig. 2.1, can be divided into the following stages: The aerodynamic forces obtained from a CFD solution for the initially undeformed geometry are projected to translate the aerodynamic surface loads from the CFD mesh to structural loads in the CSM domain. A structural model is employed by the CSM solver to calculate a displacement solution that corresponds to the input loads. Prior to the projection of the displacements to the fluid domain, the structural solution is typically multiplied by a global relaxation factor which collectively scales down the displacement values. Relaxation can improve the convergence characteristics by reducing the risk of overshooting. This may not only accelerate the convergence rate but also helps to avoid excessive deformations, making the process more robust. Next, the displacements are transferred back to their respective CFD surfaces, employing an operation that is inverse to the displacement projection method. The surface displacements of the CFD mesh are then input into a CFD volume mesh deformation algorithm, which allows the solver to approximate a new flow solution based on the deformed geometry. Thereby, the CFD-CSM loop is completed. By comparing the new loads and displacements to those from prior iterations, a convergence criterion is evaluated which will eventually halt the process if no other stopping condition has been reached beforehand.

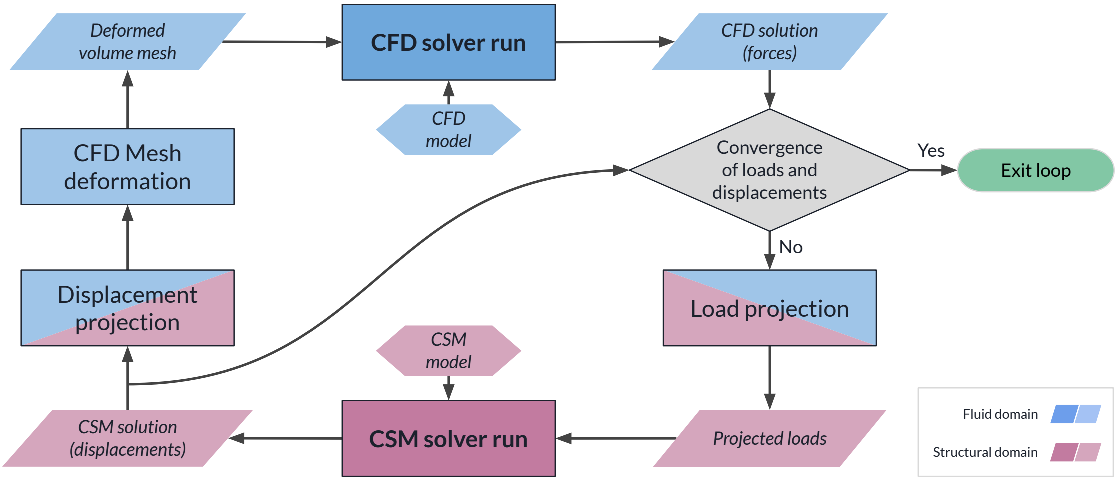


Figure 2.1.: Flowchart of a steady CFD-CSM cycle

Successfully conducting a multi-disciplinary analysis (MDA) requires a variety of software components, potentially requiring differing data structures, to interact seamlessly with each other, preferably in a time-efficient manner. For this purpose, the *FlowSimulator* (FS) environment has been jointly developed by the *German Aerospace Centre* (DLR), *Airbus* and *Onera*. The *FlowSimulator* suite provides a framework that aims to integrate the applications necessary for running multi-disciplinary simulations. The software is structured in layers, as illustrated in Fig. 2.2. The data layer at the core of *FlowSimulator's* architecture, named *FSDataManager* (FSDM), provides interoperable data-containers and basic functionalities for data handling, such as mesh selection tools. FSDM is designed for fast in-memory data exchange and optimised for parallelised operations on HPC (high-performance computing) systems. To allow for high computational speeds, FSDM is written in C++. One level up is the plugin layer: It encompasses an (open-ended) collection of partially independent software components with interfaces to FSDM which have to work cooperatively during an MDA simulation, each providing a distinct functionality. This includes the CFD or CSM solvers as well as plugins for trimming, mesh deformation etc. A central aspect of *FlowSimulator's* design philosophy is that no two applications of the plugin layer can exchange data between each other; all communication has to pass through the data layer. This allows for a better structured flow of information and minimises the risk of unwanted interdependencies between the applications. Additionally, this design is intended to enhance the exchangeability and maintainability of individual components. The third and last layer is the control layer, which provides an access point for the user to interact with the framework and control its operation. The control layer is written in *Python* as to maximise its flexibility and accessibility. To harmonise the different coding languages, a *Python* wrapper is implemented for FSDM, allowing for direct user interaction with *Python* scripts in the control layer. In order to further facilitate the setup of multi-disciplinary simulations, a software package named *FSDLRControl* has been developed at DLR [36]. It provides a collection of pre-defined scenarios for recurring use cases,

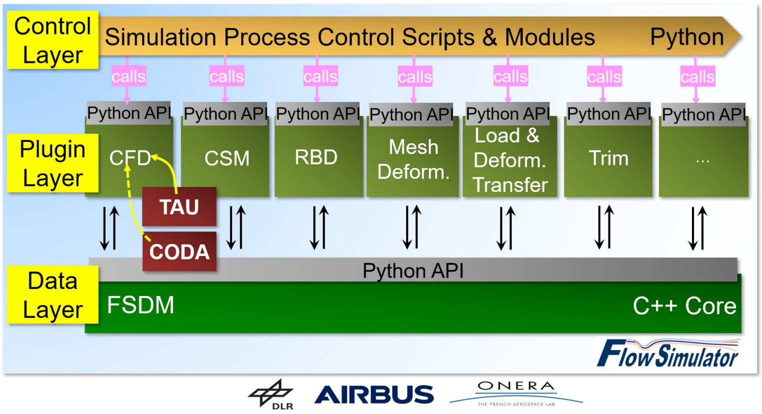


Figure 2.2.: Software architecture of the FlowSimulator suite; from Reimer et al. (2021) [36] (with courtesy of the author)

including one for steady CFD-CSM analyses. Through a input parameter file, the user can conveniently select a scenario, define relevant settings for the coupled simulation and determine which solver implementation is used. For each involved discipline, *FlowSimulator* incorporates a selection of solvers, which are attributed to the plugin layer. The following subsections describe the solvers that are the most important and relevant to this work.

2.1.1. CFD and CSM Solvers

The CFD and CSM solvers represent the functional core of the *FlowSimulator* environment, numerically approximating the underlying physics problem. One choice of possible CFD solvers to be used in the environment is *TAU*[37]. The CFD code, which has been developed by the DLR itself, is a RANS-solver that uses a node-centered dual-grid metric, optimised for unstructured CFD grids. *TAU* is a highly validated tool with application in research and industry contexts and also the CFD solver used for the analyses conducted within the scope this work.

In order to compute structural deformations, the commercial CSM solver *MSC NASTRAN* can be used within the framework. It is a versatile and well-established structural mechanics solver. *NASTRAN* can compute static and dynamic problems employing linear as well as non-linear methods. Technically, the application does not belong to the *FlowSimulator* environment, making it an external plugin [38]. For this particular case, a custom interface module, *FSNastranInterface*, has been implemented to convert data between FS and *NASTRAN*. Since the structural mechanics domain typically involves considerably fewer data points compared to the CFD domain, this file-based data exchange has no significant impact on the computational speed of the whole process [39].

2.1.2. Load-Deformation Transfer

In order to integrate the CFD and the CSM solvers in a unified CFD-CSM loop, their respective outputs have to be mapped to inputs of the other domain. The requirements for CFD and CSM meshes differ vastly from each other. This is why these meshes usually vary considerably in size but also in terms of their geometry. Thus, a projection method has to be applied for interpolating data from one domain to the other, as illustrated in Fig. 2.3.

For reasons of physical validity, the method has to be conservative with regard to the total forces being projected and the energy, or work, applied to the system. For transferring displacements from the structural domain (CSM) to the flow domain (CFD), the interpolation problem can be expressed in this form [40]:

$$\mathbf{U}_{CFD} = \mathbf{H}\mathbf{U}_{CSM} \tag{2.1}$$

where $\mathbf{U}_{CFD}$ and $\mathbf{U}_{CSM}$ denote generalised displacement vectors in the respective domain and $\mathbf{H}$ represents the so-called coupling or projection matrix. Conservation of energy is

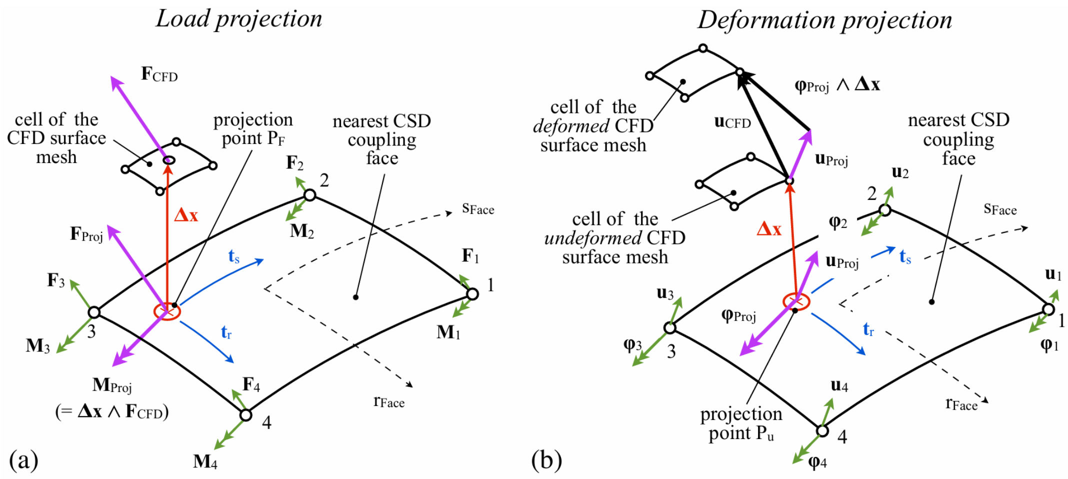


Figure 2.3.: Finite Interpolation Element approach: a) load projection (left) with forces **F**, moments **M** and geometric offset $\mathbf{\Delta}x$, b) displacement projection (right) with displacements **u** and rotations $\boldsymbol{\varphi}$, tangential vectors **t**; from Reimer (2020) [40] (with courtesy of the author)

guaranteed if Eq. 2.2 is simultaneously true:

$$\mathbf{F}_{CSM} = \mathbf{H}^T \mathbf{F}_{CFD} \tag{2.2}$$

One of the well-established methods applied for conducting a load- or displacement-transfer are *finite interpolation elements* (FIE). They make use of the same shape functions that are employed by the *Finite Element Method* to interpolate quantities based on a direct node-to-cell mapping. Although, this method usually produces good interpolation results regarding the transfer of loads, it sometimes fails to recreate sufficiently smooth displacement field projections, considering the in parts very high accuracy requirements of CFD meshes. In these cases, spline-based approaches, such as the moving-least squares method (MLS), can present a valid alternative since these methods are inherently capable of generating continuous displacement fields. With MLS, a concatenation of low-order polynomials, one defined for each CFD surface node, are fitted to the displacement solution of the structural solver. For each polynomial fitting, the input displacements are weighted using a so-called radial basis function (RBF) with local support. By using this type of function, only a finite number of local input nodes inside a given support radius are considered for fitting the polynomial coefficients, as illustrated in Fig. 2.4. During the setup of a coupled simulation with *FSDLRControl*, the number of considered nodes can be specified by a parameter called *numNextCenter*. A larger parameter value generally leads to a smoother projection field, as more input terms are incorporated, but this also increases computational cost. Generally, the goal is to find a compromise that leads to a sufficiently smooth projection result while preserving locality of the input displacements.

Both of the presented methods, FIE and MLS, are implemented in *FlowSimulator* via an additional software module[1] [41] which can also be evoked by the aforementioned *FSDLR-Control* suite.

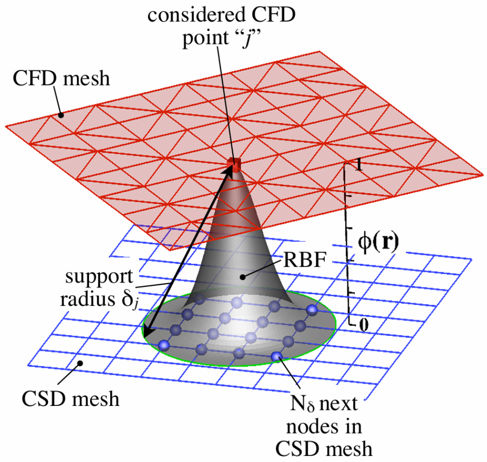

considered CFD point "*j*"
CFD mesh
1
$\phi(\mathbf{r})$
RBF
support radius δ_j
0
CSD mesh
N_δ next nodes in CSD mesh

Figure 2.4.: Schematic of the moving least-squares technique (MLS); from Reimer (2020) [40] (with courtesy of the author)

More complex geometries, e.g. complete aircraft models, can still cause problems for the projection process. Especially at the intersections of different structural groups, such as wing and fuselage, a primitive displacement projection approach might create discontinuities or holes [42]. For this reason, a component-based application of projection methods can be used. Within so-called coupling definitions, subsets of the respective meshes can be selectively paired in such a way that the loads and displacements are only transferred in between these two components. This also allows for an individual adjustment of projection parameters for each coupling definition. A blending function is applied at the intersections to ensure a smooth transition from one component to the next. Multi-component blending provides an adaptable and user-friendly solution for projecting loads and displacement on complex geometries. High-quality projection results are essential as, for example, the deformed surface geometry is used as input for the subsequent volume mesh deformation.

2.1.3. Mesh Deformation

During a coupled CFD-CSM simulation, the mesh deformation operation takes place after a displacement solution has been obtained from the structural solver and interpolated to nodes on an associated exterior surface of the CFD mesh. Then, the mesh deformation algorithm is tasked with determining an adequate displacement value for all remaining nodes of the mesh, using the interpolated structural response as a boundary condition. According to Reimer [40], typical requirements for such an algorithm are:

[1] FSDLRCFDCSMInterpolSuite

- Computational efficiency: The computational cost per run should be low since many runs of the mesh deformation algorithm are potentially necessary during a single coupled CFD-CSM loop.
- Robustness: The procedure has to preserve mesh quality and avoid excessive distortions to ensure the CFD solver's numerical stability and solution accuracy.
- Mesh independence: The algorithm should function independently of the underlying mesh type, e.g, structured or unstructured.
- Configuration independence: The mesh deformer should be able to operate with minimal user interaction regardless of the underlying geometric configuration of aerodynamic surfaces, e.g. wings, pylons etc.

Over time, several different approaches have been elaborated to accomplish mesh deformation. In general, they can be categorised into four principal classes: 1. algebraic; 2. spline-based techniques; 3. methods based on partial-differential equations (PDE) as well as 4. hybrids of the previously named classes [40]. Some of the most prominent methods in the PDE-based group are the so-called physical analogy techniques. FlowSimulator features the so-called elasticity analogy method alongside with the spline-based RBF (radial basis function) techniques, which are therefore described in the following in greater detail. They are implemented in the module *FSDeformation* (RBF) [43] or *FSMeshDeformation* [44], respectively.

Spline-based Mesh Deformation

According to Reimer, RBF methods have gained popularity because they are simple to implement and parallelise, fast to evaluate and retain a high mesh quality even for large displacements [40]. A commonality of all spline-based mesh deformation methods is that they approximate an objective function (in this case the displacement field's projection from the CSM coupling mesh to the CFD surface mesh nodes) using a spline as an approximant which is defined by an input vector of base or centre points [40]. The spline s is typically defined as a linear combination of radial basis functions ϕ. Given the function values f_i (e.g. displacements), the approximant can be found by solving Eq. 2.3 for the unknown coefficients α_i and β_i (after de Boer [45]):

$$s(\mathbf{x}) = \alpha_i \cdot \mathbf{p}\left(\mathbf{x}\right)^T + \sum_{i=1}^{n} \beta_i \cdot \phi\left(||\mathbf{x} - \mathbf{x}_i||\right) \tag{2.3}$$

where

$\mathbf{p}\left(\mathbf{x}\right)^T$: low-order linear polynomial

$\mathbf{x}_i$: base points

The first term of Eq. 2.3 consists of globally defined linear polynomials $\mathbf{p}(\mathbf{x})^T$ that are additionally required as they allow for a representation of rigid-body transformations by the approximant, which would otherwise only be possible if the RBFs are positive-definite [40]. The second term includes the locally defined radial basis functions ϕ. RBFs are functions with only one argument, a distance metric, which is typically simply the Euclidean norm $||x||_2 = \sqrt{\sum_{i=1}^{d} x_i^2}$ [46]. Thus, they are characterised by a radial symmetry. A variety of radial basis function ϕ exists which can be split into two groups: RBFs with local and RBFs with global support, depending on whether or not they approach zero for increasing values of the argument r. For the problem at hand, non-local RBFs, such as the thin-plate spline or the cubic volume spline, have proven to result in higher mesh qualities [40]. Additionally, the approximant is multiplied with a blending function which depends on the wall-distance of the input coordinate. A crucial aspect of the RBF method's implementation in *FSDeformation* is a step called centre point reduction. It reduces the number of CFD nodes which are considered for the spline approximation to an evenly-spaced subset of surface nodes. Since CFD meshes usually have a very high number of surface nodes, this is necessary to expedite the mesh deformation process. Consequently, the resulting spline can only represent the aforesaid subset of nodes exactly; the remaining surface nodes are merely approximated. The alternative mesh deformation approach (Elasticity Analogy method), on the other hand, does not require an approximation. The structural mechanics analogy approach, which the EA method belongs to, is introduced in the following section.

Structural Mechanics Analogy-based Mesh Deformation

Physical analogy-based deformation is based on the insight that a mesh can be represented as a solid body that deforms according to its mechanical properties. Finding a the displacement values for the interior nodes of a volume mesh therefore transforms into a problem of solving structural mechanics equations. Since this requires the inversion of potentially large stiffness matrices, physical analogy methods can be computationally demanding. On the other hand, the physical analogy deformation is a robust approach, usually producing deformed meshes of high-quality. The implementation available in *FlowSimulator* via the *FSMeshDeformation* library, provides a linear and a non-linear variant of the so-called *Elasticity Analogy* (EA) method. A disadvantage of the implemented EA method is that it can only be used on unstructured meshes. Additionally, because of the higher computational cost, the intended usage of the EA method in the context of *FSDLRControl* is typically limited to repair procedures on those mesh regions for which the RBF approach has failed. In this work, however, it is used exclusively to deform the entire CFD mesh. As mentioned before, EA renders exact representations of the entire displacement field, whereas the RBF method only yields an approximation. This choice is made to accommodate for the requirements of the actuator disk model (see Sec. 4.1 for a detailed explanation).

2.2. Propeller Aerodynamics

Propellers are one of the most widely-used methods for thrust generation in aviation. They present a very efficient mean of propulsion for aircraft operating below the transsonic flight regime [47]. Propellers generate aerodynamic forces similar to a wing, albeit through a rotary motion instead of a translational one which has implications for their interaction with the surrounding flow.

Geometric Considerations

One of the defining features of a propeller is its rotation. Given an angular velocity Ω, the rotational velocity can be found with Eq. 2.4:

$$u_{rot}(r) = \Omega \cdot r \tag{2.4}$$

where r denotes the radial coordinate. Considering the propeller moving through the air at an onflow velocity u_∞ parallel to its axis of rotation, Eq. 2.5 gives the resultant velocity flow relative to the blade as:

$$u_{rel} = \sqrt{u_{rot}^2 + u_\infty^2} \tag{2.5}$$

The blade's local Mach number is an important figure because a propeller's performance generally deteriorates the closer it operates to the speed of sound, as losses from compression and sonic shocks increase [5]. Combining Eq. 2.4 and 2.5, the highest local Mach number, found at the propeller's tip, Ma_{tip} can be obtained with Eq. 2.6 by inserting the propeller radius r_{Prop} as the radial coordinate:

$$Ma_{tip} = \frac{u_{tip}}{c} = \frac{1}{c} \cdot \sqrt{\left(\Omega \cdot r_{Prop}\right)^2 + u_\infty^2} \tag{2.6}$$

where

u_{tip} : relative propeller velocity at tip

c : speed of sound

r_{Prop} : propeller radius

Analogous to other airfoils, propeller blades generate an aerodynamic force depending on their angle of attack. Due to their rotary motion, the angle of attack, and thus the relative velocity, is directly affected by the propeller's rotation speed. Since the rotational velocity and therefore the angle of the incoming flow also depends on the radial position, most propeller designs exhibit a pitch angle variation along the radius. With such a twisted blade, a favourable load distribution can be achieved. Figure 2.5 shows a blade section's angle of attack α as a function of the blade pitch β and the apparent inflow (or helix) angle ϕ, which results from a combination of the relative velocity V_R and the induced flow w.

This apparent velocity component is a result of the propeller forces acting on the fluid and it leads to a slightly reduced (effective) angle of attack. The angles depicted in Fig. 2.5 are denoted with reference to the airfoil's zero-lift angle α_{ZL} instead of the typically used chord.

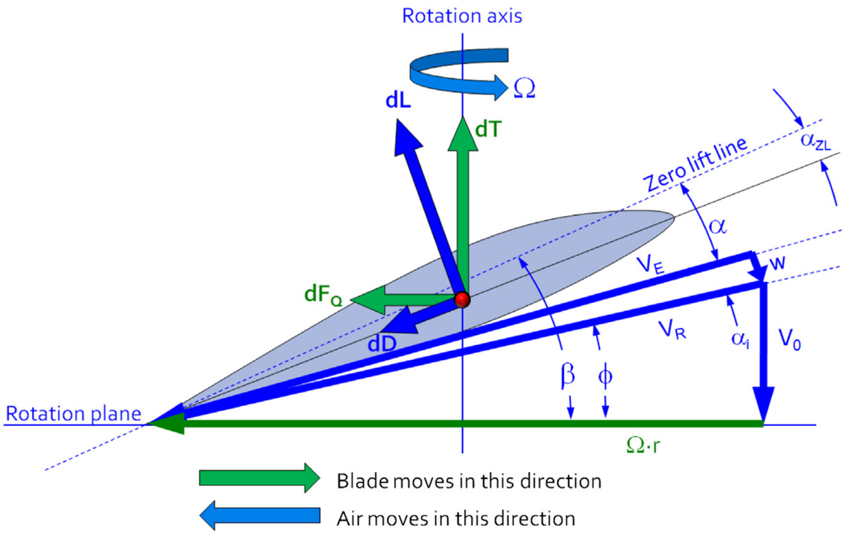


Figure 2.5.: Propeller blade: flow conditions and sectional forces at an airspeed $u_\infty = V_0$; by Gudmundsson (2014) [5]

Blade Element Theory

Blade Element Theory (BET) provides a method to approximate propeller loads, namely its thrust T and its torque Q, through the integration of forces acting on infinitesimal radial sections dr. The thrust T is defined as a force acting along and the torque Q as the moment around the propeller axis (see also Fig. 2.5). To obtain these load quantities, BET uses lift and drag coefficients of the cross-sectional airfoils defining the propeller blade's shape to first calculate the aerodynamic forces. In practice, these coefficients can e.g. be obtained by measurements or low-fidelity methods. Following BET, the differential lift dL and drag dD are given by Eq. 2.7 and 2.8:

$$dL = \frac{1}{2}\rho u_{eff}^2 \cdot c(r) \cdot C_l(\alpha_{eff}, r) \cdot dr \tag{2.7}$$

$$dD = \frac{1}{2}\rho u_{eff}^2 \cdot c(r) \cdot C_d(\alpha_{eff}, r) \cdot dr \tag{2.8}$$

where

ρ :	air density
u_{eff} :	effective air speed
$c(r)$:	chord length at radial coordinate r
C_l :	lift coefficient of blade section
C_d :	drag coefficient of blade section
α_{eff} :	effective angle of attack

Through a rotational transformation, lift and drag can be converted into the propeller-related thrust and torque components. As previously mentioned, the effective angle of attack α_{eff} and the closely related effective velocity v_{eff} are quantities emerging from the propeller's own induction flow.

Propeller efficiency

Together with contributions from viscous drag, the induced flow diminishes the propulsive efficiency of a propeller. The propeller efficiency is commonly defined as the ratio between propulsive power P_{prop} and engine power output P_{engine} [5]:

$$\eta_P = \frac{P_{prop}}{P_{engine}} = \frac{TV}{P_{engine}} \tag{2.9}$$

With the non-dimensionalised quantities C_P (coefficient of power), C_T (coefficient of thrust) and the so-called advance ratio J:

$$C_P = \frac{P}{\rho n^3 D^5} \tag{2.10}$$

$$C_T = \frac{T}{\rho n^2 D^4} \tag{2.11}$$

$$J = \frac{U_\infty}{nD} \tag{2.12}$$

where

n :	propeller speed (revolutions per time interval)
D :	propeller diameter

Eq. 2.9 can also be rewritten as:

$$\eta_P = J\frac{C_T}{C_P} \tag{2.13}$$

A well-known technique for calculating the induced velocity of a propeller is the actuator disk model, which is based on the principles of momentum theory.

2.3. Actuator Disk Model

The actuator disk is a widely-applied technique in CFD analyses employed for efficiently simulating rotors flows, as it allows for the complex processes involved in rotor aerodynamics to be simplified to a boundary condition on a two dimensional disk surface. Through time-averaging, the disk model reduces the transient process to a problem which can be solved in a steady RANS simulation.

2.3.1. Theoretical Foundation

The actuator disk model is grounded in the theoretical framework of classical momentum theory. The theory was developed over the course of multiple decades in the late 19th and early 20th centuries through the contributions of numerous renowned researchers, including Rankine, Froud, Joukowski, and Betz, among others [48]. In its most basic form, the one-dimensional formulation of momentum theory reduces the interaction between a propeller and the surrounding flow domain to its primary force, namely thrust. The force is assumed to be uniformly distributed across a flat, permeable disk surface, illustrated in Fig. 2.6. This allows the interaction of the propeller blades to be emulated as a discontinuous step in the pressure field, shown in Fig. 2.7. Since the resulting velocity profile remains smooth, the theory does not violate the continuity condition (inherent to continuum mechanics).

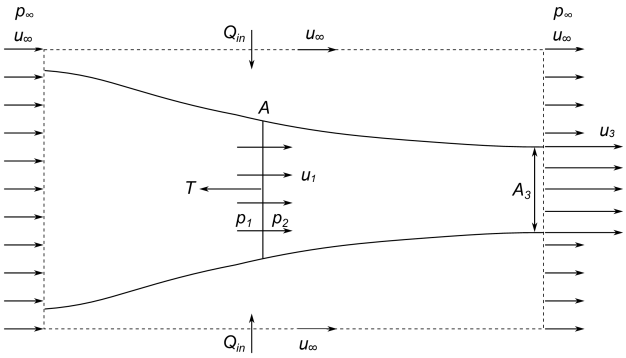


Figure 2.6.: Schematic of an actuator disk with relevant flow quantities; by Deters (2015) [49]

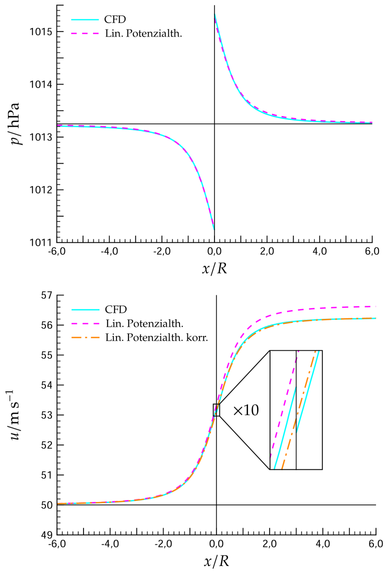


Figure 2.7.: Distribution of flow quantities: pressure p (top) and flow velocity u (bottom); by Raichle (2017) [6]

Furthermore, momentum theory relies on the assumptions of a inviscid, non-compressible fluid, and, in the case of the one-dimensional formulation, symmetry in radial direction. With that, it is possible to set up the momentum balance of the field which, when simplified, gives rise to the following relationship for the thrust T [48]:

$$T = \Delta p_1 \cdot A = \rho \cdot \int_A u_3 \cdot (u_3 - u_\infty)\, dA \tag{2.14}$$

with pressure differential Δp_d, cross-sectional disk area A_d, density ρ, wake velocity u_3 and farfield velocity u_∞. Since the pressure increment Δp_d as well as the wake velocity u_3 are uniformly distributed Eq. 2.14 can be rewritten as [48]:

$$T = \Delta p_1 \cdot A = \rho \cdot u_3 \cdot (u_3 - u_\infty) \cdot A_3 \tag{2.15}$$

Together with the combined Bernoulli equations from the two parts of a single streamline (upstream and downstream of the disk) Eq. 2.15 simplifies to the well-known equation first conceived by Froude in 1889 [48]:

$$\overline{u}_1 = \frac{1}{2}\,(u_3 + u_\infty) \tag{2.16}$$

With the obtained average disk velocity $\overline{u}_1$ and the disk's surface area A, which is assumed to be known, it is then possible to compute the power exchanged between the disk and the flow as $P = T \cdot \overline{u}_1$. The induced velocity w is commonly defined as: $w = \overline{u}_1 - u_\infty = u_3 - \overline{u}_1$ [48].

One crucial feature present in real-world propeller flows which is not yet covered by the one-dimensional formulation is swirl. In 1920, Joukowski was able to expand the actuator disk model by implementing propeller torque defined as a constant circulation term. Thereby, the Joukowski-disk, also referred to as two-dimensional momentum theory, is able to represent the flow of a propeller rotating at a constant rate under axially uniform, parallel inflow conditions. Still, only with the additions contributed by Glauert in 1935, the actuator disk model was able to become the versatile tool for engineering problems that it is today, as he amalgamated momentum theory and blade-element theory. With this so-called *Blade Element Momentum* method (BEM) the forces imparted on the flow by the propeller can be derived from lift and drag components of a virtual 2D airfoil. This allows propeller data in the form of aerodynamic coefficients, which might be obtained from measurements or some virtual propeller design tool, to be used as input to an actuator disk's boundary condition in a CFD solver.

2.3.2. Implementation in the CFD solver TAU

Because of its versatility, the actuator disk (AD) has been incorporated in several established CFD solvers. The following section describes exemplarily how the model is implemented in DLR's code *TAU*, since it is the solver used exclusively for the simulations in this work.

According to its user guide, the actuator disk model in *TAU* is derived from classical momentum theory, albeit in an adapted formulation in order to ensure conservative fluxes while accounting for compressibility and swirl flow [50]. To impose a virtual propeller force, a momentum source term is introduced to the cells constituting the disk surface. The unknown quantity of each cell's source term can be computed in two ways:

1. The user has the option to prescribe a load directly by inputting sectional loads in radial direction which are integrated along the circumferential direction. Thus, a momentum source term is assigned to each cell of the AD.

2. The resulting propeller forces are calculated from *Blade Element Theory* (BET). For this, geometric and aerodynamic propeller data have to be defined in *TAU's* user input files. The geometric data consists, among other variables, of the propeller's chord length and twist distribution. The aerodynamic input holds data tables of lift and drag coefficients for an arbitrary number of angles of attack. At runtime, the local flow conditions, inflow velocity and angle, are iteratively determined for each cell. Together with a preset rotational speed of the propeller, BET is employed to determine the resulting propeller forces.

At the level of the CFD grid, the actuator disk is defined by two congruent boundaries, the *actuator inlet* and the *actuator exhaust*. The circular boundaries consist of pairs of connected nodes which are topologically different but geometrically identical. Thereby, the solver is able to handle the required discontinuity immanent to the AD model. For reasons of topological and algorithmic consistency with the pre-existing *TAU* code, the two boundaries are implemented as a pair of periodic boundaries enclosing a degenerated volume (see Fig. 2.8).

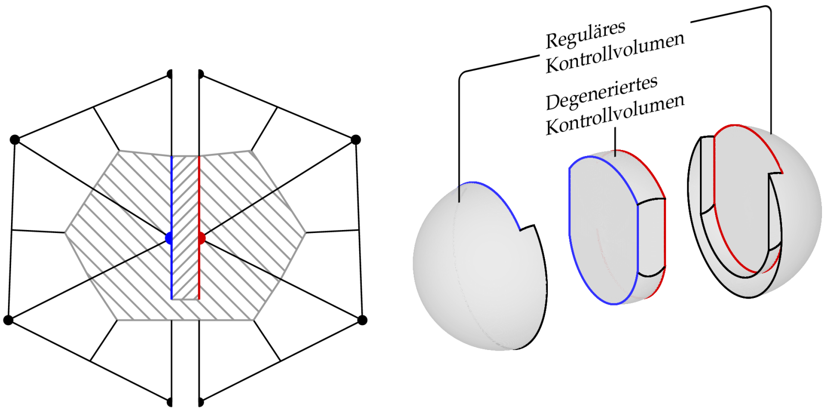


Figure 2.8.: 2D section (left) and 3D representation (right) of the actuator disk's topology in TAU; by Raichle (2017) [6]

In the case of *TAU*, the CFD grid file has to contain additional information for the AD model to operate. This AD meta-data holds lists of the associated nodes on the inlet and exhaust boundaries as well as information about the disk geometry. This information is required to translate quantities from the normalised, non-dimensional form calculated by the model to the geometry of the CFD mesh and vice versa. The geometric data is comprised of the inner and outer disk radii, the disk's normal vector, and its centre point. This data can be appended to the mesh with the help of the utility programme *setup_taugrid*, provided in the *TAU* software package, or, if using the software ANSA, during mesh generation. [51] Further details regarding the implementation of the actuator disk can be found in [6].

3. Test Case Definition and Model Setup

This chapter outlines the preparatory work leading up to the coupled CFD-CSM analyses and introduces various software tools used to generate the models of the involved disciplines and automate the workflow. The chapter begins by detailing the origins of the utilised reference aircraft within the EU-project HERA (see also Sec. 1.1) and the adoption of this conceptual design as a test case for the present work. The following sections explain the design of the parametric-associative CAD geometry, the CFD mesh generation, and describe the structural model (CSM mesh) employed in the analysis. Thereby, this chapter establishes the components of the CFD-CSM toolchain required for setting up the coupled simulation. For each component in the chain, care was taken to ensure its robustness and adaptability as to its intended application in multi-disciplinary analysis. Additionally, the setup puts focus on the interchangeability of the employed models, allowing for future expansion and improvement of the toolchain.

3.1. Test Case Definition

The reference aircraft used for all subsequent investigations in this thesis is derived from a conceptual design, intended as a technical demonstrator, developed within the scope of the EU-project HERA (see also Sec. 1.1). The aim of the HERA project is to develop a hybrid-electric regional aircraft in the 70-seat capacity segment with a targeted range of approx. 500 nm. The concept has a defined maximum take-off weight (MTOW) of 32 t and a design cruise altitude of 25,000 ft. More specifically, the design variant adapted for this thesis, denoted as *Use Case B* (UCB), is defined as a high-wing plane in a six-propeller DEP-type (Distributed Electric Propulsion) configuration. It features a (combined) series/parallel hybrid architecture, meaning that on each side the innermost propeller is powered by a thermal engine exclusively, whereas the remaining two smaller outer propellers are driven by electric motors. The electric power is either provided by hydrogen fuel cells, batteries, or a generator coupled to the main thermal engine [52].

The test case definition attempts to replicate HERA's design specifications by adhering to the top-level aircraft requirements (TLAR) whenever possible. Although it being a preliminary design, the HERA aircraft provides a test environment with proportionally sized components and consistent design variables, enabling this work to emulate realistic operating conditions. Still, the test case primarily fulfills an exemplary function, as this work is focused on methods and aims to develop an adaptable toolchain, independent of the underlying use-case. Project members, primarily *Airbus Defence & Space*, contributed to this thesis by providing the structural wing model and the foundations for the wing's CAD geometry.

3.2. Model Parametrisation and Automatic Mesh Generation

3.2.1. CAD Model Generation

The software *CATIA V5* by *Dassault Systèmes* is used for the generation of the 3D CAD geometry in this work. The *CATIA* model inherited from *Airbus* included a parameterised surface model of a half-wing in cruise (or clean) configuration. In order to improve the model's design structure and update stability, it was decided to rebuilt it, keeping only the included airfoil and the model's parametrisation. The parametrisation comprises multiple input parameters, which can be driven directly, as well as so-called relations (i.e. formulas) connecting the input values to various dependent internal parameters. The most relevant design parameters for the wing along with their values, as specified by HERA's TLAR, are shown in Table 3.1; the resulting wing planform is shown in Figure 3.2. Additionally, the model includes a semispherical farfield surface with a radius $r_{farfield}$ measuring $100 \times c_{root}$. The farfield is connected to a circular symmetry plane surface at $y = 0$, which collectively form the outer bound of the flow domain (see Fig. 3.1). For the given parameters, the resulting mean aerodynamic chord c_{MAC} of the wing equates to 3,166.7 mm. An overview of the relations and its associated parameters can be found in Appendix A.

Input parameters:

parameter	*spec. value*
wing area S_{ref}	73 m²
aspect ratio $AR = b^2/S_{ref}$	10.0
rel. thickness t	0.18
taper ratio $\lambda = c_{tip}/c_{root}$	0.62
zero-sweep position $x_{0,sweep}$	55% $\cdot c$
tip twist angle ϵ_{tip}	-2°

Dependent parameters:

parameter	*res. value*
wing span b	27,018.5 mm
kink y-position y_{kink}	9,456.5 mm
root chord c_{root}	3,082.5 mm
tip chord c_{tip}	1,991.1 mm

Table 3.1.: Wing model CAD parameters

The wing design uses only a single profile which defines the wing's cross-section along the entire span. On the outer wing, from the kink position to the wing tip, the profile is

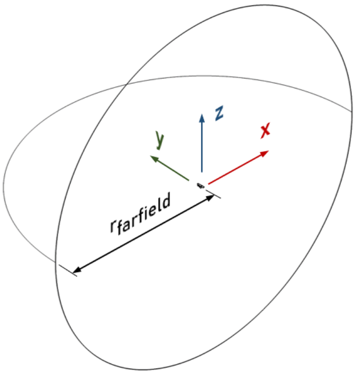


Figure 3.1.: Outer flow domain

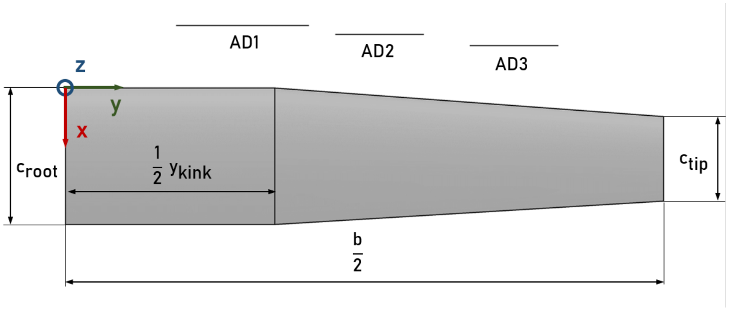


Figure 3.2.: Wing planform

scaled down progressively according to the defined aspect ratio, with the zero-sweep axis serving as reference. Additionally, the outer wing section is rotated around the 25%-chord axis by a linearly increasing twist angle. Since it is only applied from the kink outwards, the inner wing remains prismatic, independent of the tip twist angle. To represent the three propellers intended by the design, three disks are added to the CAD model forming the actuator disk boundaries. Each disk surface is parametrised in terms of its size (radius), position (x, y, z) and orientation (angle of incidence). With respect to the design specifications devised within the HERA project, the geometric propeller parameters were mostly still undetermined at the time the CAD model was created. So far, the specification documents only defined the spanwise position of the innermost propeller at 29.6%. However, the structural model already included nodes for the two outer propellers, positioning them at spanwise locations of 52.5% and 75.0% respectively. In alignment to these locations, table 3.2 shows the other geometric propeller positions defined for the baseline geometry:

parameter	*AD1*	*AD2*	*AD3*
radius	1.50 m	1.00 m	1.00 m
x-position w.r.t. LE[1]	-1.41 m	-1.38 m	-1.35 m
z-position w.r.t. LE[1]	-0.50 m	-0.30 m	-0.30 m
angle of incidence	0°	0°	0°

Table 3.2.: Geometric propeller parameters

[1] with respect to leading edge

To account for the preliminary stage of the design process and to reduce modeling complexity, it was decided to integrate neither propeller hubs nor nacelles to the CAD model at this point. Although a slight overprediction of propeller performance can be expected in the CFD analysis, the overall wing aerodynamics including propeller interference effects are still modeled with reasonable accuracy.

Automated Geometry Output

At DLR's Institute of Aerodynamics and Flow Technology a remote machine, named *CATFLOS*, has been set up which can be tasked with regenerating and outputting a CAD model without the need for manual intervention. To be compliant with *CATFLOS*, all parameters in the CAD model are incorporated in a design table. This allows them to be manipulated from outside the model, which is required for automated geometry updates.

However, parametric CAD models are prone to cause crashes or produce erroneous output when an unfavourable parameter combination is used, especially as parameter values are not input manually e.g. because they are automatically generated during a design optimisation. In order to assess the model's robustness, the parametrisation was tested by inputting parameter combinations for a wide range of values. The test confirmed that the CAD model is able to handle even extreme wing design variables without causing an update error. (For more details on these tests see below in Subsection 3.2.2.) Another potential source of errors, which might not be detected immediately, are intersections between a propeller disk and the wing surface or another propeller. This would obviously cause an error during the subsequent CFD run, if the mesh generation process had not already failed beforehand. However, the CAD software would most certainly still run error-free and produce a undesired geometry output. To be able to avoid or handle this type of error, two failure modes have to be distinguished. 1.) To prevent contact between two propeller disks, it suffices to enforce that the y-offset between their two centre points is larger than the sum of their radii. This constraint is simple to implement and is always going to create enough spatial separation to exclude an intersection. 2.) Precluding an unintended contact between a propeller and the wing is more complicated due to the irregular shape of the wing. Therefore, the *CATIA* model includes a fail-safe mechanism. In case of a propeller surface intersecting the volume enclosed by the wing, an update error is provoked by design. This would allow the error to be caught by an adequate error-handling method, stopping it from being passed further. Furthermore, the protective volume can be inflated with an arbitrary offset value, thereby creating a controllable perimeter.

3.2.2. CFD Grid Generation

Constructing CFD grids of high quality generally requires effort and expertise. In part, grid quality can be assessed easily with the help of geometrically defined metrics, such as skewness, aspect ratio or non-orthogonality, which most meshing programmes are able to compute immediately and optimise for. But above all, a mesh's quality is determined by its capacity to facilitate an accurate and fast converging CFD solution. However, here lies the central conflict in mesh generation: High resolving solutions demand a fine mesh, whereas the shortest computing time is achieved with the coarsest grid possible. Consequently, grid generation often necessitates an iterative approach which can involve manual interaction, complicating the automation of the process. This is also one reason, why mesh deformation can present a superior strategy for aerodynamic shape optimisation compared to mesh regeneration. A high-quality mesh can simply be deformed into the desired shape, which simultaneously saves the computational cost associated with regenerating a mesh. Since the mesh topology is unchanged, the subsequent CFD run can be started from the solution obtained in the previous cycle, adding to the cost benefit. However, this technique also has its limits, since excessive deformations can deteriorate the geometric mesh quality. So much so, that it can lead to slower convergence or solver errors. One can easily see how such issues could occur for the test case investigated in this study, when attempting to analyse different propeller arrangements. Due to the relative proximity between the propeller disks and the wing, even moderate variations of the propeller location would produce large geometric distortions of the intermediary cells. This is particularly bad, as these cells would be located in the region most affected by the propeller's induction flow. A high grid resolution and quality is paramount for accurately resolving the large gradients characterising this flow region. Consequently, a grid regeneration approach is the more suitable strategy for studying a wide variety of propeller-wing configurations.

Against this background, the meshing software *ANSA* (version 24) is chosen for the design of CFD grids in this work [51]. It offers a set of functionalities which considerably simplify the setup of an adaptable automatic mesh generation procedure. Apart from allowing for parallised processing, *ANSA* provides a *Python* wrapper module through which the programme can be controlled externally. Moreover, *ANSA's* internal workflow is also designed with automation in mind. Many of these functionalities are comprised in an environment called *Batch Meshing*. The *Batch Meshing* workflow involves a compilation of so-called scenarios customised for the different regions typically found in CFD meshes. The scenarios used for the mesh generation process in this case are: the *Meshing Scenario* for the surface meshing; the *Layer Scenario* for the boundary layer region; and the *Volume Scenario* for the remaining interior volume. Within each scenario, sessions can be defined for an arbitrary set of geometric entities (e.g. CAD surfaces). A session stores a collection of parameters, such as cell sizes, growth ratios, cell types and so on, which are applied to the geometry by automatic meshing algorithms to build up a particular part of the mesh. On execution, *Batch Meshing* sequentially processes the pre-defined sessions to successively construct a complete volume mesh. Once a collection of scenarios and sessions is established and its meshing parameters are tuned, the process can be applied also to a modified CAD geometry as long as the input remains topologically unchanged. An important functionality supporting the *Batch Meshing* process is *ANSA's* feature detection. Based on a number of selection criteria, geometric features, such as edges, holes etc., can be detected automatically. This facilitates the application of special treatments to critical regions, for instance the leading and trailing edges of a wing. To conclude, all of these features combined allow for a highly automated and efficient grid generation workflow. Using the aforementioned approach, a grid was first designed manually by iteratively adjusting parameters until a high-quality output with good convergence characteristics was obtained. The identified parameter settings were then reused for additional, automatically generated grids.

The grid of the baseline geometry is made up of 4.2 million nodes, constituting a total of 5 million cells (cell type breakdown in Table 3.3). The wing surfaces are covered by unstructured quadrilateral cells which have a maximum length of 60 mm, or 1.9% of the mean aerodynamic chord c_{MAC}. The wing's leading edge is discretised with 365 nodes. The trailing edge, which has a thickness ranging between 1.1 and 1.7 mm, is meshed with a structured surface grid. It proved to be advantageous to extend the structured cell region from the trailing edge to the thinnest part of the adjacent wing tip surface. With that, mesh quality of the surrounding boundary layer and volume mesh was improved and solver stability increased. This was done by splitting the wing tip into two surfaces and enabling *ANSA's* feature detection for the thin section near the trailing edge, such that it would be (falsely) identified as another trailing edge (see Fig. 3.3). The actuator disk surfaces are resolved with trilateral cells of a maximum edge length of 3.8% $\times c_{MAC}$ (120 mm). The boundary layer region of the wing consists of an anisotropic quasi-structured layer mesh. It contains 59% of all the cells in the grid. The layer mesh is constructed by progressively advancing the wing surface mesh in face-normal direction using a growth ratio of 1.2 until the isotropic threshold of 0.8 is met. With a first-layer height of 3 μm, the setup is able to achieve a y^+-value[1] below one at design cruise conditions.

total no. of cells	5.07×10^6
– *Hexa*	3.74×10^6
– *Tetra*	0.96×10^6
– *Penta*	0.16×10^6
– *Pyramids*	0.22×10^6

Table 3.3.: CFD Mesh: cell count and type breakdown

Automating the mesh generation process in *ANSA* was highly facilitated by a pre-existing *Python* script, written by the DLR colleague Attravanam. It uses *ANSA's Python* module to execute a full *Batch Meshing* procedure, from CAD model import to grid file export. It allows for the geometry import of CATPart, IGES and STEP files. Additional user input has

[1] dimensionless index for boundary layer mesh quality; calculated by relating boundary layer flow quantities to the mesh geometry

to be provided through a text file. The text file holds e.g. information about import and export filenames or attributions between meshing parameters, mesh sessions and CAD surfaces. Although the *Python* code already provided most of the functionality needed for automation, it had to be expanded for the purpose of this work in order to accommodate the special needs of the actuator disk. In its original form, the script assigns a generic boundary condition to each CAD surface. After the mesh generation has finished, the desired boundary condition can be specified by reassigning a boundary treatment in the so-called boundary mapping file, which is a required input needed by *TAU*. However, this approach would be insufficient in the case of the actuator disk, since the nodes constituting its boundary must always exist in pairs (as explained in Section 2.3.2), and the grid must contain the actuator disk's meta information (position, axis, radii). *ANSA* does provide these functionalities, but of course only if the boundary condition is selected and the meta data specified *during* the generation process. Therefore, in the extended version of the automation script, the boundary conditions along with all meta data are read in from the user input file and assigned in-situ.

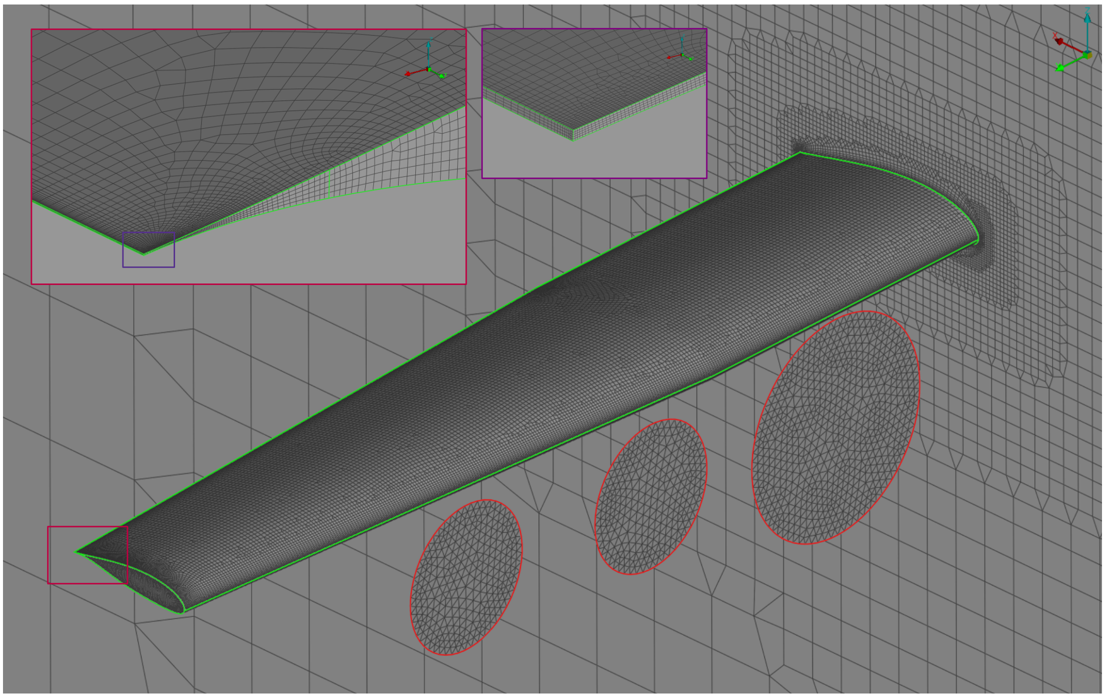


Figure 3.3.: Surface mesh with detail of the quasi-structured mesh at the rear wing tip edge

Finally, the automation script runs a series of quality checks on the resulting volume mesh. The quality checks are also a feature implemented in *ANSA* and, therefore, wrapped to the *Python* module, meaning they can be evoked with function call from the automation script. For instance, the quality checks evaluate a number of geometric quality metrics such as aspect ratios, skewness and warping, and they confirm that the mesh does not include any negative cells. The evaluation criteria are adopted from a pre-defined selection customised for the *TAU* solver which is included in *ANSA*. In case a check fails a test, the script calls a repair function which tries to fix the apparent issue with the mesh. Finally, the mesh is exported to a *netCDF* file, compliant with *TAU's* format specification. The total runtime required for the mesh generation procedure is less than 4 minutes.[1]

In order to verify the adaptability and robustness of the automated mesh generation process, four wing CAD geometries were generated for testing purposes applying excessive values to a selection of geometric parameters (see Table 3.4). For that, one or two parameters are varied at a time while the remaining ones are kept constant, adhering to values of the baseline geometry (as specified in Table 3.1). The quality was tested first with *ANSA's* internal quality checkers and then quantified using *TAU's* utility programme *setup_taugrid*. None of the variants showed a significant deterioration in terms of geometric mesh quality compared to the baseline wing design. This is even true for the variants 1 and 3, for which almost no spatial separation between their propeller disks was available. One occurrence of reduced quality could still be detected for a limited number of tetrahedral cells of variant 2 at the low end of the quality distribution. Using *TAU's smooth_taugrid* function, the minimum score of the tetrahedrals could be improved from 0.0143 to 0.0663. Overall, this test confirms that the setup, including the parametric CAD model, is able to comply with the, realistically much narrower, parameter ranges which are to expected in an actual wing design optimisation.

variant	*modifications*	*rel. quality score (mean)*	*rel. quality score (min.)*
0	(none)	100% (0.908)	100% (0.044)
1	$S_{ref} = 36.5\ m^2\ ;\ \epsilon_{tip} = -6°$	99.6%	118%
2	$AR = 20\ ;\ \lambda = 0.31$	99.6%	88.6%
3	$AR = 5$	100%	118%
4	$t = 0.12$	100%	118%

Table 3.4.: Wing model CAD parameters

3.3. Structural Model

The wing's structural model was generated by *Airbus Defence & Space* and delivered in the form of a *NASTRAN* input file. It is made up of 2516 shell elements (of which: *Quads*: 2411; *Trias*: 105) and 1945 line elements (CROD, CBAR), connecting 2156 individual nodes.

[1] computed on a workstation with 12 CPUs at a clock rate of 3.6 GHz

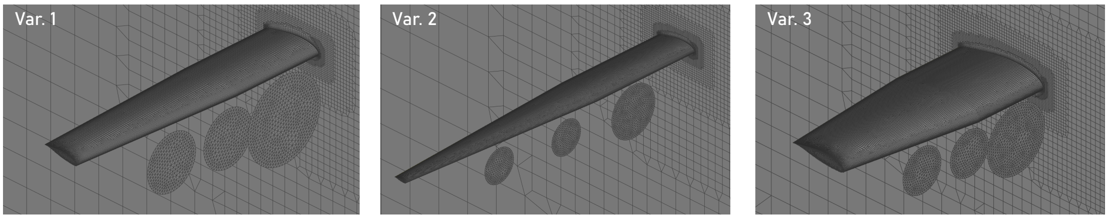


Figure 3.4.: Exemplary results from testing the robustness of the automatic meshing setup with extreme design variations

The model is characterised by a central wingbox, constructed from QUAD4 elements. The ribs are supported by CROD elements and the spars, which connect the ribs in spanwise direction, are mainly reinforced with CBAR elements. This construction accounts for the bulk of the wing's structural integrity. Attached to the front of the central wingbox are auxiliary structures forming the leading edge as well as three separated trailing edge devices (flaps) connected with hinges. Additionally, three nodes representing the mass-centres of the propellers are incorporated. To distribute the propeller loads, each of them is attached to a *Rigid Body Element* (RBE3) which interpolates loads and displacements onto 12 connected nodes on the central wing box, all located on an individual rib at approx. the same spanwise position. Most of the wing's structure is built from a composite material. Apart from the masses of the materials, the model includes structural point masses for fuel and the propellers, amounting to a total structural wing mass of 4,535 kg.

The structural model is derived from an Airbus-internal research aircraft which is very similar to the one to be developed in the HERA project in terms of its overall characteristics. Still, the model has not been fully adapted to the HERA use case which means there are slight deviations, e.g. in the airfoil geometry. This also entails that structural sizing of the present model has not been accomplished yet for load cases representative of HERA's mission and design requirements. Since this thesis is mainly focused on the interworkings of the overall toolchain than on the exact properties of individual components, this structural model, nonetheless, provides an eligible addition to the test case. At the time when the tasks of this thesis were defined, it was originally planned to employ a very simplistic beam stick model as a provisional solution. That was no longer necessary after ADS contributed the current, much more sophisticated model to the project. However, with that, parametrising the structural model was no longer feasible within the scope of this work, as making modifications would become significantly more involved.

Nonetheless, a few smaller adaptions necessary for proper integration of the structural mesh in the toolchain were implemented. For example, *Single Point Constraints* were added to the nodes of the central wing box at the wing's root. This was required to suppress small displacements in the zx-plane which would cause mesh deformation defects in the boundary layer mesh near the symmetry plane. In order to verify the compatibility between the CSM and the CFD mesh and to identify adequate projection parameters, displacement projection was tested using structural solutions obtained from a modal analysis (NASTRAN Solution 106). Because the structural and the CFD model are based on different unit and coordinate systems, data transformation is a prerequisite for the projection step. The transformation includes a scaling operation to translate the lengths from millimeters, which is used in the structural model, to meters, used in the CFD mesh. Additionally, a linear translation has to be performed to account for the offset of the model's origins: $(x; y; z) = (118.0\text{ mm}; 0.0\text{ mm}; -30.0\text{ mm})$. After applying these transformations, the displacement projection could be tested. It was found that the separation between the three flap segments led to a discontinuous deformation state. These discontinuities impaired a smooth projection of the remaining wing's displacement solution onto the CFD surfaces. This issue could be resolved by excluding the trailing edge devices. To achieve this, a set containing only the nodes of the central box and the leading edge section was added to the model. Using only this subset of nodes as input for the *moving-least-squares* (MLS) projection method significantly improved the smoothness of the projected wing surface in the CFD domain, as the continuous displacements field in the right plot in Fig. 3.5 showcases.

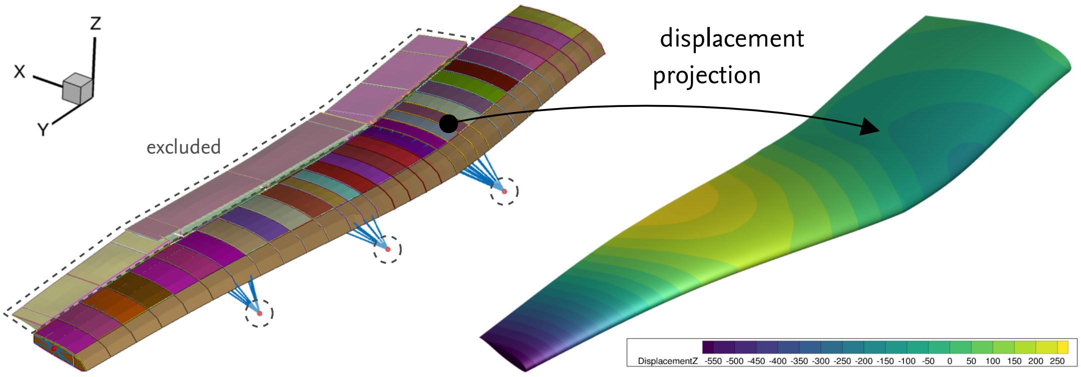


Figure 3.5.: Displacement field of the 6th eigenmode in the CSM domain (left) and projected to the CFD domain (right) and illustrated with a colour plot of the z-displacement in mm (scaled by a factor of 500)

4. Integration of Actuator Disks into Coupled CFD-CSM Analyses

This chapter describes the integration process of the actuator disk (AD) model into the CFD-CSM framework. The first section starts by defining the setup used for initial trial runs of a coupled CFD-CSM simulation including the AD. These runs are conducted with the goal of obtaining a general assessment of the AD's performance within the pre-existing *FlowSimulator* (FS) software environment. The following section outlines the errors that emerged related to the employment of the actuator disk. The chapter closes by detailing the implementation of several solutions, tackling the observed issues as well as a test of the setup's robustness.

4.1. Numerical Setup of the Preliminary Assessment

Having the CFD and CSM meshes for the test case at hand available, it is possible to setup the complete CFD-CSM simulation. For the purpose of executing the various steps in a CFD-CSM loop and handling the exchange of data, the software suite *FSDLRControl* is used. It operates from within the *control layer* of FS and provides a selection of pre-defined scenarios for various aeroelastic problems, including one for steady CFD-CSM analyses that is to be used here. *FSDLRControl* allows the user to adapt various parameters of the coupled simulation with minimal interaction through a text-based, YAML-formatted user-input file. Apart from revealing AD-related errors, identifying a fully functional setting of coupling parameters is an additional goal of this assessment. The final parameter settings which were obtained from this preliminary assessment and used in the subsequent analysis can be found in Appendix C.

A central aspect of a coupled CFD-CSM simulation are the linkages of various nodes and surfaces between the CSM domain and the CFD domain to allow for a proper transfer of loads and displacements. For that, *FSDLRControl* lets the user define pairings of coupled geometric entities. A coupling method can be selected for each pairing. In case of the test case at hand, two types of pairings are to be distinguished. The first type relates to the wing. Its surfaces in the CFD domain are again paired only with a subset of CSM wing nodes to exclude the flaps for the reasons mentioned previously (see Sec. 3.3). The moving-least-squares (MLS) method is used for projecting the wing's loads and displacement between the two domains. The principal parameter affecting this method's behaviour is *numNextCenter* which defines the number of closest neighbouring nodes that are considered for the polynomial fitting (see also Sec. 2.1.2). Thereby, the projection method's level of locality is controlled. A value of 100 has shown to result in a smooth projection for the given models. The second pairing type is utilised for coupling the actuator disks. Since there is merely a singular node in the CSM mesh, representing the propeller, the *nearest-neighbour* (NN) coupling method is applied.

Volume mesh deformation is another crucial component of the CFD-CSM process. *FlowSimulator* offers two principal classes of deformation techniques: radial basis function (RBF) mesh deformation and elasticity analogy (EA) mesh deformation (see also sec. 2.1.3). Both are well-proven techniques, which would in principle be capable of producing high-quality volume meshes for the deformation to be expected with this wing configuration. However, with the RBF technique implemented in FS, input deformations are not reproduced exactly for every node, but only for the base points which were used for the spline-fitting. The remaining field's deformation is only an approximate representation. This is inconvenient for the deformation of the actuator disk boundaries, as the implementation of the AD model relies on the fact that its inlet and exhaust boundaries are exactly congruent. Because this could not be ensured up to machine accuracy with a spline-based mesh deformation technique, the EA method is selected for the following analyses. In the mesh deformation setup, the AD surfaces and the wing surface are assigned to the group of *fixed* boundaries. This designation implies that these surfaces themselves are not subject to deformation during the process. However, the displacements of the wing and the ADs, as calculated by the structural solver, are taken as input to the mesh deformation algorithm. In contrast, the symmetry plane's boundary is assigned a *no-normal-movement* condition which, in this case, restrains any deformations in y-direction. Not fixing all of the symmetry plane's displacement components is done to improve the resulting mesh quality, in particular near the intersection with wing surface. Additional research is necessary to quantify the actual error which would be introduced by using the RBF-based technique and evaluate the effect against potential gains in computational speed.

For the following test runs, the model is simulated under the conditions defined for cruise flight in HERA's design specifications [53]. With a given design cruise altitude of 25,000 ft and a true air speed of 300 kts (154.3 m/s), the boundary conditions stated in table 4.1 follow in accordance with the ICAO standard atmosphere (ISA). Under these conditions, the cruise Mach number Ma_{cr} equates to 0.5, the Reynolds number Re_{cr} to 16.46 million, using the mean aerodynamic chord as reference length. An angle of attack of 1 degree is chosen. To simplify the simulation of a reasonable propeller flow, a load is directly applied to the AD boundaries as a constant surface load, amounting to a total thrust of 7.6 kN of which 50% are contributed from the innermost disk (AD1) and 25% from the two outer ADs (AD2, AD3) each.

Since these trials are also intended for testing the limits of the CFD-CSM toolchain and

atmospheric quantity	*value*
ambient temperature t_{cr}	238.6 K
air pressure p_{cr}	37600.9 Pa
air density ρ_{cr}	0.5489 kg/m³
sonic speed c_{cr}	309.64 m/s
dynamic viscosity μ_{cr}	1.5541×10^{-5} Pa·s

Table 4.1.: Atmospheric conditions at cruising altitude acc. to ISA

proofing its robustness, extensive additional loads in z-direction are applied to the wing at multiple spanwise locations. The magnitude of the extra forces goes beyond of what the wing is likely to be subjected to in a real-world scenario. This approach was nonetheless chosen specifically for the purpose of exaggerating possible erroneous behaviour, highlighting effects that might have been hard to detect otherwise. Making use of the *FSDLR-Control* feature to apply extra loads to individual nodes of the structural model, additional forces in z-direction F_Z are introduced as specified in table 4.2. The forces are defined so that mainly the outer two disks are displaced, whereas the innermost AD remains mostly unaffected by the wing's deflection. Therefore, the positive forces are only applied on the outboard half of the wing and counter-balanced by an equal and opposite set of forces between the relative spanwise positions of 0.35 and 0.47. This way, the effects of a wider range of displacements can be studied simultaneously.

rel. spanwise location $y/\frac{b}{2}$	0.35	0.41	0.47	0.59	0.67	0.75	0.83	0.92	1.00
applied force F_Z [kN]	-160	-160	-160	80	80	80	80	80	80

Table 4.2.: Additional loads applied in the test setup

4.2. Observed Issues and Implemented Solutions

4.2.1. Symmetry Plane Deformation

One of the first observed issues resulted in the CFD-CSM cycle terminating after a single iteration. During the second CFD solver run, the *TAU* code failed to start while processing the deformed CFD mesh. The error message from *TAU* indicated that the issue was the non-planarity of the symmetry plane boundary. Upon closer examination, it became evident that the deformation of the mesh in the preceding step had resulted in out-of-plane deformations of the symmetry plane boundary. Although the displacements were minimal, they exceeded the internally specified limit, resulting in the immediate termination of *TAU's* execution.

To resolve this issue, additional Python code was incorporated into the FSDLRControl plugin, which steers the sequence of operations in the steady CFD-CSM scenario. This extra code comprised a simplistic projection function designed to suppress any out-of-plane deformations. As the symmetry plane surface is an element of the global zx-plane, this was achieved by explicitly setting the y-coordinate of every node located on the boundary to zero. In order for this to be accomplished, the symmetry plane's boundary ID, which is a unique identifier assigned during mesh generation, must be provided in the user-input file. This ensures that the correct surface mesh is selected at runtime. The projection function is invoked on two occasions during the course of a single iteration of the CFD-CSM simulation. The first function call is made after the displacement projection between CFD-CSM coupling meshes, the second one after the CFD volume mesh deformation. The rationale for not invoking the function solely after the deformed volume mesh has been constructed is to guarantee that the mesh deformation process is initiated with a planar symmetry plane surface. Based on its coupling definition, the symmetry plane would be expected to exhibit no surface-normal displacements whatsoever. However, due to the deflection of the wing, out-of-plane deformations can be evoked at the intersection of the wing and the symmetry plane. If fed to the mesh deformation operation, these imperfections could result in the deterioration of the volume mesh quality, as the deformations could propagate to the interior of the mesh. A more generally applicable long-term solution to this problem would be to inhibit the occurrence of out-of-plane displacements at the symmetry plane in the first place. This could either be done by incorporating the symmetry, inherent to the CFD mesh, also into the structural model (e.g. through mirroring), or by modifying the mesh deformation method, such that it enforces narrower tolerances if the *no normal movement* condition is applied.

The implementation of this workaround pursued in this work enabled *TAU* to execute without any runtime errors on the deformed CFD mesh. A limitation of this approach is that it is not universally applicable. In this specific instance, projecting the symmetry plane's nodes was feasible by prescribing a y-component of zero, which would evidently fail for an arbitrarily oriented surface.

4.2.2. Updating the Actuator Disk's Meta Data

Once the issue caused by the symmetry plane deformation was resolved, a coupled CFD-CSM simulation run could be completed successfully, using the setup introduced in Sec. 4.1. However, the flow solutions revealed that the actuator disks had begun to generate less or even negative thrust, as a result of the deformation. This, in turn, created disturbances of the wing circulation and in case of AD3 a backflow region around the disk, illustrated in Fig. 4.1. The figure also shows the development of AD mass flow data during the CFD solver run in the first cycle of the coupled simulation. The mass flow $\dot{m}$ through a surface of area A can be quantified by $\dot{m} = \rho \cdot u \cdot A$, where u is the disk-normal through-flow velocity and ρ the fluid's density. Thus, the mass flow is directly proportional to the flow velocity and, therefore, also connected to the AD thrust (see also sec. 2.3). The disks' mass flow diagram on the left of Fig. 4.1 indicates that the strength of the reversal effect is linked to the magnitude of each disk's displacement, as the disks further outboard were more severely affected while the innermost disk's mass flow remained nominal.

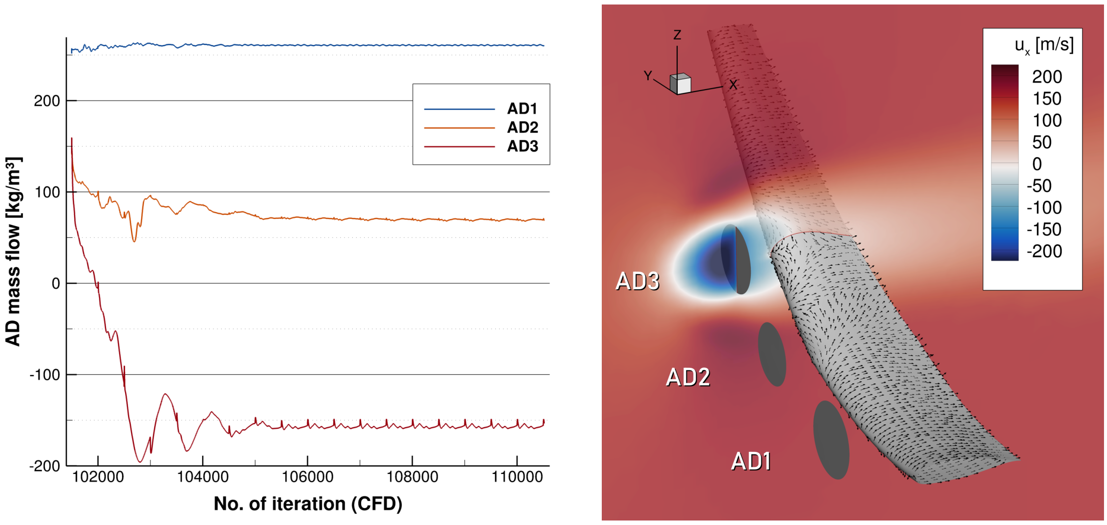


Figure 4.1.: AD mass flow divergence: plot of mass flow monitoring data (left), colour-plot of the flow velocity's x-component at an xz-section through AD3 with vectors indicating the wing's skin friction (right)

A brief investigation of this effect was conducted, using a range of different load factors. It showed that a linear relationship between an AD's displacement and the mass flow appears to exist. Moreover, the mass flow diverges not until the disks are displaced by an amount slightly higher than their radii. This strongly suggests that the issue could be caused by a modeling error that leads to falsely calculated disk loads. A potential cause for the error might be outdated AD meta-data stored in the mesh. This meta-data holds geometric information about each actuator disk (its radius, position, and axis) which is required by *TAU* so that the solver can transform values from dimensionalised external mesh data to normalised internal modeling quantities and vice-versa (see also: Sec. 2.3.2). When a *TAU* mesh is imported to *FlowSimulator*, this meta-data is processed and stored in an appropriate data container provided by an *FSDataManager* (FSDM) sub-class. Since this data is apparently not updated in accordance with the wing's current deformation state during the coupled CFD-CSM simulation run, it is a likely cause for the erroneous actuator disk thrust. To make the actuator disk an integral part of the CFD-CSM toolchain, the meta-data has to be retrieved from the mesh geometry during each iteration to match the disks' actual position and orientation of the deformation state.

Therefore, the *FSDLRControl* suite driving the CFD-CSM simulation was expanded by implementing the *ActuatorDiskCoupling* module. It's principal function is to update all actuator disk meta data, more specifically each AD's center point and normal vector, using the current displacements and rotations obtained from the most recent structural solver run. The module is comprised of two classes:

***class* ActuatorDiskManager (ADM)** is the central class which executes the main sequence of operations and serves as the exclusive interface for interaction with the module. For that, a singe public method is implemented to provide the functionalities `ADM` is designed to fulfill: the `run()` method.

***class* PairingComponent (PC)** is the ancillary class which is used by the `ADManager` class to organise and process data. It handles operations such as mesh data extraction and updating disk meta-data. Each `PairingComponent` instance represents an actuator disk's geometry, either in the CFD or in the CSM domain. To differentiate between the two domains, the class has a *type* attribute that can either be *Source*, relating to the CSM domain in this case, or *Target* for entities in the CFD domain, respectively.

For the module to operate, the software requires information about the associativity between surfaces and nodes representing the AD in the structural domain and in the fluid domain, analagous to the user-defined coupling pairings which allow for the transfer of loads and displacements. Thus, at the start of the coupled simulation, the software instantiates CFD-CSM pairings for each actuator disk for the geometry specified in the user-input file. During the simulation, the *ActuatorDiskCoupling* module is invoked from *FSDLRControl's* module `CFDCSMCouplingScenario`. As soon as an `ADM`-object is instantiated, the following sequence of operations is triggered:

1. `getUserInputAsDict()` retrieves user input as a *Python* dictionary.
2. `assemblePairingDefs()` processes the user-input and checks it for inconsistencies. Then, the method assembles so-called pairing definitions by creating two associated instances of the `PairingComponent` class for each actuator disk coupling found in the user-input: one *Source* and one *Target* component.

The method is called at the beginning of the steady CFD-CSM loop before the first iteration of the simulation is initiated, such that user-input related errors are handled prior to any computation-heavy operations. The constructor method also creates attributes of FSDM and *FSDLRControl* objects needed for logging and parallel processing. What the method explicitly omits is making mesh-related operations or even instantiating any FSDM mesh objects. This is done because at the moment the `ActuatorDiskManager` is initialised, mesh imports might not have been finalised, yet.

As the simulation progresses, the `ADManager's` `run()` method is called for the first time. This is done from within the CFD-CSM loop after the volume mesh deformation of the first iteration has finished, thus immediately before the CFD solver run commences. The `run()` command initiates the sequential execution of four class methods. Figure 4.2 illustrates how each one of `ADManager's` methods corresponds to a call of another method from the `PairingComponent` class.

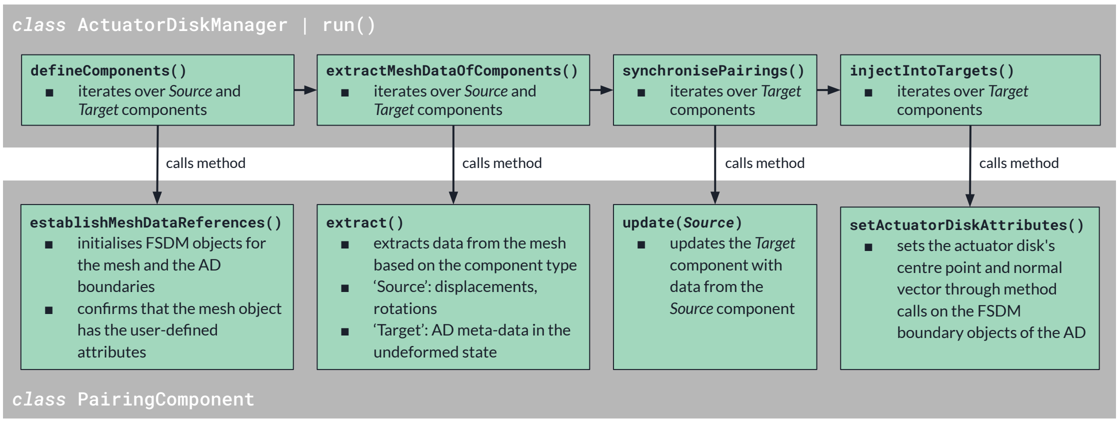

class ActuatorDiskManager | run()
defineComponents()
iterates over Source and Target components
extractMeshDataOfComponents()
iterates over Source and Target components
synchronisePairings()
iterates over Target components
injectIntoTargets()
iterates over Target components
calls method
calls method
calls method
calls method
establishMeshDataReferences()
initialises FSDM objects for the mesh and the AD boundaries
confirms that the mesh object has the user-defined attributes
extract()
extracts data from the mesh based on the component type
'Source': displacements, rotations
'Target': AD meta-data in the undeformed state
update(Source)
updates the Target component with data from the Source component
setActuatorDiskAttributes()
sets the actuator disk's centre point and normal vector through method calls on the FSDM boundary objects of the AD
class PairingComponent

Figure 4.2.: Flowchart of the sequence of method calls executed by the `ActuatorDiskManager.run()` method

Each method's purpose is explained in greater detail in the following:

`ADM.defineComponents() | PC.establishMeshDataReferences()`

creates references for both types of components to various objects of the FSDataManager, such as meshes (class FSMesh) and actuator disk boundaries (class FSPeriodicBoundary). Additionally, it is asserted that the user-defined data can actually be retrieved from the referenced objects: for *Source* components, it checks if the mesh has the specified cell attribute (e.g. boundary ID); for *Target* components, it identifies and verifies the existence of the specified actuator disk in the mesh.

`ADM.extractMeshDataOfComponents() | PC.extract()`

extracts the desired geometric information from the mesh. In case of a *Source* component, the method retrieves displacement and rotation vectors for all CSM nodes specified by the user and stores the average value of each quantity. In case of components of type *Target*, the extraction method is only called once in order to retrieve the position (centre points) and orientation (normal vector) of the undeformed state.

`ADM.synchronisePairings() | PC.update(`*`Source`*`)`

updates the *Target* component data by using the displacements and rotations values stored in the associated *Source* component in the argument. For the displacements, this is done by adding the values to the position of the undeformed state, obtained during the first extraction. A potential coordinate transformation between the CFD and the CSM meshes is taken into account as well. If needed, the user can specify a translation and a scaling vector in *FSDLRControl's* input file. For rotations, a function imported from the `scipy` library is used to apply an extrinsic Euler-rotation to the disk's normal vector, taking the three rotation angles from the CSM solution as input.

`ADM.injectIntoTargets() | PC.setActuatorDiskAttributes()`
finalises the update of actuator disk meta-data by overwriting the now obsolete centre point and normal vector with the newly calculated data. To do that, the method uses a setter-function of the boundary object (FSPeriodicBoundary) which is a data container for the actuator disk boundaries. This setter-function had previously not existed and had to be specifically added within FSDM for this purpose.

Due to the large scale of the meshes and requirements for high computational speed, these types of simulations are usually computed in parallel to balance the loads between multiple processors. To enable parallel processing, additional adaptations of the code had to be made which are not covered by the previous explanation. In general, the *ActuatorDiskCoupling* module makes use of pre-existing functionalities available in *FSDLRControl's* utility modules to gather, scatter and broadcast data between the parallel processes. For more details, the source code of the *ActuatorDiskCoupling* module can be found in Appendix B.

Finally, the module was tested by comparing the locations and orientations of the actuator disks as stated by the CSM solution with *TAU's* output of AD meta-data. To allow for easier monitoring, suitable logging messages had been implemented to both the *ActuatorDiskCoupling* module and *TAU*. The test as well as a manual inspection of the deformed meshes (CFD and CSM) confirmed that AD meta-data is consistently updated in each iteration of the coupled CFD-CSM simulation. This is also in agreement with the obtained CFD solutions which show that the actuator disks exhibit a nominal mass flow and generate the expected thrust.

4.2.3. Suppressing the Expansion of Actuator Disk Surfaces

With the *ActuatorDiskCoupling* module properly implemented, it is ensured that the actuator disks' meta-data are in line with the current configuration's deformation state. However, another irregularity emerged in the actuator disks' mass flow data. This time, it appeared as if the mass flow values varied in proportion to the disks' displacement (see Fig. 4.3). Unlike before, when the mass flow divergence pointed to an error related to the velocity of the disk, the cause of the current issue are the disk areas. A measurement of the surface areas comparing the undeformed with the deformed state showed that the disk areas had expanded as a result of the wing deflection. More specifically, the area inflation was found to be a side effect of the displacement projection method used. The employed projection method, following a nearest-neighbour approach, determines the CFD domain's nodal displacements as a combination of two quantities in the CSM solution: the associated node's displacement and the (linear) product of its rotation and the

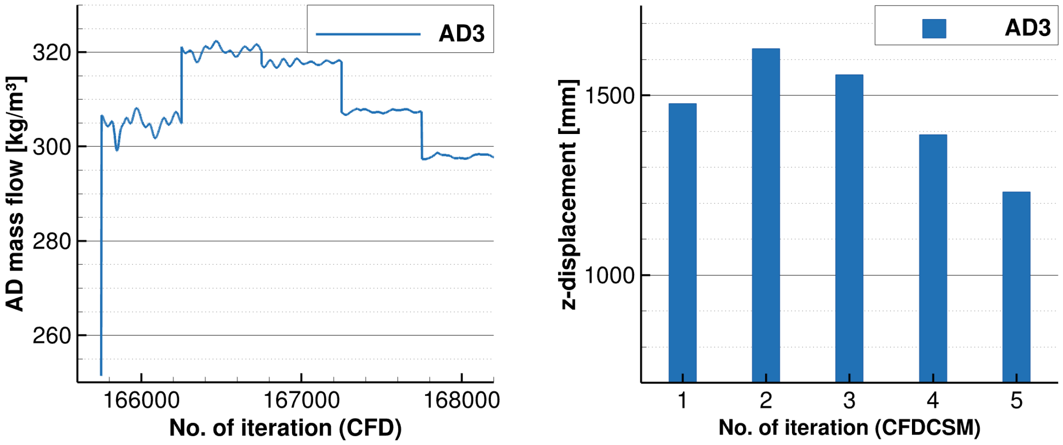


Figure 4.3.: Diagrams of AD3 mass flow monitoring data (left) and chart of AD3 displacements (right)

offset vector between the CFD and the CSM node (see also Sec. 2.1.2). Due to the wing bending under load, the actuator disks are subjected to a significant rotation around their normal axes (global x-axis). This linear formulation is chosen deliberately to ensure that the projection method does not violate the energy conservation principle, which requires force/moment and displacement/rotation projections to be aligned. It does, however, also lead to the previously mentioned area inflation effect, illustrated in Fig. 4.4.

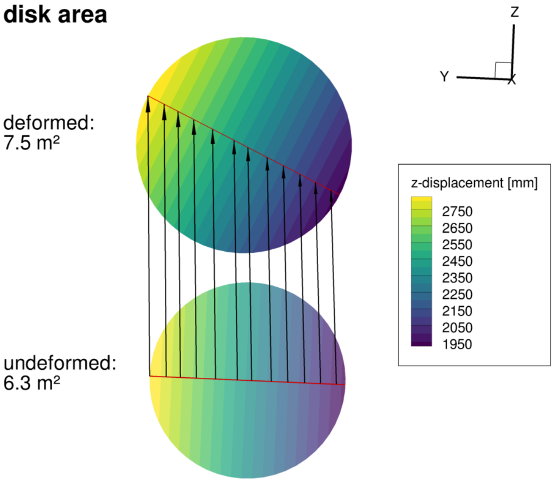


Figure 4.4.: Illustration of the area expansion for AD3 caused by a linearised rotation method

In order to inhibit the expansion of the disk surfaces, the disk-normal rotation was eliminated by explicitly setting the specific value of the ADs' CSM nodes to zero before the displacements (and rotations) are projected. Although, a notable disadvantage of this solution is that it has a negative impact on mesh quality. By suppressing only the disks from rotating, they experience a *relative* rotation with respect to the deflected wing surface. Additionally, a shearing motion between the disks is introduced which distorts the intermediary volume mesh. Figure 4.5 shows the decreased relative mesh quality (based on the mesh quality metric devised by Knupp [54]) for a y-section of the mesh between two disk surfaces. The depicted scoring is relative to the quality of the initial (undeformed) mesh. The highest score of 1 represents a quality equal to the initial state, whereas a score of 0 marks the point at which a cell degenerates (i.e. negative volume).

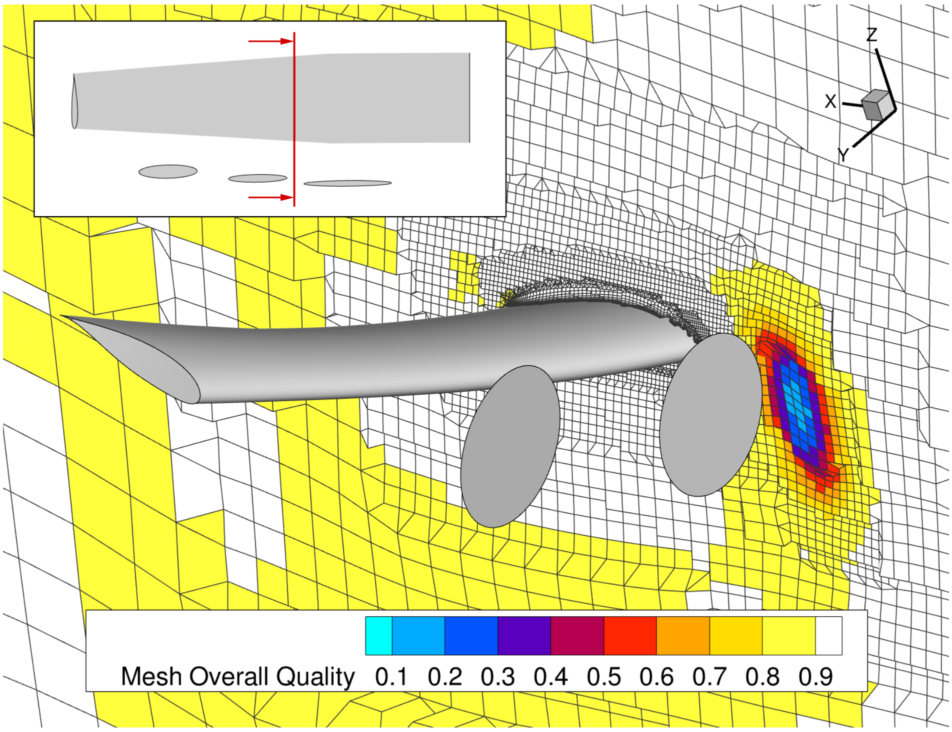


Figure 4.5.: Colour plot of the overall mesh quality at a section in between actuator disks 1 and 2

Due to the time constraints of this work, this approach was, nonetheless, chosen as it is simple to implement and the negative effects on geometric quality are still tolerated by the CFD solver. It should also be noted that the quality degradation is more severe under the extreme loads of this test setup than of what is expected under realistic conditions. Nonetheless, a better solution to the problem would be the implementation of a projection method for the AD surfaces which uses (non-linear) trigonometric functions. Neglecting the energy conservation principle, which is ensured with the original, linear projection technique, is in this case irrelevant since the actuator disk rotations are structurally decoupled from the rest of the wing, not affecting its total strain energy.

4.3. Toolchain Robustness

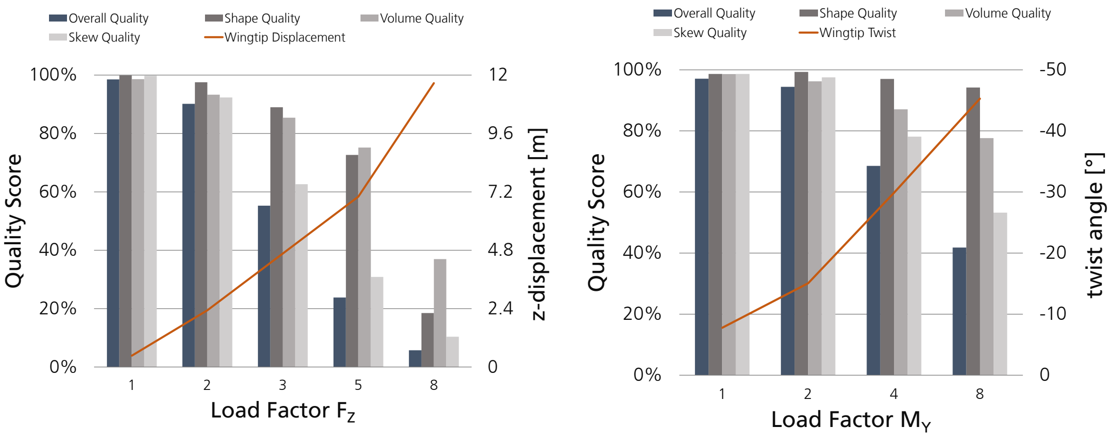
A maxim for the development of the CFD-CSM toolchain at hand is its ability to handle a wide range of design parameter inputs. This quality, denoted by the term robustness, is critical for the toolchain's intended application for optimisation problems. Since an optimisation's primary purpose is to identify combinations of design variables that are as yet unknown, the underlying simulation framework ideally has to be capable of tolerating every theoretically realisable state, so as to not exclude a potential optimisation solution. However, verifying the toolchain's proper functioning under every condition is impossible due to the infinite size and typically high dimensionality of the parameter space. One approach for testing large parameter spaces is to define a set of edge cases and verify only these, as it was done with the automatic mesh generation process, implemented in this work. Using extreme design parameter inputs, the mesh generation setup proved its robustness in handling wing geometries that well exceeded the range of practically sensible designs. Also, the AD-integration process, covered in the previous section 4.2, has confirmed that the setup is apparently able to tolerate large positive wing deflections and negative propeller thrust.

Regarding the AD-integrated CFD-CSM setup, a possible way to define its robustness can be the tolerance towards arbitrary deformations as they might appear during a design optimisation. This is because an optimiser might converge towards a design with a lower structural stiffness than the baseline model, leading to larger displacements of the wing and the surrounding mesh. Highly deformed cells in a mesh can have detrimental effects, in particular for the CFD solver's convergence stability. In general, the more distorted the cells, the more likely a CFD run diverges, with negative cells representing the limit, which in case of the solver *TAU* causes the run to fail immediately. The *FlowSimulator* plugin *FSMeshQuality* can evaluate a mesh based on the generalised scoring metric *Overall Mesh Quality*. It is calculated from the product of three other geometric quality metrics: *Shape-*, *Volume-*, and *Skew-Quality* (acc. to the theory of Knupp [54]). As explained above, the score can range from a value as high as 1, matching the quality of the undeformed cell geometry, to 0 at which point the cell collapses and its volume becomes negative. Such a quality metric can therefore be an appropriate indicator for the CFD solver's expected behaviour.

Following this reasoning, the robustness of the present CFD-CSM setup is to be tested by subjecting the model to artificially added wing loads in order to systematically create large deformations, while evaluating the resulting mesh quality. Because the CFD solver's convergence behaviour is often decided by the worst quality regions in a mesh, the minimum value of the *Overall Mesh Quality* is considered a representative indicator and is therefore examined in the following.

For testing the robustness, two types of loads are applied: the first one is a set of equal forces in z-direction applied to the central wing box which are distributed over a section of the span, causing the wing to deflect upwards; the second type is a moment around the

y-axis applied at the wing's tip, inducing a negative twist of the wing. The set of forces, denoted F_Z, are evenly applied to 12 nodes ranging from a relative spanwise location of 24.5% to the tip, amounting to 120 kN of force in total. The moment M_Y exerts a torque of -12.5 kNm. Applying the two load types separately, scaled by the given load factors, results in the minimum mesh qualities and deformations shown in Fig. 4.6. Both diagrams highlight the robustness of the setup considering the immense displacements exhibited by the wing. The deterioration of the mesh quality is predominately driven by the declining *Skew Quality*. The maximum torsion moment (load factor of 8) is the only tested case which produced a failure of the subsequent CFD solver run. However, under that load the wing exhibits an excessive tip twist angle of about -45°. Apart from that case, the CFD runs in the test ran without any errors. Overall, the results underscore the stability of the mesh deformation procedure within the spectrum of realistically occurring deformation states. It has to be acknowledged that the occurring deformations by far exceed the validity range of the geometrically linear assumptions, underlying the structural solution, which is only acceptable in the context of this test setup.



a) positive z-forces F_Z

b) negative torsion moment M_Y

Figure 4.6.: Minimum mesh quality and deformation for wing under artificial loads

Under real-word conditions, the wing is deformed by a more complicated loading condition defined by the aerodynamic forces which themselves depend on the current deformation state. In order to examine the combined effects, another set of simulations were conducted in which the wing is simultaneously subjected to multiples of the forces F_Z, and the moment M_Y, in addition to the aerodynamic forces occuring under cruise conditions. The applied mesh deformation technique is the *Elasticity Analogy* method. The resulting minimum mesh qualities are given in the Table 4.3a. The results again confirm the robustness of the setup under a wide range of load cases. Even for the worst measured score of 6.2, which involved a wing tip displacement of around 7 m (or 54% of the half-span), the following CFD solver run continued without any errors. Although the mesh quality deteriorates more rapidly under combined loads, the superimposed deformations of the forces F_Z and the moment M_Y actually lead to an improvement of the mesh's quality, as the best score is achieved for the maximum load case of the moment M_Y.

For comparison, this test was repeated using a slightly altered structural model which allowed the actuator disks to rotate freely again. As explained in the previous section, the ADs' rotation had to be restricted in order to prevent their surface areas from growing under deformation (see Sec. 4.2.3) which showed to have a negative impact on the mesh quality. This could be validated through the results of this test given in Table 4.3a. For the lower load factors, the achieved mesh quality is notably higher, although the scores also quickly deteriorate with increasing additional wing loads, approximately matching the scores of the previous test with suppressed AD-rotation.

		$F_Z\cdot$ LF + aerodyn. forces		
	LF	1	2	5
$M_Y\cdot$ LF	1	54.0	28.2	8.1
	2	66.7	35.3	9.0
	4	69.7	55.3	11.5

a) with constrained AD rotation

		$F_Z\cdot$ LF + aerodyn. forces		
	LF	1	2	5
$M_Y\cdot$ LF	1	91.6	28.3	6.2
	2	89.7	35.2	7.1
	4	70.0	44.6	9.7

b) with freely rotating ADs

Table 4.3.: Minimum mesh quality in percent for combined load cases with various load factors (LF)

To conclude, this investigation showed that the CFD-CSM toolchain is highly robust and retains its functionality even for high wing deformations. Even though, constraining the actuator disks rotation causes no immediate failure, it negatively impacts the mesh quality and should ideally be avoided.

5. Coupled CFD-CSM Analyses and Design Variations

The following chapter covers the conducted CFD-CSM analyses employing the integrated actuator disk model. The first section compares the aerodynamics of the flexible wing using coupled CFD-CSM simulations with results obtained from monodisciplinary CFD simulations for the hypothetically rigid (non-deforming) configuration. The second section contains the results from multiple design variations to assess their aerodynamic performance and quantify their sensitivities. The chapter begins by explaining the setup of these simulations and outlining the required preparatory steps.

5.1. Numerical Setup and Preparatory Steps

The numerical setup utilised for the main CFD-CSM analyses of this thesis makes use of the existing functional parameter settings determined during the preliminary investigations covered in Section 4.1. A complete version of the input parameter YAML-file used in the coupled CFD-CSM simulations can be found in Appendix C. The most notable differences to the prior simulations have to do with the boundary conditions: To allow for a reasonable comparison between the rigid and the flexible wing's performance or between different wing designs, the following analyses do not use a fixed angle-of-attack setting. Instead, they include a trim cycle to achieve the specific lift coefficient required for level flight under cruise conditions. Moreover, the CFD simulations will involve a propeller thrust trim. Further details regarding these trim runs are explained in the corresponding paragraphs of the following section which covers the CFD-only simulations done in preparation of the full CFD-CSM runs.

5.1.1. CFD Simulations

The CFD analyses, used to simulate the aerodynamics of the hypothetically rigid wing geometry, are conducted for two purposes: First, they present a reference case for the simulations of the flexible wing geometry studied in the comparative analysis; second, the obtained CFD results serve as starting solutions for the coupled simulations, intended to reduce their runtime. As with the preliminary assessment covered in the previous chapter, the onflow conditions are again chosen to be consistent with the cruise conditions defined in HERA's top-level aircraft requirements [53] (see also Table 4.1). The simulations employ the implicit *Backwards Euler* time-stepping scheme. For spatial discretisation, the Jameson scheme, a central discretisation with 2$^{\text{nd}}$ order accuracy, is used for almost all computations with *Matrix Dissipation* engaged. Only when a simulation is started from scratch, a 2$^{\text{nd}}$ order upwind scheme (AUSMDV) is employed for the first 2,000 iterations. This has shown to improve the solver's stability in the starting phase significantly. In accordance with prior research [30], the Spalart-Allmaras (SA) turbulence model is selected; more specifically the SA-negative variant of the model. Additionally, the curvature correction method SARC is engaged.

For the following analyses, the actuator disks' forces are no longer defined through directly prescribed surface loads, but instead determined with *Blade Element Theory* (BET), taking the local flow conditions present at the actuator disk into account. The implementation of propeller torque does not only result in a more realistic representation of the propeller flow by introducing a swirl component to the wake, but has the additional benefit of yielding results for the propeller's propulsive efficiency. To set up the BET formulation of the AD, geometric and aerodynamic data of a propeller has to be specified in *TAU's* user-input files. Since a propeller geometry has not been determined yet for HERA's conceptual design, existing propeller data is adopted for the purpose of this work. Because the data is stored in a non-dimensional format, it can easily be adapted to match any desired propeller radius.

The utilised propeller data is inherited from the *SynergIE* project [55], which also investigated a 70-seat regional propeller aircraft, similar to HERA's design. In the related paper cited above, the propeller data has been used to simulate various *Distributed Electric Propulsion* (DEP) configurations with the help of CFD analyses. The aerodynamic data has been generated with a low-fidelity propeller design software, neglecting 3D and compressibility effects. Because of the involved simplifications, it is not unlikely that the propeller performance is overpredicted, especially under off-design conditions. In the cited source, the authors state propulsive efficiencies of over 90% [55]. Prior to the analysis, the original propeller data had to be slightly adjusted by simplifying its radial distribution to be constant in order to improve numerical stability. The data tables used in this work are given in Appendix E.

A crucial difference regarding the BET formulation of the actuator disk relates to the determination of a thrust setting. Instead of directly prescribing the load, with the BET formulation the thrust has to be determined iteratively by adjusting the propeller blade's pitch angle parameter. Depending on the local flow conditions at the disk and a preset rotation speed, a resultant thrust force is created. In order to identify a trimmed state with respect to lift and thrust, a two-staged approach is chosen. First, only the lift coefficient is trimmed starting from a reasonable initial guess for the angle of attack and the propeller blade's pitch angle based on experience. Once the suitable angle of attack has been found for the targeted lift force, a secondary trim cycle for the actuator disk thrust is conducted.

For trimming the angle of attack, *TAU's* internal *target-clift* method was used. With a maximum take-off weight m_{MTOW} of 32 t and the boundary conditions referenced in Table 4.1, the cruise lift coefficient $C_{L,cruise}$ can be found with the following equation (eq. 5.1):

$$C_{L,cruise} = \frac{L_{cruise}}{\frac{1}{2} \cdot \rho \cdot u_{\infty}^{2} \cdot S_{ref}} = \frac{m_{MTOW} \cdot g}{\frac{1}{2} \cdot \rho \cdot u_{\infty}^{2} \cdot S_{ref}} = 0.6898 \tag{5.1}$$

The ***target-clift*** trim routine was activated after the solver completed the first 1,000 iterations to allow the approximation to stabilise. Using the flow solution obtained from the lift-trim cycle, the thrust trim routine is initiated. For that, thrust target values for cruise flight were defined based on an estimation of the required cruise thrust in order to reproduce a roughly realistic propeller flow. Letting the thrust be equal to an estimated total drag force D_{total}, the combined thrust force $T_{prop,comb.}$ can be approximated using eq. 5.2. Assuming a lift-to-drag ratio $\eta_{est.}$ of 20, the total thrust amounts to roughly:

$$T_{prop,comb.} = D_{total} = \frac{L_{cruise}}{\eta_{est.}} = \frac{m_{MTOW} \cdot g}{\eta_{est.}} \approx 16\ kN \tag{5.2}$$

for the full aircraft, resulting in 8 kN of thrust for the semi-wing model at hand. The total thrust is distributed among the three actuator disks according to their individual disk area in relation to the total disk surface area. The propeller's rotation speeds are defined by setting their relative blade tip velocity at cruising speed to 0.75 Ma in order to keep them below the transsonic regime. Using eq. 2.6, the resulting rotation speeds are 1104.6 rpm (revolutions per minute) for the large propeller (AD1) and 1657 rpm for the two smaller propellers (AD2, AD3), equating to an advance ratio of 2.8 (see also equation 2.12).

The utilised thrust trim routine works by iteratively varying each AD's pitch angle parameter to arrive at the targeted thrust force. During this process, *TAU's* ***target-clift*** trim method is engaged as well to simultaneously balance out the lift fluctuations induced by the propeller flow. In a second step, each disk's force vector is decomposed into two components, using the current angle of attack: a lift component, augmenting the wing lift, and a forward thrust component, directly opposing the drag force. Then, the script adjusts the targeted (wing) lift coefficient based on the calculated lift component generated by the actuator disks. This cycle is repeated until the lift coefficient stabilised and the force balance between the ADs' thrust forces and the defined thrust targets have converged below an error of 1 N. The obtained CFD solution can be used as a starting solution in a subsequent CFD-CSM run.

5.1.2. Coupled CFD-CSM Simulations

A typical CFD-CSM simulation conducted in the context of this thesis usually converged within 12 CFD-CSM cycles, accumulating at most around 15,000 CFD solver iterations. Considering that a relatively low CFL number of 5 was used for the CFD computations, there is likely potential to further accelerate the convergence rate. A CFD-CSM run is considered converged when the sum over all displacement and force vectors varies less than 10^{-4} from one cycle to the next. As previously mentioned, the parameter settings for the coupled CFD-CSM analysis are mostly identical to the preliminary setup used in Section 4.1. Additionally, all following CFD-CSM simulations were conducted with *TAU's target-clift* trim routine activated. Because the wing deflection causes variations of the aerodynamic forces, it is necessary to keep the lift trim engaged for the coupled simulation to further adjust the angle of attack. Moreover, *TAU's Cauchy convergence* method is activated during the coupled simulations in order to reduce the number of CFD iterations in each CFD-CSM cycle to the necessary minimum. With *Cauchy convergence* engaged, *TAU* monitors specified aerodynamic quantities and continues the approximation until their maximum variation over a given number of iterations has become smaller than a pre-defined threshold. This is illustrated in the plot in Fig. 5.1 which shows the concatenated monitoring data of all CFD iterations during a complete coupled CFD-CSM simulation. Apart from the angle of attack adjustments controlled by the trim routine, the figure depicts the varying lengths of the several CFD runs conducted over the course of this exemplary simulation. As shown, most of the CFD iterations occurred within the first CFD-CSM cycles between which the variability in the structural domain (displacements and loads) is also the highest. In the first three CFD-CSM cycles, the CFD runs reach the defined maximum of 2,000 iterations. As the CFD-CSM simulation converges, the *Cauchy convergence* criterion terminates the CFD-runs before the iteration limit is reached, as soon as the lift coefficient has sufficiently stabilised. For that, the lift coefficient has to vary less than 1×10^{-4} and the drag coefficient less than 1×10^{-2} over 100 CFD iterations. For the final run, the convergence criterion was narrowed to 5×10^{-5} (and 1×10^{-4} for the drag coefficient) in order to reduce the error residual of the resulting flow solution. This is why the last run needed the most CFD iterations in the example given in Fig. 5.1.

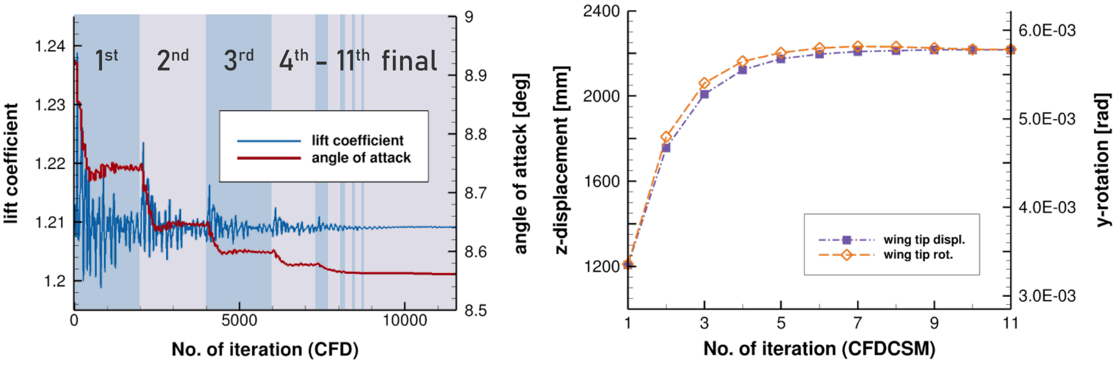


Figure 5.1.: Compiled monitoring data over 11 CFD-CSM cycles and a final run: lift coefficient and angle of attack (left); wing tip displacement and rotation (right)

As a side-effect of the lift trimming during the coupled simulation, force and displacement fluctuations between different CFD-CSM cycles are amplified, resulting in an overshooting which slows the convergence. To attenuate this effect, the relaxation factor applied for CSM under-relaxation was lowered from the default value of 0.7 to 0.55.

Due to the trilateral interaction between lift, thrust and angle of attack, an additional thrust trim cycle would be required during the coupled CFD-CSM simulations as well. As the lift trim routine varies the wing's angle of attack, it also changes the propeller's inflow conditions, leading to slightly altered thrust outputs. To adequately account for this interaction mechanism, propeller-wing configurations like this one need a dedicated trim algorithm, which would be out of scope of this thesis. Because the involved thrust discrepancies are relatively small (< 1%), thrust trimming is neglected for the coupled simulations conducted in this work.

5.2. Comparative Analysis

The analysis in this section discusses the differences between the pure CFD simulations of the rigid geometry and the coupled CFD-CSM simulations of the flexible wing, examining the static aeroelastic effects present caused by the wing deflection. Two different load cases were simulated: level-flight (1g) under cruise conditions and a (pseudo[1]) 2.5g pull-up maneuver at sea-level altitude. This 2.5g maneuver was selected for comparison since it marks one of the dimensioning load cases for transport aircraft (acc. to the flight envelope defined in CS 25.331 [56]). For post-processing the results, the in-house developed software tool *AeroForce* is used [57]. *AeroForce* calculates aerodynamic forces and attributed coefficients by integrating skin friction and pressure values over the corresponding surfaces. The software is also used in this analysis to generate spanwise distributions of the aerodynamic loads acting on the wing.

Regardless of the simulated maneuver, the general flow phenomenology is largely dominated by the propellers due their strong induction flow. To be able to differentiate the various sources of influence on the aerodynamic forces of the wing, an additional simulation under cruise conditions was conducted with all three propellers inoperative. This allows the propellers' influence to be isolated and distinguished from the aeroelastic effects. Figure 5.2 depicts the wing's spanwise distributions of lift and drag, comparing the AD-on and the AD-off simulation. The forces are given as non-dimensional coefficients and are scaled by the term c_{MAC}/c (with c denoting the local chord length), such that the curves represent the wing's global load distribution. The diagrams clearly reveal the interference effects of the three propellers on both the wing's lift and drag. As shown in the left diagram in Fig. 5.2, the lift distribution undulates as a consequence of the propellers affecting the local flow velocity and its angle of attack. The superposition of these two phenomena leads to an alternating increase and decrease of the local lift force, which depends on the propellers sense of rotation. For the present case, the effects are a results of the propellers rotating in an outboard-up direction. The interference effects are in good

[1] The positive 2.5g case is simulated under simplified, steady-state conditions. The underlying assumptions are explained in the respective paragraph further below.

agreement to existing literature [4], showing high resemblance to the qualitative load distribution illustrated in Fig. 1.4 of the literature review. As with all the other simulations, the lift trim routine was engaged during these two runs. When propellers are shut off, the wing has to maintain an angle of attack of 3.23° to achieve the required cruise lift, compared to only 3.14° if the propellers are operational.

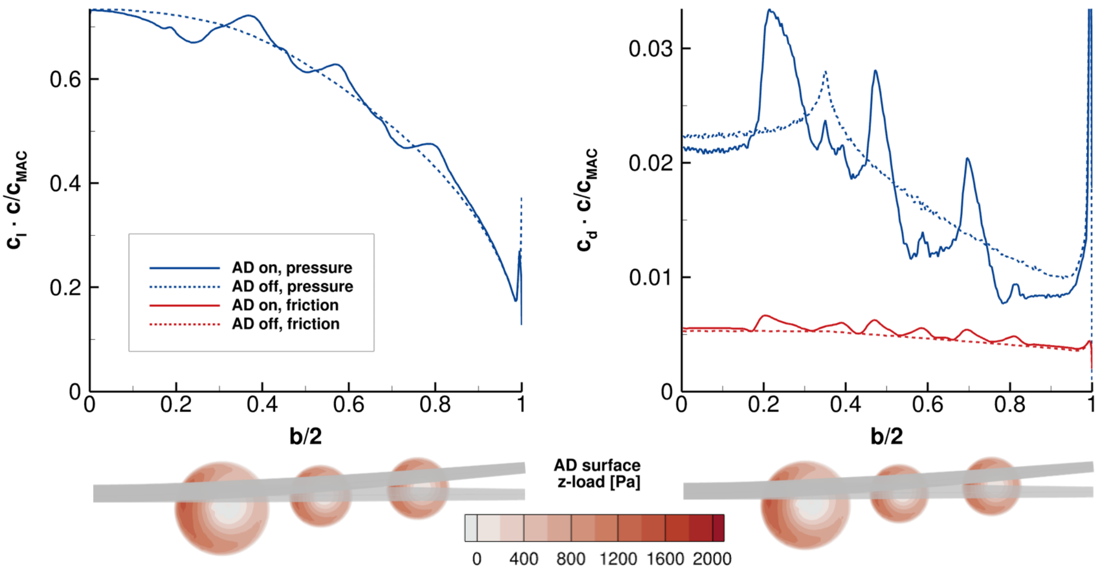


Figure 5.2.: ADs active versus inactive: AD surface load and spanwise lift and drag distributions at cruise conditions

The diagram on the right-hand side in Fig. 5.2 b shows the corresponding drag distributions, which reveal a significant propeller influence. In particular the pressure component of the drag is characterised by three prominent peaks at the spanwise locations at which the propellers' disk load, and therefore its induction flow, is the highest. However, for the wing segments which are not directly subjected to the propellers' slipstream, the propeller-off case experiences a higher pressure drag. The elevated drag profile can be explained with the slightly higher angle of attack needed when the propellers are idle. The friction-induced drag component, which has a less substantial contribution to the total drag altogether, is also less affected by propeller interference. The distribution shows small areas of increased friction drag at locations close to the propellers' outer rim. These locations coincide with the peaks of the radial load distribution of the propeller which induce a higher local flow velocity, raising friction drag. Note: The actuator disk z-load, illustrated in Fig. 5.2, is defined in the disk-local coordinate system in which z denotes the propeller's thrust axis.

Another distinct feature of the spanwise drag distribution exhibited by both the AD-on and the AD-off case is the pressure drag spikes at the kink-location and at wing tip. The two anomalies also appear in the pressure coefficient plots shown in Fig. 5.3. Around the kink's location, this results in a diminished suction peak; at the wing's tip, a vortex is shed from tip profile's edge lowering the pressure near the trailing edge. In both locations the geometry is characterised by sharp edges, impairing the wing's aerodynamic efficiency.

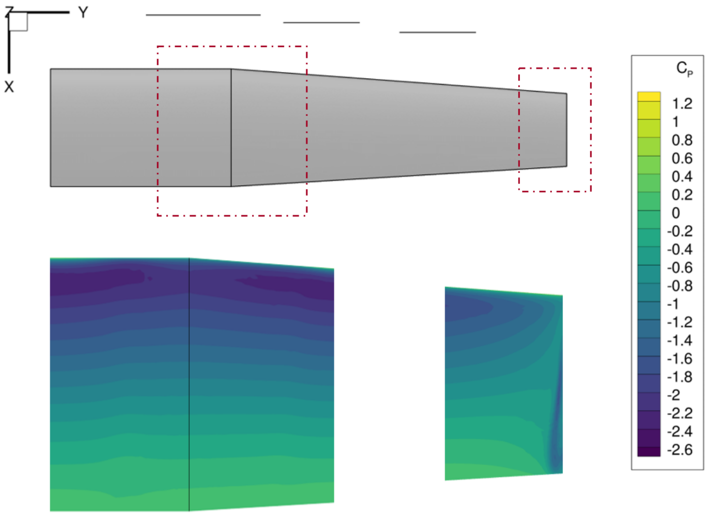


Figure 5.3.: Pressure coefficient C_P at the kink and the tip of the wing (2.5g case)

Under a load of 1g at cruise conditions, the structural deformation of the wing is characterised by an upward deflection, with a tip displacement of 847 mm in z-direction, corresponding to 6.3% of the half-span, and small negative (nose-down) tip rotation of -0,28° around the y-axis. The wing surface has a drag coefficient C_D of 0.02944, achieving a lift-to-drag ratio (LoD) of 24.99, which is only fractionally less than the rigid geometry which reached an LoD of 25.03. Correspondingly, the load distribution curves shown in Fig. 5.4, in which the coupled CFD-CSM results are compared with the CFD-only simulations, deviate only marginally from each other. A noticeable difference between the two cases is that the flexible wing's centre of lift is slightly shifted inboard. This agrees with the expected effect of a deflecting wing which causes the aerodynamic force at the outer wing sections to rotate inwards. Analogous to the shifted lift distribution, the drag is increased on the inner wing and slightly reduced for further outboard wing sections. Overall, for the comparatively stiff structural model at hand, the influence of the aeroelastic effects on the load distributions appears relatively small, especially when compared to the propeller interference. Although, it should be taken into account that deviations in the lift distributions are to some degree masked by the lift trim trying to maintain the total lift force. The drag distributions reveal the difference more distinctly since the drag is affected by the trimming-induced changes in angle of attack. Moreover, the employed structural model exhibits little to no twisting deflection under load. Twist angle variations have a profound influence on the wing's load distribution (also shown in the following section 5.3). However, by being almost unswept, this wing geometry is less susceptible to bending-twist coupling. Additionally, the particular design of the structural model gives this wing a high torsional stiffness, further reducing its tendency to twist.

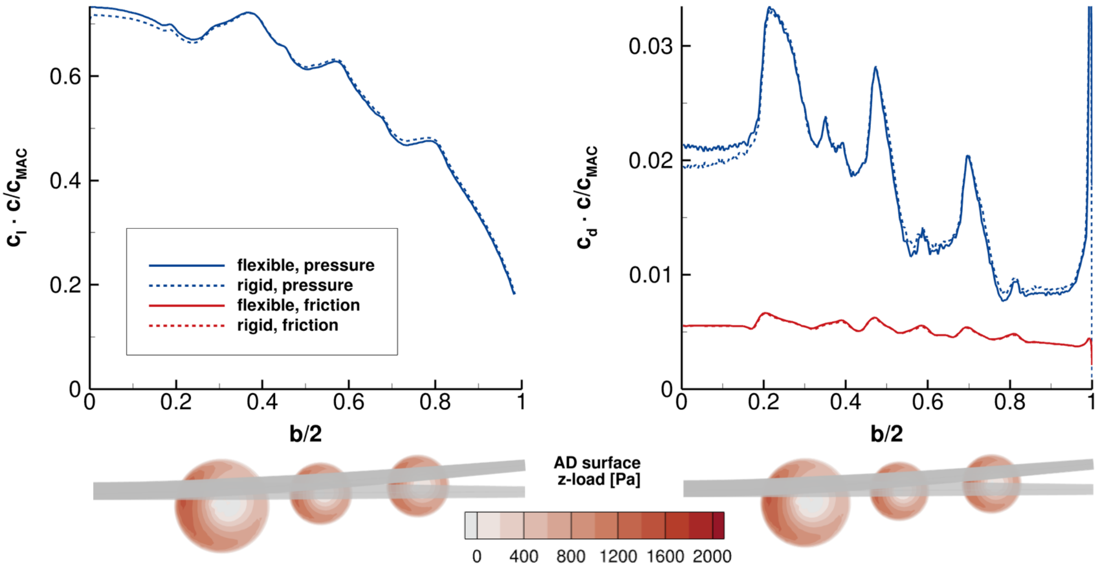


Figure 5.4.: Wing under 1g-load: AD surface load and spanwise lift and drag distributions

The 2.5g load case is simulated under steady-state conditions, despite it representing a dynamic maneuver. To allow for this simplification, it is assumed that the aircraft is at the low point of a circular trajectory representing a 2.5g downwards acceleration, resulting in the gravitational force being at a 90° angle to the current inflow direction. The radius of the circular trajectory is assumed to be large enough so that the maneuver can be approximated using steady boundary conditions. Different to all other conducted CFD analyses, this case is simulated at sea-level altitude (using standard-atmospheric conditions) and at a reduced airspeed of 0.35 Ma. This is done to decrease the required lift coefficient. In preparation for the coupled simulation, a CFD-only run was conducted in which the angle of attack was trimmed so that the wing generates a lift force equaling 2.5 times the aircraft's weight. This angle of attack setting is used in the subsequent CFD-CSM run to reorient the vector of the gravitational force in the parameter input for the structural solver. During the coupled simulation, this vector is kept unchanged, neglecting a small error introduced by the angle of attack changes happening during the CFD-CSM run. The magnitude of the weight forces in the structural solver settings is also multiplied by a factor of 2.5.

With this setup, the wing exhibits a maximum displacement at the tip of 2.2 m and a small tip twist angle of +0.33°. The resulting load distributions are shown in Fig. 5.5. The tip displacement equates to approx. 16% of the half-span which is generally considered above the limit for the geometric linearity assumption utilised in these simulations. One error introduced through this simplification is that the wing geometry begins to stretch under growing displacements, progressively increasing its surface area. In this case, the wing gained 1.6% in area as a consequence of the geometrically linear calculation. The increasing wing area could also explain why the flexible wing requires a smaller angle of attack (8.76°) than the reference case of the rigid geometry (8.92°) while achieving the same lift coefficient. Since the reference surface area used by *TAU* to non-dimensionalise the loads is fixed, the resulting coefficients are slightly overpredicted causing the lift trim algorithm to lower the angle of attack. Another potential reason for the lower angle of attack could be the slight positive (nose-up) twist, observed at the wing tip, which locally increases the effective angle of attack. The resulting lift distribution of the flexible wing (shown in Fig. 5.5) is again slightly shifted towards the inner wing sections, similar to the 1g case. Analogous to the lift distribution, the drag is marginally higher than the rigid geometry on the inner wing and lower on the outboard portions. Overall, the flexible wing has a drag coefficient C_D of 0.0627, resulting in a lift-to-drag ratio of 19.24 compared to only 18.92 for the rigid wing.

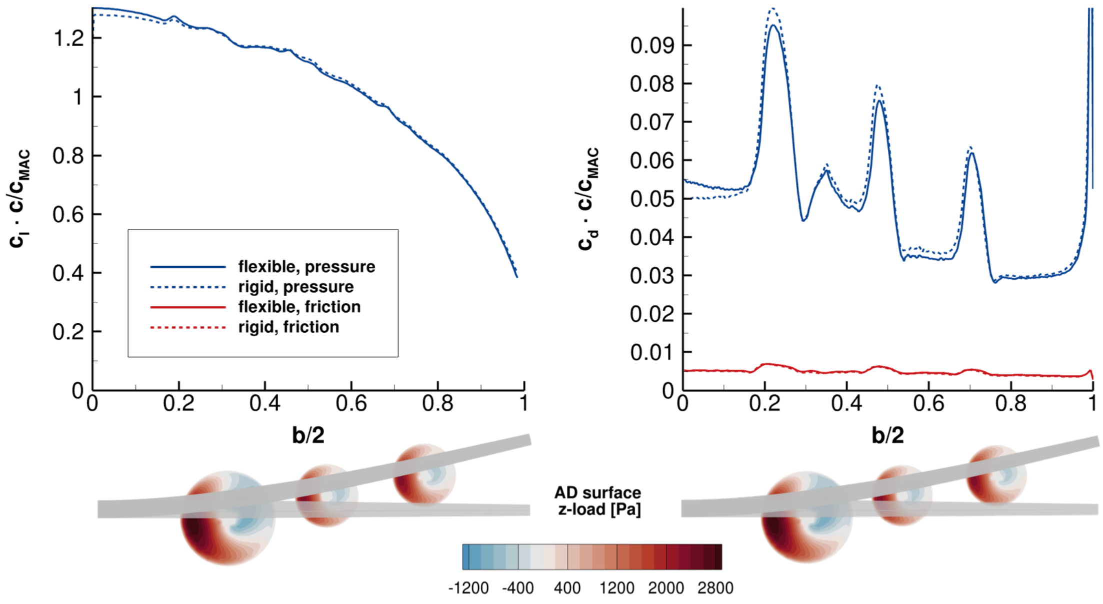


Figure 5.5.: Wing under 2.5g-load: AD surface load and spanwise lift and drag distributions

Another noticeable effect of the high angle of attack can be seen for the actuator disk loads. The outboard parts of the disk surface have begun to generate negative thrust while the positive loads on the inboard side (down-going blade) significantly increased compared to the 1g case to compensate for the losses. However, the calculated propulsive efficiencies of the ADs for the 2.5g case turn out higher than 100%, exceeding the physically possible limit. This shows the limitations of the utilised propeller data under off-design conditions for the reasons stated in Sec. 5.1.1.

In conclusion, the comparative analysis revealed static aeroelastic effects despite the high torsional stiffness of the wing and outlined the interference effects caused by the propellers. In general, the pure CFD simulations slightly underestimate the wing drag given a constant lift coefficient. The results are also summarised in Table 5.1 below. The 2.5g case highlighted the limitations of this approach emerging for high wing loads and deformations and under off-design conditions. Nonetheless, the implemented setup for the AD-integrated CFD-CSM simulations remained operational. This confirms the tolerance of this setup for more flexible wing models, which can be important in the context of a potential application in an optimisation scenario. In order to exploit the optimisation's full potential, the setup should be able to explore a given design space as best as possible.

	1g		*2.5g*	
	rigid	*flexible*	*rigid*	*flexible*
angle of attack (alpha)	3.01°	3.14°	8.93°	8.76°
lift coefficient	0.6761	0.6762	1.2090	1.2092
drag coefficient	0.02704	0.02701	0.06389	0.06505
lift-to-drag ratio	25.03	24.99	18.92	18.57
wing tip z-displacement	–	847 mm	–	2217 mm
wing tip y-rotation	–	-0.28°	–	0.33°

Table 5.1.: Results of the coupled CFD-CSM and the CFD-only simulations

5.3. Design Variations

One of the main benefits of the automated toolchain established in this work is its ability to efficiently generate design variations of the embedded model's geometry. To demonstrate this feature, six wing and propeller parameters were selected and varied to examine their effect on the design's aerodynamic performance. In case of the wing, the aerodynamic performance is measured using the lift-to-drag ratio (LoD) and in case of the propellers by evaluating their propulsive efficiencies (calculated using eq. 2.13). The parameters selected for this analysis are: the wing's tip twist angle, the propeller radii, angles of incidence, and vertical positions, the propellers' sense of rotation and their thrusts. The parameter values for the variations are given below in Table 5.2.

For each selected parameter, two design variations are generated: one with an increased value and another one with a decreased value while the remaining parameters are kept constant to the baseline state, which serves as reference. The baseline's design parameters are specified in Tables 3.1 and 3.2. In case of the three propeller-specific parameters (radius, angle of incidence, z-position), the parameters are varied in groups of three, for all ADs simultaneously. Using the automated mesh-generation setup, a new CFD mesh is generated for each parameter variation. While the parameter changes affecting the propeller position are accounted for by an adaption of the respective CSM mesh nodes, other modifications to the structural model are not made (see also Sec. 3.3). This implies, for example, that regardless of the wing twist distributions the same structural mesh is employed. Prior to the coupled CFD-CSM simulation conducted for this analysis, lift and thrust trim cycles (equal to the procedure described in Sec. 5.1.1) had to be completed for each design variant.

The following discussion about all the variants' performances selectively outlines the most impactful effects of the specific parameter changes, focusing on the highest sensitivities. Diagrams showing the entirety of the aerodynamic and propulsive efficiency results can be found in Appendix D.

Var01, Var02 | Wing Tip Twist Angle Variation

The tip twist angle is a parameter which affects the twist distribution of the wing. For this specific CAD model, the twist is applied linearly from the kink-location outwards up to the tip. A positive twist angle change represents a positive y-rotation of the tip profile, similar to a positive angle-of-attack variation.

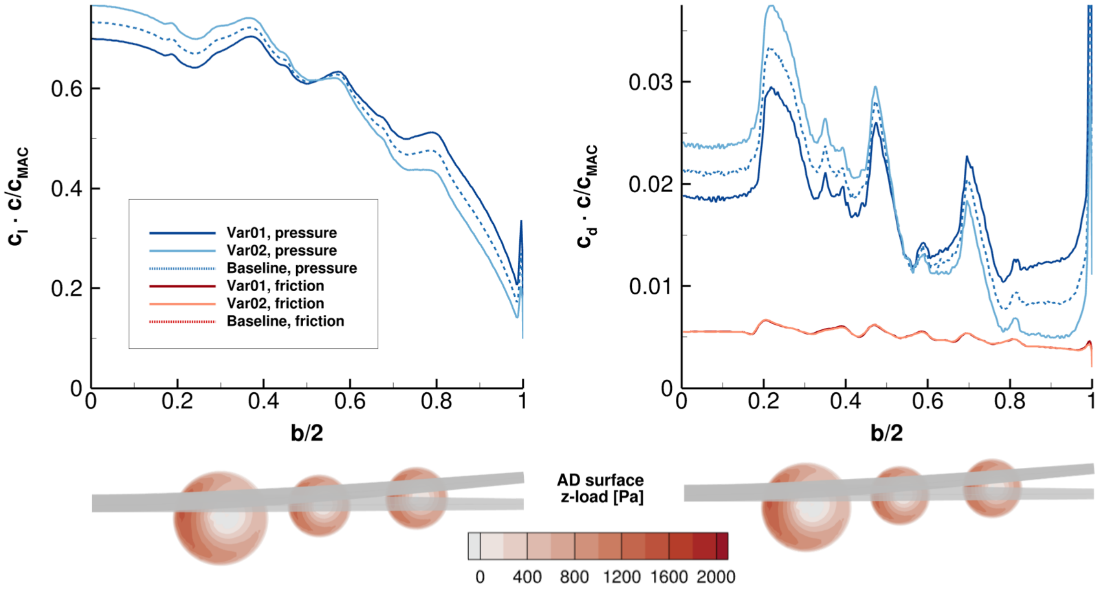
Twist is a variable utilised in wing design for adjusting a wing's lift distribution into a beneficial shape. This twist angle effect can be recognised in the lift distribution plot shown in Fig. 5.6: a large negative twist angle causes the local lift to decrease faster towards the outer wing (*Var02*), whereas the untwisted wing (*Var01*) generates more lift on the outboard wing sections. Even though it appears as if the wing twist changes the wing load on the inner wing section (up to approx. 35% of the half-span), the difference in local lift is caused indirectly by the lift trim routine. The angle of attack for *Var01* is 2.73°, whereas *Var02* requires an angle of 3.56° to achieve the same lift coefficient. This does also impact the two designs' aerodynamic efficiencies. While *Var01's* lift-to-drag ratio of 25.0 is approximately similar to the baseline, *Var02* only reaches an LoD of 24.7. The propulsive efficiencies, on the other hand, are basically unaffected by this parameter variation.

Var03, Var04 | Propeller Radius Variation

Larger propellers are generally attributed to higher propulsive efficiencies as the blade loading is reduced because thrust is distributed over a larger area. A similar result can be

	Parameter	*Baseline*	*Variation +*		*Variation -*	
	wing tip twist angle	-2°	*Var01*	0°	*Var02*	-4°
AD1	radius	1.5 m	*Var03*	1.65 m	*Var04*	1.35 m
	angle of incidence	0°	*Var05*	5°	*Var06*	-5°
	z-position	-0.5 m	*Var07*	0.0 m	*Var08*	-1.0 m
AD2	radius	1.0 m	*Var03*	1.1 m	*Var04*	0.9 m
	angle of incidence	0°	*Var05*	3°	*Var06*	-3°
	z-position	-0.3 m	*Var07*	0.0 m	*Var08*	-0.6 m
AD3	radius	1.0 m	*Var03*	1.2 m	*Var04*	0.8 m
	angle of incidence	0°	*Var05*	3°	*Var06*	-3°
	z-position	-0.3 m	*Var07*	0.2 m	*Var08*	-0.8 m
	props.' sense of rotation	outboard-up	*Var09*			inboard-up
	total prop. thrust	8.0 kN	*Var10*	10.0 kN	*Var11*	6.0 kN

Table 5.2.: Design variations



a) Lift distribution

b) Drag distribution

Figure 5.6.: Load distributions: wing tip twist angle variations vs. baseline (Var01: 0° twist angle; Var02: -4° twist angle)

seen for the present design variations. Compared to the baseline design, the two variants (*Var03* and *Var04*) have the propeller radii of AD1 and AD2 either increased or decreased by 10%; or, in case of the outer propeller (AD3), even varied by 20%. To realise the size variations, two new meshes had to be generated using the automated mesh generation process. The rotation speeds of the propellers were adjusted based on their radius so that their tip equals 0.75 Ma. The large propeller variant (*Var03*) in Fig. 5.7 achieves higher propulsive efficiencies compared to the other one. The smallest disk of *Var04*, AD3, shows a significant drop in efficiency. Due to its small size, it was no longer able to generate the required thrust. Even an increase of AD3's rotational speed to a tip speed Ma_{tip} of 0.8 was not able to achieve the required thrust output. So, for this particular disk, the thrust trim converged on the blade pitch value for (lower) maximum thrust which represents a sub-optimal operating condition. The wing's efficiency (see Fig. 5.7 left) is noticeably affected by radius changes, with the smaller propeller design achieving a higher performance than the baseline. This is likely because a smaller portion of the wing is submerged in the propellers' wakes, as can be inferred from the slightly narrower pressure drag spikes, shown in Fig. 5.8.

Var05, Var06 | Angle of Incidence Variation

The propeller disk's angle of incidence directly affects the direction of the propeller's thrust vector. In this case, an increased angle of incidence refers to a more upward facing

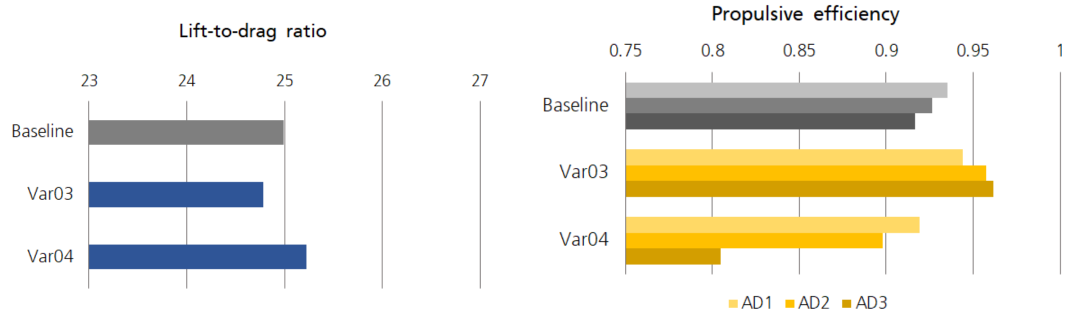
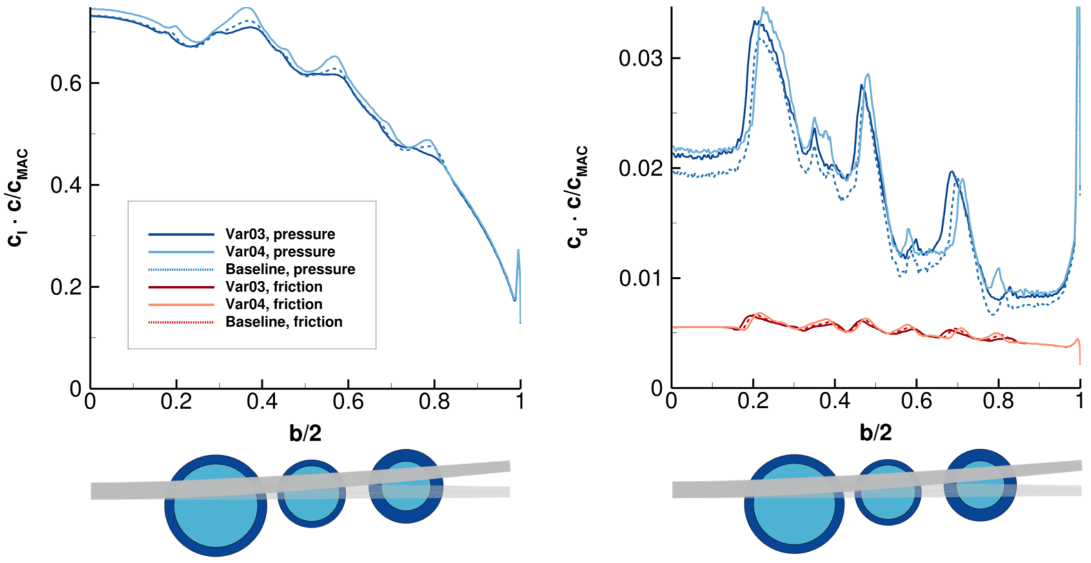


Figure 5.7.: Aerodynamic and propulsive efficiency for larger (Var03) and smaller propellers (Var04) compared to baseline



a) Lift distribution

b) Drag distribution

Figure 5.8.: Load distributions: propeller radius variations vs. baseline (Var03: large propeller radius; Var04: small propeller radius)

propeller and a negative angle to a downwards rotation, with 0° leading to a parallel alignment with the aircraft's longitudinal axis. Rotating the propeller upwards augments the generation of lift, but it also causes higher local angle of attack fluctuations at the propeller blades, which are theoretically detrimental to the propulsive efficiency (see also Sec. 2.2). With the propeller input data used in the analysis, an efficiency gain can be seen for the larger angle of incidence of *Var05* (see Fig. 5.9). This gain is likely not physically accurate especially since AD1 achieves an efficiency of around 100%. The lower angle of incidence variant shows a slightly reduced propulsive efficiency but a distinctive increase for the aerodynamic efficiency of the wing. This is likely caused by a stronger flow induction on the upper side of the wing, augmenting its lift generation.

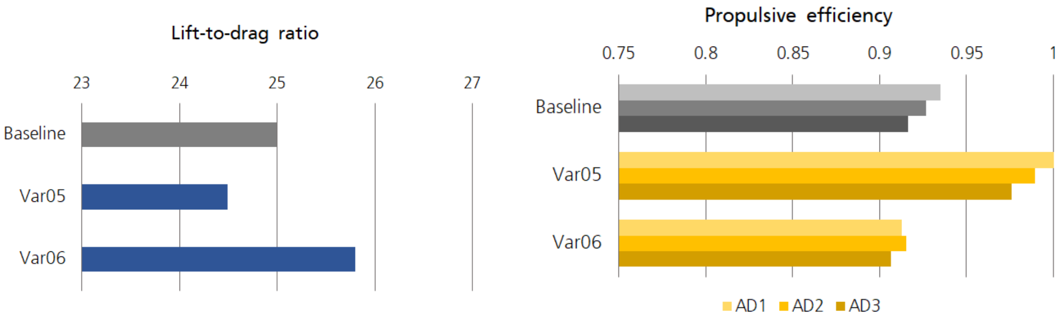
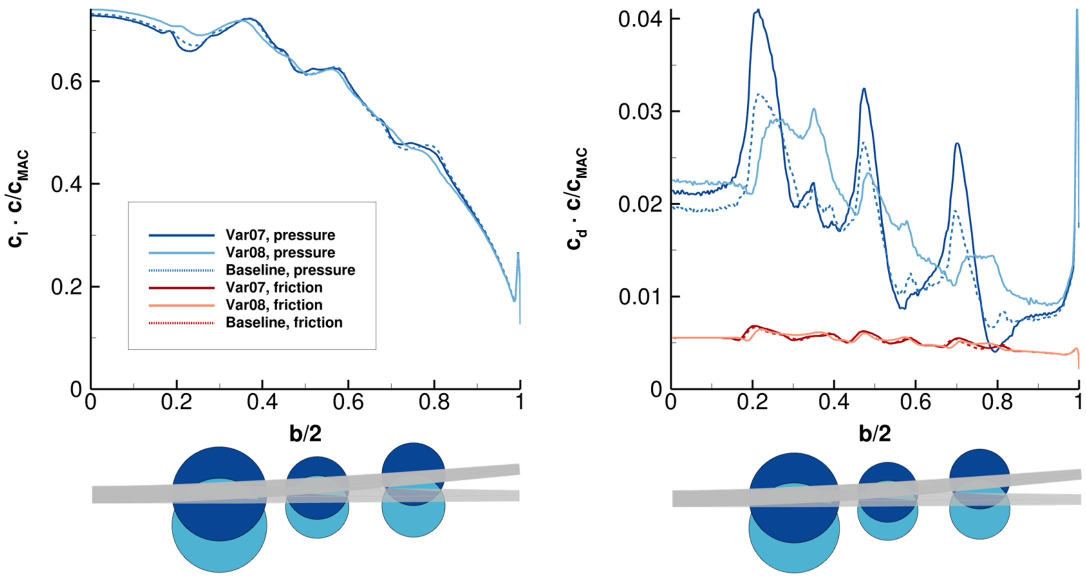


Figure 5.9.: Aerodynamic and propulsive efficiency for high propeller disk angle of incidence (Var05) and low angle of incidence (Var06) compared to baseline

Var07, Var08 | Variation of the Propeller Z-Position

The vertical propeller position affects the propeller-wing interaction by changing the local flow angle as well as the wing's level of immersion in the wake flow, as explained in the literature review (see Sec. 1.2). Therefore, wing aerodynamics are more sensitive to changes of the propeller position in vertical than in spanwise direction [4]. Accordingly, the lift distribution of the variant with low propeller positions (*Var07*), shown in Figure 5.10 left, appears slightly smoother, meaning less disturbed, than the curve for *Var08*. While the variants in lift are comparatively small, distinctive differences can still be observed for the drag distributions of these variants. The high propeller positioning amplified the peaks in the drag distribution, whereas the lowered propeller positions caused the peaks to be dispersed over a wider wing section. The resulting propulsive efficiencies obtained for the two variants are nearly identical. Despite these results, the differences in the overall wing efficiency observed for the two corresponding designs are almost negligible: With a value of 24.9, the high propeller positions of *Var07* is equal to the lift-to-drag ratio measured for the baseline, whereas the low propeller position variant (*Var08*) shows only a minor increase to 25.1.



a) Lift distribution

b) Drag distribution

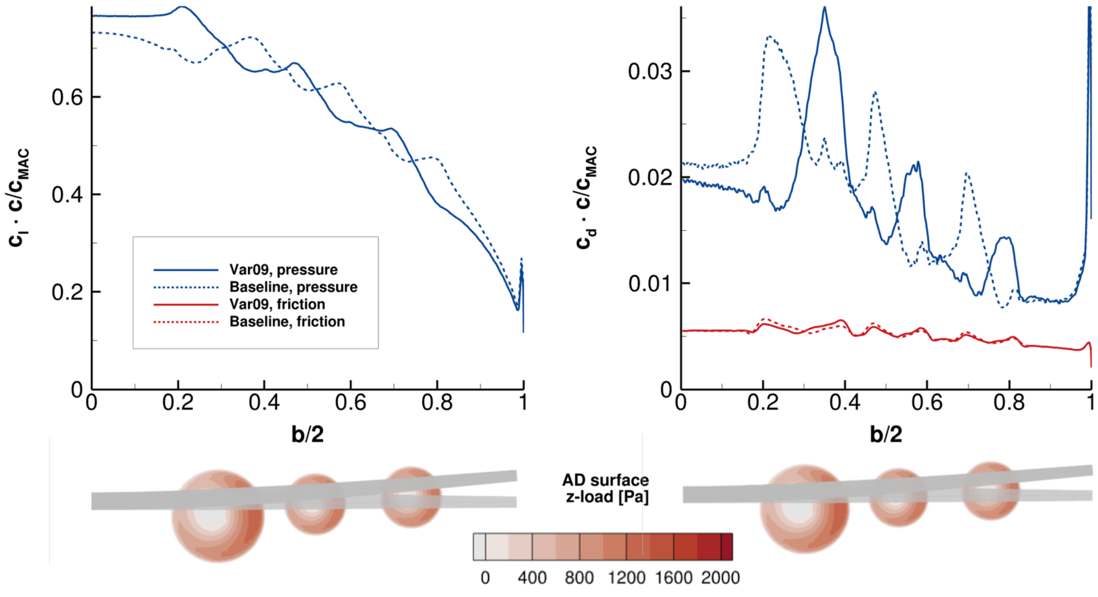
Figure 5.10.: Load distributions: propeller z-position variations vs. baseline (Var07: high propeller position; Var08: low propeller position)

Var09 | Variation of the Propellers' Sense of Rotation

The employment of *Blade Element Theory* for calculating the virtual propeller blade loads on the actuator disk boundaries enables the embedding of propeller swirl into the flow simulation. For that, the propellers' sense of rotation has to be defined. The baseline design features outboard-up (or clockwise) rotating propellers. Prior research on propeller-wing interaction has shown that the rotational direction of the propellers has a great impact on the wing's load distribution [4]. The results indicate that the aerodynamic efficiency is negatively affected by an outboard-up rotation. In agreement with these findings, *Var09*, for which the rotation has been inverted to an inboard-up motion, sees a profound efficiency improvement with an LoD of 26.4 as compared to the baseline value of 25.0. As shown by the load distribution plots in Fig. 5.11a, the lift curves of the inboard-up rotating propellers exhibit an inverted undulation which causes the centre of lift to shift towards the aircraft's middle axis. Analogous to that, the distinct spikes of the drag distribution in Fig. 5.11b are inverted as well and now align with the outboard portion of the propeller disk where the maximum thrust load is generated.

Var10, Var11 | Propeller Thrust Variation

Varying the propeller thrust changes the induced velocities in its slipstream flow, which raises the aerodynamic load on the wing sections immersed in the slipstream. This means



a) Lift distribution

b) Drag distribution

Figure 5.11.: Load distributions: inboard-up (Var09) and outboard-up (baseline)

that not only the lift but also the drag force increases, especially in case of the present tractor configuration (propeller upstream of the wing) investigated in this analysis. Even though, both variants show a relative improvement of aerodynamic efficiency compared to the baseline, as shown in Fig. 5.12, the lower thrust variant (*Var11*) achieves the higher LoD value of 25.5, caused by a drag reduction. The propulsive efficiency is highest for the variant with the lowest thrust as this consequently results in the lowest disk loading.

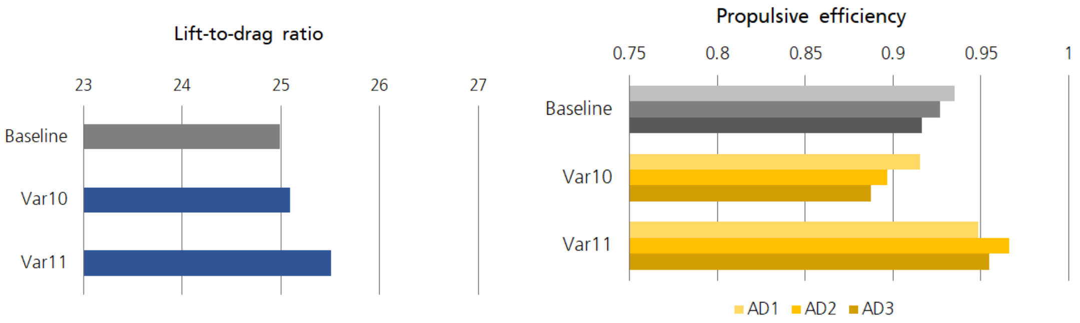


Figure 5.12.: Aerodynamic and propulsive efficiency for a high thrust (Var10: 10 kN) and a low thrust setting (Var11: 6 kN)

5.3.1. Summary

The results of the design variation analysis highlight how strongly the wing is affected by propeller interference. The largest improvement in aerodynamic efficiency compared to the baseline design was achieved by changing the propellers' sense of rotation to an inboard-up direction which raised the lift-to-drag ratio (LoD) from 25.0 to 26.4. Reducing the propeller thrust, also has a beneficial effect on wing efficiency: A thrust reduction from 8 to 6 kN led to the second-highest LoD improvement of around 2%, caused by a reduction in wing drag. The angle of incidence variation, on the other hand, indicates that the propeller slipstream can augment lift generation if the wake flows, at least partly, over the wing. A slight downwards angled propeller axis (negative incidence angle) generated a higher lift force while the aerodynamic efficiency increased as well.

The resulting propulsive efficiencies generally less impacted by most of the parameter variations done in this analysis. Gains in propulsive efficiency can be seen for the variant with bigger propeller radii (*Var03*) and lower thrust setting (*Var11*). As both cases involve lowered disk loading, these results seem plausible. However, the biggest increase in propulsive efficiency is achieved by (*Var05*) which had a positive propeller incidence angle. From the consequently higher inflow angle a reduction in efficiency would be expected. As similar efficiency gains could be seen for the 2.5g case (see Sec. 5.2), which also involved high local angles of attack at the propeller disk, the validity of the propeller data under off-design conditions is questionable.

The setup of this analysis does not allow for a definitive conclusion on the aerodynamic performance. Since only one parameter was varied at a time, potential cross-correlations between parameters are out of the scope of this approach. The results indicate, however, that these interaction effects are present and highlight the need for a multivariate, multidisciplinary design optimisation.

Regardless of the specific results, this part of the analysis confirms the usability of the AD-integrated CFD-CSM setup with regard to future use cases such as trade-off studies and design optimisations. The analysis showcased that the setup is flexible and robust against design variations introduced by a potential optimisation algorithm.

6. Conclusion and Outlook

The present work successfully established a modeling toolchain for fluid-structure coupled analyses (CFD-CSM) of aircraft with distributed propulsion using the actuator disk model (AD) to simulate the propellers. For that, the AD model is incorporated into the *FlowSimulator* software suite, used by industry and DLR (German Aerospace Center) as a framework for high-fidelity multidisciplinary analysis. The integration of the AD model's capability is achieved with an additional software module that ensures the transfer of coupling data between the structural domain and the CFD domain. The software is implemented in the *FSDLRControl* suite, a FlowSimulator plugin developed by DLR for steering CFD-CSM simulations. This module is executed in each cycle of the coupled CFD-CSM simulation, updating the positions and orientations of all actuator disk boundaries in the CFD mesh after a structural solution has been calculated. In combination with fixes to identified issues in the *FlowSimulator DataManager* (FSDM) implementation related to the actuator disk model, the expansion of *FSDLRControl* provides the core functionality which allows the AD model to be used in coupled CFD-CSM simulations. Moreover, in order to set up the toolchain for a future application in optimisation scenarios, this work established an automated CFD mesh generation process based on a parametric CAD wing model. The wing design is derived from a concept developed in the context of the EU-project HERA, which investigates novel architectures for regional aircraft with hybrid propulsion. Despite the adoption of this concrete reference aircraft, the design is only used as a provisional test case, serving an exemplary purpose. The toolchain is set up so that the implemented models are easily exchangeable, allowing it to be applicable to other use cases. In addition to adaptability, robustness was a central aspect of the toolchain's development. Therefore, all of the chain's components have been thoroughly tested using wide-ranging parameter inputs, confirming that they remain functional even under challenging boundary conditions.

Deploying the AD-integrated CFD-CSM setup, two types of analyses were conducted using the reference aircraft design from HERA. The first one investigated fluid-structure interactions by comparing the solutions from coupled CFD-CSM simulations with results obtained from monodisciplinary CFD simulations of the rigid wing geometry. The results showed that for the given wing model the aeroelastic effects are measurable but smaller than the influence of the propeller interference. The propeller flow strongly affects the wing aerodynamics by locally increasing lift as well as drag forces on the wing surface. The second analysis studied the sensitivities of the reference aircraft's wing with respect to its aerodynamic performance by varying six design parameters and measuring the effect on aerodynamic and propulsive efficiency. The results revealed a significant influence of numerous design parameters on the wing's performance. The magnitude of the propeller-wing interference is highly sensitive to the wing's level of immersion in the propeller slipstream, which in turn depends on the specific geometry of the configuration as well as operational settings of the propellers. Moreover, since the propeller-wing interaction also depends on the aircraft's angle of attack, it has an effect on the aircraft's trim state. In particular, the interdependencies between lift, angle of attack and propeller thrust underscore the necessity for a dedicated trim algorithm for these types of propeller configurations.

An important finding while utilising the actuator disk model for coupled CFD-CSM simulations is that the surface area of any disk boundary in the CFD mesh expands as a result of the wing deflection. This undesired behaviour is a consequence of the specific way in which *FlowSimulator's* displacement projection technique handles rotations. The currently available projection method approximates rotations by linearly transforming angles into displacements. The introduced error is negligible for small angles. In case of the actuator disk boundaries, however, a significant area expansion was observed even for moderate wing displacements, as the disks follow the rotation of the deflecting wing. An improvement to the workaround used in this work would be to amend the current displacement projection method with a geometry-preserving non-linear operation suited for rigid-body rotations.

Another potential improvement to the AD-coupling software module mentioned at the beginning would be to more tightly integrate the exchange of data required by the actuator disk model. Since the exchanged information is purely geometric, it is readily available in the respective mesh data and therefore accessible from within *FlowSimulator's* core/data layer (FSDM). After the CFD mesh deformation is completed, the positions and orientations of actuator disk boundaries could be computed automatically using the deformation state stored in the mesh data. This would make retrieval of user-input, currently required for the module to operate, obsolete and simplify the employment of the actuator disk model in coupled CFD-CSM simulations using *FlowSimulator*, which would also make the model more accessible.

Capable computational techniques for modeling the aerodynamics of propeller aircraft are important, considering for example the large design flexibility that has emerged as a result of the advent of more electrified propulsion architectures (i.e. distributed electric propulsion). Multidisciplinary analysis and optimisation (MDA/O) represents an effective methodology for investigating these large design spaces. By integrating the actuator disk model into the established MDA/O framework, this work makes a valuable contribution to DLR's modeling toolbox, extending its capabilities for propeller aircraft simulations. In the future, the process developed in this work can be expanded with a parametric structural model and integrated into a larger MDO toolchain, allowing it to be used in trade-off studies and gradient-based optimisation scenarios. In order to identify ever more efficient aircraft designs within narrowing margins for improvement, multidisciplinary approaches are essential for exploiting the available potential. In the light of progressing global warming, the aviation industry faces the challenging task to drastically reduce its greenhouse-gas emissions in the upcoming decades. Computational modeling and optimisation techniques can facilitate this imminent transition by minimising the amount of required design cycles during development. With that, these techniques help to save associated costs and bring aviation closer towards a climate-compatible future.

Bibliography

[1] D.S. Lee et al. "The contribution of global aviation to anthropogenic climate forcing for 2000 to 2018". In: (). DOI: 10.1016/j.atmosenv.2020.117834.

[2] HERA Consortium, ed. *HERA Project Homepage: Designing the third-era of Aviation*. 2024. URL: https://project-hera.eu (visited on 03/15/2024).

[3] Ludwig Prandtl. *Mutual influence of wings and propeller*. 1921.

[4] L.L.M. Veldhuis. "Propeller Wing Aerodynamic Interference". Dissertation. Delft: Technische Universiteit Delft, 2005.

[5] Snorri Gudmundsson, ed. *General Aviation Aircraft Design*. Boston: Butterworth - Heinemann, 2014. ISBN: 978-0-12-397308-5.

[6] Axel Raichle. *Flusskonservative Diskretisierung des Wirkscheibenmodells als Unstetigkeitsfläche*. Ed. by Deutsches Zentrum für Luft- und Raumfahrt. 2017.

[7] By Alan Buis NASA's Global Climate Change Website. "A Degree of Concern: Why Global Temperatures Matter". In: (19.06.2019). URL: https://climate.nasa.gov/news/2865/a-degree-of-concern-why-global-temperatures-matter/.

[8] P. R. Shukla, Jim Skea, and Andy Reisinger, eds. *Climate change 2022: Mitigation of climate change*. Geneva: IPCC, 2022. ISBN: 9789291691609. URL: https://www.ipcc.ch/report/ar6/wg3/downloads/report/IPCC_AR6_WGIII_FullReport.pdf.

[9] IEA. *Global CO2 emissions from transport by sub-sector in the Net Zero Scenario, 2000-2030*. Ed. by IEA. Paris, 14.06.2023. URL: https://www.iea.org/data-and-statistics/charts/global-co2-emissions-from-transport-by-sub-sector-in-the-net-zero-scenario-2000-2030-2.

[10] International Energy Agency. *Energy Technology Perspectives 2020*. 2020.

[11] CAPA - Centre for Aviation, ed. *Global aircraft order backlog hits 13 years. Airbus beats Boeing on key metrics*. 2024. URL: https://centreforaviation.com/analysis/reports/global-aircraft-order-backlog-hits-13-years-airbus-beats-boeing-on-key-metrics-672470.

[12] Steven J. Davis et al. "Net-zero emissions energy systems". In: *Science (New York, N.Y.)* 360.6396 (2018). DOI: 10.1126/science.aas9793.

[13] D02 Committee. *Specification for Aviation Turbine Fuels*. West Conshohocken, PA, 2023. DOI: 10.1520/D1655-22A.

[14] Quan Li et al. "The road towards high-energy-density batteries". In: *The Innovation Energy* 1.1 (2024), p. 100005. ISSN: 3006-418X. DOI: 10.59717/j.xinn-energy.2024.100005.

[15] Alexander Bills et al. “Performance Metrics Required of Next-Generation Batteries to Electrify Commercial Aircraft”. In: *ACS Energy Letters* 5.2 (2020), pp. 663–668. ISSN: 2380-8195. DOI: 10.1021/acsenergylett.9b02574.

[16] D. Cecere, E. Giacomazzi, and A. Ingenito. “A review on hydrogen industrial aerospace applications”. In: *International Journal of Hydrogen Energy* 39.20 (2014), pp. 10731–10747. ISSN: 03603199. DOI: 10.1016/j.ijhydene.2014.04.126.

[17] Saurav Tiwari, Michael J. Pekris, and John J. Doherty. “A review of liquid hydrogen aircraft and propulsion technologies”. In: *International Journal of Hydrogen Energy* 57 (2024), pp. 1174–1196. ISSN: 03603199. DOI: 10.1016/j.ijhydene.2023.12.263.

[18] Wolfgang Grimme, Sven Maertens, and Axel Classen. *Market uptake and impact of key green aviation technologies: Final report.* First edition. Luxembourg: Publications Office of the European Union, 2023. ISBN: 978-92-76-99708-5.

[19] Council of the European Union, European Parliament. *Regulation 2023/2405 on ensuring a level playing field for sustainable air transport (ReFuelEU Aviation).* 2023.

[20] Matthias Braun, Wolfgang Grimme, and Katrin Oesingmann. “Pathway to net zero: Reviewing sustainable aviation fuels, environmental impacts and pricing”. In: *Journal of Air Transport Management* 117 (2024), p. 102580. ISSN: 09696997. DOI: 10.1016/j.jairtraman.2024.102580.

[21] L. Bock and U. Burkhardt. “Contrail cirrus radiative forcing for future air traffic”. In: *Atmospheric Chemistry and Physics* 19.12 (2019), pp. 8163–8174. DOI: 10.5194/acp-19-8163-2019. URL: https://acp.copernicus.org/articles/19/8163/2019/.

[22] Europäische Kommission. *Der europäische Grüne Deal: Erster klimaneutraler Kontinent werden.* Ed. by European Commission. 2024. URL: https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/european-green-deal_de.

[23] Clean Aviation. *Members.* 13.05.2024. URL: https://www.clean-aviation.eu/members.

[24] “Hybrid-Electric Regional Architecture”. In: (13.01.2023). URL: https://cordis.europa.eu/project/id/101102007/reporting.

[25] Clean Aviation. *Programme overview and structure.* 13.05.2024. URL: https://www.clean-aviation.eu/programme-overview-and-structure.

[26] Mark D. Moore. “Misconceptions of Electric Aircraft and their Emerging Aviation Markets”. In: *52nd Aerospace Sciences Meeting.* Reston, Virginia: American Institute of Aeronautics and Astronautics, 2014. ISBN: 978-1-62410-256-1. DOI: 10.2514/6.2014-0535.

[27] L. L. M. Veldhuis. “Review of Propeller-Wing Aerodynamic Interference”. In: *24TH INTERNATIONAL CONGRESS OF THE AERONAUTICAL SCIENCES* ().

[28] John W. Draper and Richard E. Kuhn. *Investigation of the aerodynamic characteristics of a model wing-propeller combination and of the wing and propeller.* 1956.

[29] L.L.M. Veldhuis. "Review of Propeller-Wing Aerodynamic Interference. ICAS 2004-6.3.1". In: *24th International Congress of the Aeronautical Sciences, Yokohama, Japan.* 2004.

[30] Sung-Eun Kim, Shin Hyung Rhee. "Efficient Engineering Prediction of Turbulent Wing Tip Vortex Flows". In: *Computer Modeling in Engineering & Sciences* 62.3 (2010), pp. 291–310. ISSN: 1526-1506. DOI: 10.3970/cmes.2010.062.291. URL: http://www.techscience.com/CMES/v62n3/25541.

[31] Angel Gomariz-Sancha, Moira Maina, and and Andrew J. Peace. "Analysis of propeller-airframe interaction effects through a combined numerical simulation and wind-tunnel testing approach". In: *53rd AIAA Aerospace Sciences Meeting 2015.* Red Hook, NY: Curran, 2015. ISBN: 978-1-62410-343-8. DOI: 10.2514/6.2015-1026.

[32] Shaka Chu et al. "Aerodynamic Analysis of Multi-Propeller/Wing Interaction using the Actuator Surface Model". In: *AIAA Scitech 2021 Forum.* Reston, Virginia: American Institute of Aeronautics and Astronautics, 2021. ISBN: 978-1-62410-609-5. DOI: 10.2514/6.2021-1661.

[33] Omkar Walvekar and Satyanarayanan Chakravarthy. "An Unsteady Reynolds–Averaged Navier–Stokes–Large Eddy Simulation Study of Propeller–Airframe Interaction in Distributed Electric Propulsion". In: *Aerospace* 11.1 (2024), p. 17. DOI: 3390.

[34] Michael Schollenberger et al. "Aerodynamic interactions between distributed propellers and the wing of an electric commuter aircraft at cruise conditions". In: *CEAS Aeronautical Journal* 15.2 (2024), pp. 255–267. ISSN: 1869-5582. DOI: 10.1007/s13272-023-00706-6.

[35] Shamsheer S. Chauhan and Joaquim R. R. A. Martins. "RANS-based aerodynamic shape optimization of a wing considering propeller-wing interaction". In: *AIAA Scitech 2020 Forum.* Reston, Virginia: American Institute of Aeronautics and Astronautics, 2020. ISBN: 978-1-62410-595-1. DOI: 10.2514/6.2020-1764.

[36] Lars Reimer et al. "Virtual Aircraft Technology Integration Platform: Ingredients for Multidisciplinary Simulation and Virtual Flight Testing". In: *AIAA Scitech 2021 Forum.* Reston, Virginia: American Institute of Aeronautics and Astronautics, 2021. ISBN: 978-1-62410-609-5. DOI: 10.2514/6.2021-1202.

[37] Norbert Kroll and Jens K. Fassbender, eds. *MEGAFLOW - Numerical Flow Simulation for Aircraft Design: Results of the second phase of the German CFD initiative MEGAFLOW, presented during its closing symposium at DLR, Braunschweig, Germany, December 10 and 11, 2002.* Vol. 89. SpringerLink Bücher. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg, 2005. ISBN: 9783540323822. DOI: 10.1007/3-540-32382-1.

[38] Lars Reimer. *FSNastranInterface: DLR GitLab Repository.* 2019. URL: https://gitlab.as.dlr.de/FSNastranInterface/FSNastranInterface (visited on 08/29/2024).

[39] Bernd Stickan, Hans Bleecke, and Silvio Schulze. “NASTRAN Based Static CFD-CSM Coupling in FlowSimulator”. In: *Computational Flight Testing: Results of the Closing Symposium of the German Research Initiative ComFliTe, Braunschweig, Germany, June 11th-12th, 2012*. Ed. by Norbert Kroll et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 223–234. ISBN: 978-3-642-38877-4. DOI: `10.1007/978-3-642-38877-4`.

[40] Lars Reimer. “Simulation Environment for CFD-based Aeroelastic Analysis of Aircraft”. Dissertation. Aachen: RWTH Aachen, 2020.

[41] Lars Reimer. *FSDLRCFDCSMInterpolSuite: DLR GitLab Repository*. 2024. URL: `https://gitlab.as.dlr.de/FSDLRCFDCSMInterpolSuite/FSDLRCFDCSMInterpolSuite` (visited on 08/29/2024).

[42] Carsten Braun. “Ein modulares Verfahren für die numerische aeroelastische Analyse von Luftfahrzeugen”. PhD thesis. Aachen, 2008. URL: `https://publications.rwth-aachen.de/record/50006`.

[43] Bernd Stickan. *User Guide FSDeformation Module*. 2020.

[44] Arne Rempke. *User Guide FSMeshDeformation Module*. Dresden, 2024.

[45] A. de Boer, M.S. van der Schoot, and H. Bijl. “Mesh deformation based on radial basis function interpolation”. In: *Computers & Structures* 85.11 (2007), pp. 784–795. ISSN: 0045-7949. DOI: `10.1016/j.compstruc.2007.01.013`. URL: `https://www.sciencedirect.com/science/article/pii/S0045794907000223`.

[46] Bernd Stickan. *Explanation of AEROSTABIL Limit-Cycle Osccillations via High-Fidelity Aeroelastic Simulations*. Ed. by Deutsches Zentrum für Luft- und Raumfahrt. Göttingen, 2018.

[47] Pedro Alves, Miguel Silvestre, and Pedro Gamboa. “Aircraft Propellers—Is There a Future?” In: *Energies* 13.16 (2020). ISSN: 1996-1073. DOI: `10.3390/en13164157`. URL: `https://www.mdpi.com/1996-1073/13/16/4157`.

[48] Gijs A. M. van Kuik. *The fluid dynamic basis for actuator disc and rotor theories*. Amsterdam: IOS Press, 2018. ISBN: 978-1-61499-866-2. DOI: `10.3233/978-1-61499-866-2-i`. URL: `http://ebooks.iospress.nl/book/the-fluid-dynamic-basis-for-actuator-disc-and-rotor-theories`.

[49] Robert W. Deters, Gavin K. Ananda, and Michael S. Selig. “Slipstream Measurements of Small-Scale Propellers at Low Reynolds Numbers”. In: *33rd AIAA Applied Aerodynamics Conference*. Reston, Virginia: American Institute of Aeronautics and Astronautics, 2015. ISBN: 978-1-62410-363-6. DOI: `10.2514/6.2015-2265`.

[50] Institute of Aerodynamics and Flow Technology, ed. *TAU-Code User Guide: Release 2018.1.0*.

[51] BETA CAE Systems. *ANSA pre-processor - BETA CAE Systems*. 2024-07-05. URL: `https://www.beta-cae.com/ansa.htm` (visited on 08/29/2024).

[52] María Rodriguez Montes. *D5.1.2B - 2035 First concept interim configuration UCB: Version 1.0*. Dec. 20, 2023.

[53] Alessandro Sgueglia. *D2.1.2 - First TLAR: Revision 1*. June 29, 2023.

[54] Patrick M. Knupp. “Algebraic Mesh Quality Metrics”. In: *SIAM Journal on Scientific Computing* 23.1 (2001), pp. 193–218. ISSN: 1064-8275. DOI: 10.1137/S1064827500371499.

[55] Stefan Biser et al. “Design Space Exploration Study and Optimization of a Distributed Turbo-Electric Propulsion System for a Regional Passenger Aircraft”. In: *AIAA Propulsion and Energy 2020 Forum*. Reston, Virginia: American Institute of Aeronautics and Astronautics, 2020. ISBN: 978-1-62410-602-6. DOI: 10.2514/6.2020-3592.

[56] Unknown Authors. *Certification Specifications for Large Aeroplanes CS-25: Amendment 3: Annex to ED Decision 2007/010/R*. 2007.

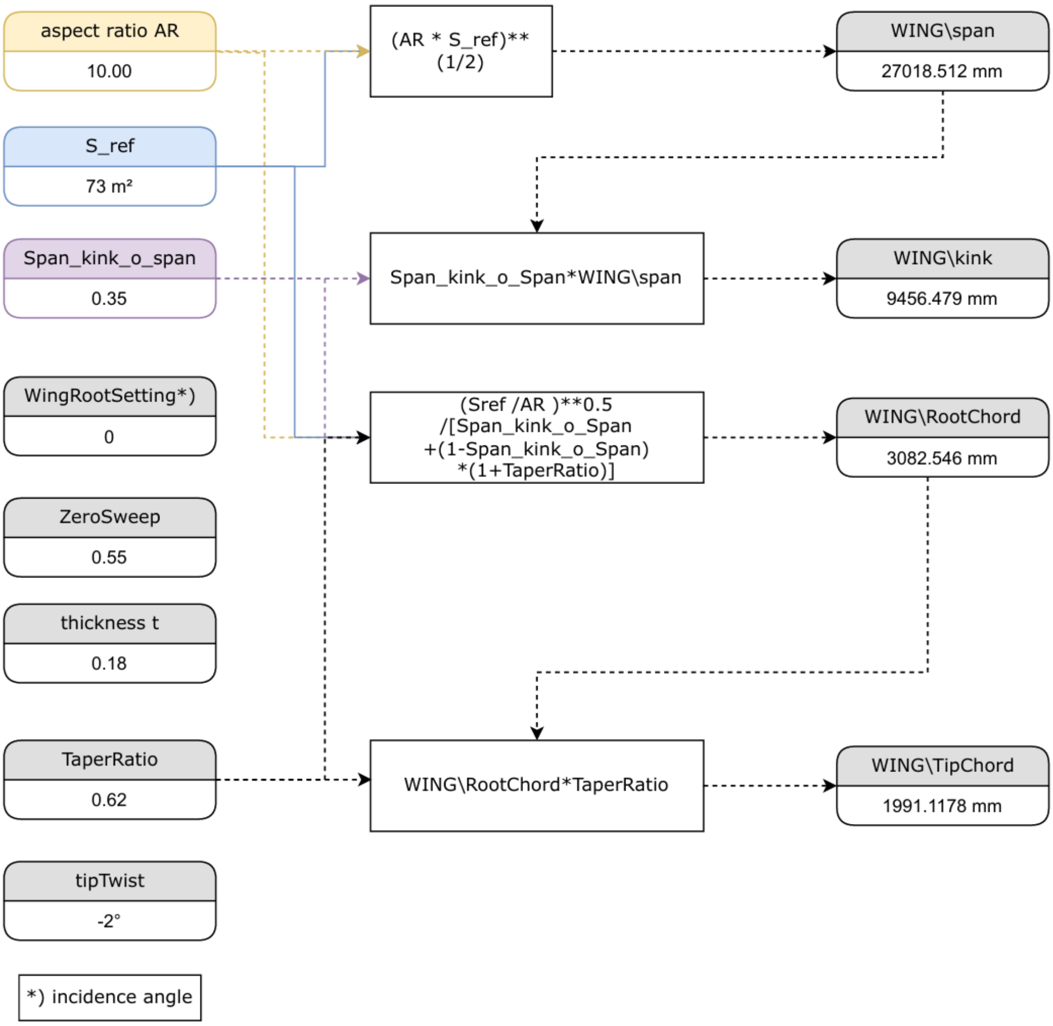
[57] Jochen Wild. *AeroForce: Thrust/Drag Bookkeeping and Aerodynamic Force Breakdown over Components*. Ed. by DLR-AS-TFZ. 2018.

A. Wing Model: CAD Parameters and Relations

CleanSetting_HERA.CATPart -

Config. 2 (after D5.1.2B spec.):

Parameter Relations Mapping



B. ActuatorDiskCoupling module: Python Code

```python
"""
The ActuatorDiskManager class ensures that the position and
orientation of any actuator disk boundaries, which might have
changed by deformations, are updated in FSDM prior to the next
TAU solver run.

REQUIRES:
  - unstructured meshes
  - CSM reference node(s) from which displacements and
    rotations can be sourced
"""

from FSDataManager import FSMeshEnums, FSVec3
from FSDLRControl import ArrayOps, UnstructMeshOps as MeshOps,
Utils as Utls
import numpy as np
from scipy.spatial.transform import Rotation as Rot


class ActuatorDiskManager:

  uiRootKey = 'ActuatorDiskCoupling'
  standardComponentTypes = ('Source', 'Target')

  @staticmethod
  def getUserInputAsDict(userInput, defaultValue):
    uiDict = userInput.Get(ActuatorDiskManager.uiRootKey,
    defaultValue)
    return uiDict

  def __init__(self, dataManager, userInput):
    self.dataManager = dataManager
    self.clac = dataManager.GetClac()
    self.log = Log()
    self.meshTools = MeshTools(self.dataManager)

    self.ui = ActuatorDiskManager.getUserInputAsDict(userInput,
    defaultValue=None)
    self.noComponentData = True

    # Initialise attributes and populate them with user input
    # data
    self.assemblePairingDefs() or self.log.error('Initialization
    of AD coupling data failed.')

    # Retrieve CFD/CSM scaling factor and offset from user
    # input
    uiKey = 'CFDCSMInterpolation/ScalingFactorsCSMDisplacementsToCFD'
    self.scalingFactorCSM2CFD = userInput.Get(uiKey, [1, 1, 1])

    self.log.info('Initialized.')
    return

  def iterateOverComponents(self, ofType=standardComponentTypes):
    for pairingDict in self.pairings.values():
      for component in pairingDict.values():
        for selectedType in list(ofType):
          if component.type == selectedType:
            yield component

  def defineComponents(self):
    for component in self.iterateOverComponents():
      success = component.establishMeshDataReferences()
```

```python
      if not success:
        return False

    # Remove noComponentData flag
    self.noComponentData = False
    return True

  def extractMeshDataOfComponents(self):
    components = self.iterateOverComponents()

    for component in components:
      success = component.extract()

      if not success:
        return False

    return True

  def synchronisePairings(self):
    for pairing in self.pairings.values():
      pairing['Target'].update(pairing['Source'])

  def injectIntoTargets(self):
    targets = self.iterateOverComponents(ofType=['Target'])
    updateInfo = ''
    for component in targets:
      normalVector, centerPoint =
      component.setActuatorDiskAttributes()
      n1, n2, n3 = normalVector
      c1, c2, c3 = centerPoint
      updateInfo += (
        f'\n\n\t"{component.pairingName}"\n'
        f'\t  new center point:  {c1:.5f} | {c2:.5f} | '
        f'{c3:.5f}\n'
        f'\t  new normal vector:   {n1:.5f} | {n2:.5f} | '
        f'{n3:.5f}'
      )
    return updateInfo

  def assemblePairingDefs(self):
    # Initialise a dict holding pairing definitions.
    self.pairings = {}

    # Iterate over pairings defined in the user input.
    for pairingName, pairingDefinition in self.ui.items():

      # Assert that two components ("Source", "Target") are
      # provided.
      assertionFailureMsg = (
        'ActuatorDiskCoupling failed: Each pairing must consist '
        + 'of two components: "Source" and "Target".'
      )
      assert len(pairingDefinition) == 2, assertionFailureMsg

      # Iterate over source and target components of the
      # pairing to populate pairingDict.
      pairingDict = {}
      for componentType in ActuatorDiskManager.standardComponentTypes:

        # Accessing the pairingDefintion dict., which is
        # defined by the user, with the standardComponentTypes
        # keys will cause a KeyError if the input is unexpected.
        try:
          componentDefinition = pairingDefinition[componentType]
        except KeyError as err:
          self.log.error(
            f'Erroneous user input: An actuator disk coupling'
            f'pair has to consist of one "Source" and one'
            f'"Target" component definition.'
          )
          raise err

        pairingDict[componentType] = PairingComponent(
          self,
          pairingName,
          componentType,
          componentDefinition
        )

      self.pairings[pairingName] = pairingDict
    return True

  def unifyADBoundaries(self):
    targets = self.iterateOverComponents(ofType=['Target'])
    for component in targets:
      component.overwriteExhaustCoordinates()

    return

  def run(self):
    self.log.info('Defining component data')
    if self.noComponentData:
      self.defineComponents() or print('Actuator Disk Manager:
      Definition of component data failed.')
    self.log.info('Extracting mesh data')
    success = self.extractMeshDataOfComponents()
    if not success:
      self.log.error('Source data extraction failed.')
      return
    self.log.info('Sync pairings')
    self.synchronisePairings()
    self.log.info('Inject AD data in target mesh')
    updateInfo = self.injectIntoTargets()

    msgPrefix = 'Actuator disk coupling completed.'
    self.log.info(msgPrefix + updateInfo)

    return


class PairingComponent:

  _standardSourceDatasets = ('Displacements', 'Rotations')

  def __init__(self, adManager, pairingName: str, componentType: str,
  componentDefinition: dict):
    self.adManager = adManager
    self.clac = adManager.clac
    self.meshTools = adManager.meshTools

    self.pairingName: str = pairingName
    self.type: str = componentType
    self.meshKey: str = componentDefinition['MeshKey']
    self.cellAttribute: str = componentDefinition['CellAttribute']
    self.cellAttributeValues: list =
    Utls.RangesToIntList(str(componentDefinition['CellAttributeValues']))

    self.mesh = None
    self.boundary = None
    self.ADAttributes = dict(centerPoint=None, normalVector=None)
    self.undeformedState = True
    self.undeformedADAttributes = dict(centerPoint=None,
    normalVector=None)
    self.meshQuantityValues = dict(Displacements=None,
    Rotations=None)

    self.logPrfx = f"\n\n[{self.adManager.clac.ProcID()}/"
      f"{self.adManager.clac.NProcs() - 1}] | {self.pairingName}:   "

    if self.type == 'Target' and self.cellAttribute == 'CADGroupID':
      assertionFailureMsg = (
        f'ActuatorDiskCoupling failed for pairing
        "{self.pairingName}":\n'
        + f'Incomplete definition of its target\'s CADGroupIDs.'
        f'Please provide both, '
        + f'the actuator disk\'s inlet and exhaust boundary
        f'marker.'
      )
      assert len(set(self.cellAttributeValues)) == 2,
      assertionFailureMsg

  def __repr__(self):
    cellAttrValueStr = [str(item) for item in self.cellAttributeValues]
    cellAttrValueStr = '\n\t\t- ' + '\n\t\t- '.join(cellAttrValueStr)
    infoString = (
      f'\n\nComponent Info for:\n'
      f'\t{self.type.upper()} of\n'
      f'\tpairing: {self.pairingName}\n'
      f'\tmesh key: {self.meshKey}\n'
      f'\tcell attribute: {self.cellAttribute}{cellAttrValueStr}'
    )
    return infoString

  def synchronizeDataInProcs(self, *npArrays, **reduce):
    synchronizedArray = []

    for npArray in npArrays:
      if reduce.get('reductionMethod') == 'max':
        gatheredArray = ArrayOps.ParallelMax(npArray,
        self.clac, skipLocalReductionOp=True)
      elif reduce.get('reductionMethod') is None:
        gatheredArray = ArrayOps.Gather(npArray, self.clac)

      synchronizedArray.append(ArrayOps.Broadcast(gatheredArray,
      self.clac))

    return synchronizedArray

  def establishMeshDataReferences(self):
    self.mesh = self.adManager.dataManager.GetMesh(self.meshKey)

    if self.type == 'Source':
      if self.mesh.HasCellAttribute(self.cellAttribute,
      FSMeshEnums.CT_Node):
        return True
      else:
        return False

    elif self.type == 'Target':

      if not self.mesh.HasPeriodicBoundaries(FSMeshEnums.CT_Node):
        self.adManager.log.error(f'ActuatorDiskCoupling: No
        actuator disks found in {self.meshKey}.')
        return False

      for bdry in self.mesh.GetPeriodicBoundaries(FSMeshEnums.CT_Node):
        # Check whether the current process has access to the
        # periodic cells
        if bdry.GetCells().IsInitialized():
          bdryMrkr = np.array(bdry.GetBoundaryMarkers().Buffer()).tolist()[0]
          # Check whether the boundary marker matches the user input
          if self.cellAttributeValues == bdryMrkr:
            self.boundary = bdry
            self.adManager.log.info(
              f'{self.pairingName} | Periodic boundary found
              for CADGroupIDs: {bdryMrkr}'
            )

      if not self.boundary.IsInitialized():
        errMsg = (
          f'ActuatorDiskCoupling: No boundaries found in
          {self.pairingName}\'s target mesh ',
          f'({self.meshKey=}) for \t {self.cellAttribute} =
          {self.cellAttributeValues}.'
        )
        self.adManager.log.error(errMsg)
        return False

      if not self.boundary.IsActuatorDisc():
        errMsg = (
          f'Boundary with {self.cellAttribute} =
          {self.cellAttributeValues} is not an actuator disk.'
        )
        self.adManager.log.error(errMsg)
      return False

    return True

def setActuatorDiskAttributes(self):
  nV = tuple(value for value in self.ADAttributes['normalVector'])
  cP = tuple(value for value in self.ADAttributes['centerPoint'])

  self.boundary.SetNormal(FSVec3(*nV))
  self.boundary.SetCenter(FSVec3(*cP))

  return nV, cP

def getActuatorDiskAttributes(self):
  if self.undeformedState:
    self.adManager.log.info(f'Getting AD data of disk'
    f'{self.pairingName}.\n{self.boundary.IsInitialized()=}')
    fsVecToNpArray = lambda vec: np.array([float(vec[n]) for n in
    range(len(vec))], dtype=np.float64)

    nV = self.boundary.GetNormal()
    cP = self.boundary.GetCenter()

    nV = fsVecToNpArray(nV)
    cP = fsVecToNpArray(cP)

    if self.clac and self.clac.NProcs() > 1:
      nV, cP = self.synchronizeDataInProcs(nV, cP,
      reductionMethod='max')

    self.undeformedADAttributes['normalVector'] = nV
    self.undeformedADAttributes['centerPoint'] = cP

    self.undeformedState = False

  else:
    nV = self.undeformedADAttributes['normalVector']
    cP = self.undeformedADAttributes['centerPoint']

  return nV, cP

def findMatchingNodesInMesh(self, userCellAttributeValues):
  # Get all cell attr. values in the mesh (shape = shape of
  # mesh's quantities)
  cellAttrValuesInMesh = MeshOps.GetCellAttributeValues(
    self.mesh, self.cellAttribute, cellType=FSMeshEnums.CT_Node
  )

  # Find mismatches: Difference between user-defined and
  # found nodeIDs
  cellAttrValuesNotFound = list(set(
    userCellAttributeValues).difference(set(cellAttrValuesInMesh)))
  remainingNodes = list(set(
    userCellAttributeValues).difference(set(cellAttrValuesNotFound)))

  # Option 1: None of user's nodes found in mesh
  if not any(remainingNodes):
    return [None]

  # Option 2: Not all of the user's nodes found in mesh
  elif any(cellAttrValuesNotFound):
    userCellAttributeValues = remainingNodes

    warning = (
      f'The following values of cell attribute'
      f'{self.cellAttribute!r} were not found '
      + f'in mesh {self.meshKey!r}: {cellAttrValuesNotFound}.\n'
      + f'Continuing with the remaining values: '
      + f'{remainingNodes}'
    )
    self.adManager.log.warning(warning)

  # Create boolean mapping of matching values in the shape
  # of cellAttrValuesInMesh
  matchingCellAttrValuesInMesh = np.isin(cellAttrValuesInMesh,
  userCellAttributeValues)

  return matchingCellAttrValuesInMesh

def getQuantityValues(self, datasetName, matchingNodes):
```

```
  quantities = self.mesh.GetUnstructQuantities(datasetName)

  for quantity in quantities:
    quantityData = MeshOps.GetQuantityValues(
      self.mesh, datasetName, quantity.GetName()
    )

    filteredQuantityValues = np.extract(condition=matchingNodes,
    arr=quantityData)

    yield filteredQuantityValues

def getValuesFromDatasets(self, datasets, nodes):
  for datasetName in datasets:
    quantityValues = np.array([value for value in
    self.getQuantityValues(datasetName, nodes)])
    nColumns = len(self.mesh.GetUnstructQuantities(datasetName))

    if quantityValues.any():
      quantityValues = np.stack(quantityValues, axis=-1)
    else:
      quantityValues = np.array([])

    # Gather the values
    gatheredValues = ArrayOps.Gather(quantityValues,
    self.clac).reshape(-1, nColumns)
    self.adManager.log.info(f'{gatheredValues=}')
    # Reduce to one value by taking the mean value over all
    # given nodes
    singularValues = np.mean(gatheredValues, axis=0)

    singularValues = ArrayOps.Broadcast(singularValues,
    self.clac)
    # If any of the broadcasted mean values is NaN that means
    # no quantity values were found in any process
    if np.isnan(singularValues).any():
      errMsg = (
        f"Specified value(s) for cell attribute "
        f"{self.cellAttribute!r} not found "
        + f"in mesh {self.meshKey!r}:
        f"{self.cellAttributeValues}. "
        + f"No other data source remaining.\nAborting "
        f"ActuatorDiskCoupling."
      )
      self.adManager.log.error(errMsg)
      return False

    print(self.logPrfx, f'Source data retrieved:\n{datasetName} : '
    f'{singularValues}')
    # Store the resulting value
    self.meshQuantityValues[datasetName] = singularValues

  return True

def extract(self):
  if self.type == 'Source':
    self.adManager.log.info('Extracting source data...')

    # Find the nodes matching the cell attribute (values)
    # specified by the user
    matchingNodes = self.findMatchingNodesInMesh(self.cellAttributeValues)

    # Obtain and store the values of the required datasets
    # (one value per quantity)
    datasets = PairingComponent._standardSourceDatasets
    self.getValuesFromDatasets(datasets, matchingNodes)
    self.adManager.log.info('Source data extraction completed.')
    return True

  if self.type == 'Target':
    self.adManager.log.info('Extracting target data...')
    # Normal vector and center point are retrieved only
    # once, in the undeformed state
    if self.getActuatorDiskAttributes():
      self.adManager.log.info('Target data extraction completed.')
      return True
    else:
      return False

def update(self, source):
```

```
if self.type == 'Target':

  # Reference the AD attributes  (center and normal)
  # from the target mesh
  normalVector = self.undeformedADAttributes['normalVector']
  centerPoint = self.undeformedADAttributes['centerPoint']

  # Reference the displacements and rotations vectors
  # from the source mesh
  displacementsVector =
  source.meshQuantityValues['Displacements']
  rotationsVector =
  source.meshQuantityValues['Rotations']

  # Reference scaling factor and translation vector
  scalingFactor = np.array(self.adManager.scalingFactorCSM2CFD)

  # Add displacement to the center point
  newCenterPoint = centerPoint + scalingFactor *
  displacementsVector

  # Define a rotation (euler angles [rad], extrinsic)
  rotationOperation = Rot.from_euler('xyz', rotationsVector)
  # Rotate the normal vector
  newNormalVector = rotationOperation.apply(normalVector)
  # Normalize the vector
  newNormalVector = newNormalVector /
  np.linalg.norm(newNormalVector)
  # Update the dictionaries holding the current AD attribute values
  self.ADAttributes.update({
    'centerPoint': newCenterPoint,
    'normalVector': newNormalVector
  })

return
```

C. Coupled CFD-CSM: FSDLRControl input file

```
ActuatorDiskCoupling:
  AD1:
    Source:
      MeshKey: CSMMESH
      CellAttribute: bdfGlobalNodeID
      CellAttributeValues: 4930001
    Target:
      MeshKey: CFDMESH
      CellAttribute: CADGroupID
      CellAttributeValues: 6,7
  AD2:
    Source:
      MeshKey: CSMMESH
      CellAttribute: bdfGlobalNodeID
      CellAttributeValues: 4930002
    Target:
      MeshKey: CFDMESH
      CellAttribute: CADGroupID
      CellAttributeValues: '8, 9'
  AD3:
    Source:
      MeshKey: CSMMESH
      CellAttribute: bdfGlobalNodeID
      CellAttributeValues: 4930003
    Target:
      MeshKey: CFDMESH
      CellAttribute: CADGroupID
      CellAttributeValues: 10-11

#Variable declarations
CFDWingCADGroupIDs: &CFDWingCADGroupIDs
  - 3
  - 4
  - 5
  - 12
CFDSymCADGroupIDs: &CFDSymCADGroupIDs
  - 1
CFDAD1CADGroupIDs: &CFDAD1CADGroupIDs
  - 6
  - 7
CFDAD2CADGroupIDs: &CFDAD2CADGroupIDs
  - 8
  - 9
CFDAD3CADGroupIDs: &CFDAD3CADGroupIDs
  - 10
  - 11

#Header
CouplingMethod: &CouplingMethod mls
Scenario: SteadyCFDCSMCoupling
EnableDebugOutput: true
FSLogLevel: 0
# FSLogLevelFile: 2
#FilenamePartsSeparator: _
#OutputFilesPrefix: Run1

CFDDomain:
  MeshKey: CFDMESH
  OutputFilesPrefix: CFDDomain/cfdDomain
  MeshImport:
    MeshImportFormat: NativeTauIO
    MeshImportFilename: ../CFD_Input/CleanSetting_HERA_240530_
      Config2_3AD_HEX_SymY0.cdf
    SolutionImportFilename: ../CFD_Input/240625_Var00-Baseline-
      Test/HERA-Wing_ansa_3AD-Conf2-Trimmed-1g.pval.39816(Config2
      _3AD_HEX_SymY0)
    RepartitioningMethod: private #zoltan
    SwitchToStandardNaming: true
    #MergeAllNonscalarDatasets: true
    #InitUndefoCoordsFromCoordsIfMissing: true
    RemoveMeshAttributes: &RemoveMeshAttributes
    - parameters
    - parameters_pval
    - parameters_surf
  MeshExport:
    MeshExportFormat: hdf5
    #ExtraMeshExportParameters: {}
  ComponentDefinitions: &CFDComponentDefinitions
    CFD_Wing: &CFDWing
      CellAttribute: CADGroupID
      CellAttributeValues: *CFDWingCADGroupIDs
    CFD_AD1: &CFDAD1
      CellAttribute: CADGroupID
      CellAttributeValues: *CFDAD1CADGroupIDs
    CFD_AD2: &CFDAD2
      CellAttribute: CADGroupID
      CellAttributeValues: *CFDAD2CADGroupIDs
    CFD_AD3: &CFDAD3
      CellAttribute: CADGroupID
      CellAttributeValues: *CFDAD3CADGroupIDs
  ComponentMeshExport:
    MeshExportFormat: tecplot
  SurfaceDefinition:
    CellAttribute: CADGroupID
    CellAttributeValues: 3-12
  SurfaceMeshExport:
    MeshExportFormat: tecplot

CSMDomain:
  MeshKey: CSMMESH
  OutputFilesPrefix: CSMDomain/csmDomain
  MeshImport:
    MeshImportFormat: nastranbdf
    MeshImportFilename: sol101.nas
    RepartitioningMethod: rcb
  MeshExport:
    #MeshExportFormat: hdf5
  ComponentDefinitions: &CSMComponentDefinitions
    CSM_Wing: &CSMWing
      CellAttribute: CADGroupID
      CellAttributeValues: 408
    CSM_AD1: &CSMAD1
      CellAttribute: CADGroupID
      CellAttributeValues: 451
    CSM_AD2: &CSMAD2
      CellAttribute: CADGroupID
      CellAttributeValues: 452
    CSM_AD3: &CSMAD3
      CellAttribute: CADGroupID
      CellAttributeValues: 453
  ComponentMeshExport:
    MeshExportFormat: tecplot
    #ExtraMeshExportParameters: {}
    ExportComponents:
    - CSM_Wing

CFDCSMCoupling:
  CSMSolver: NastranSteadyDirectSolver #NastranSteadyDirectSolver
  CFDSolver: SteadyTauSolver
  CFDMeshDeformer: ea
  MaximumNumberOfCouplingSteps: 40
  ExportIntermediateCouplingSolutions: true
  ComputeInitialCFDSolution: true
  #CFDBCParameterUpdatesInitialRun:
  CFDParameterUpdatesInitialRun:
  - Maximal time step number: 10 #originally 500
    CFL number (coarse grids): 5
    CFL number: 5
  CFDParameterUpdatesEachCouplingIter:
  - CFL number (coarse grids): 10
    CFL number: 10
    Maximal time step number: 2000
  #CFDBCParameterUpdatesEachCouplingIter:
  ComputeFinalCFDSolution: true
  #CFDBCParameterUpdatesFinalRun:
  CFDParameterUpdatesFinalRun:
  - Maximal time step number: 8000
    Error for Cauchy convergence control: 5E-5 1E-4
    Number of samples for Cauchy convergence: 250
  CorrectSymmetryCADGroupID: *CFDSymCADGroupIDs

CFDCSMConvergenceChecker:
    ConvergenceThresholdDeformation: 1e-4
    ConvergenceThresholdLoads: 1e-4
    DeformationQuantities:
      - [Displacements, DisplacementZ]
    LoadsQuantities:
      - [Forces, ForceZ]
     # - [Moments, MomentY]

CSMUnderrelaxation:
  RelaxationFactor: 0.55
  # RelaxationDatasets: []

MeshQuantityMonitoring:
  # OutputFilesPrefix: MeshQuantityMonitoring
  Monitorings:
    - MeshKey: CSMMESH
      Quantities:
      - [Coordinates, CoordinateX]
      - [Coordinates, CoordinateY]
      - [Coordinates, CoordinateZ]
      - [Displacements, DisplacementX]
      - [Displacements, DisplacementY]
      - [Displacements, DisplacementZ]
      - [Rotations, RotationX]
      - [Rotations, RotationY]
      - [Rotations, RotationZ]
      - [Forces, ForceX]
      - [Forces, ForceY]
      - [Forces, ForceZ]
      - [Moments, MomentX]
      - [Moments, MomentY]
      - [Moments, MomentZ]
      NodeIDs:
      - 4930001
      - 4930002
      - 4930003
      - 4930101
      - 4930102
      - 4930103
      - 4000006
      - 4000301
      - 4000848
      - 4000734
      - 4000146
      - 4000515
      - 4000761
      - 4000993
      - 4000250
      - 4000655
      - 4001186
      - 4000799
      - 4930004
      - 4930005
      - 4930006
      - 4930007
      - 4930008
      - 4930009
      - 4930010
      - 4930011
      - 4930012
      - 4930013
      - 4930014
      - 4930015
      - 4930016
      - 4930017
      - 4930018
      - 4930019
      - 4930020
      - 4930021
      - 4930022
      - 4930023
      - 4930024
      - 4930025
      - 4930026
      - 4930027
      - 4930028
      - 4930029
      - 4930030
      - 4930031
      - 4930032
      - 4930033
      - 4930034
      NodeIDAttribute: bdfGlobalNodeID

CFDSolver:
  TAUParameterFilename: cfd.taupara
  TAUSolverMode: turb1eq
  # UseExternalMotion: false
  # CalcLoadsFromPressure: false
  OutputFilesPrefix: CFDSolver/cfdSolver


CSMSolver:
  OutputFilesPrefix: CSMSolver/csmOutput
  WriteRestartOutput: true
  ForkModule: subprocess #os oder subprocess
  #DebugOutput: true
  #ExecutablePath: /sw/Core/Nastran/2021.1/bin/nast20211
  JobFilename: sol101.nas
  DeformationsFilename: sol101.f06
  AssignComponentsFromSets: false
  AssignComponentsOnlyForSelectedSets:
  - 408
  - 451
  - 452
  - 453
  LoadsFilename: aeroloads.bdf
  LoadSetId: 101
  SubcaseId: 1
  #ExportModalSolution: true
  #ModalSolutionFileFormat: tecplot
  #UseModeIDs: 1-10
  NodeIDAttributeName: bdfGlobalNodeID
#  ExtraLoads: # format: {<node ID>: [ForceX, ForceY, ForceZ, MomentX, MomentY, MomentZ]}
#    4930034: [0.,0.,8.E+3,0.,0.,0.]
#    4930032: [0.,0.,8.E+3,0.,0.,0.]
#    4930030: [0.,0.,8.E+3,0.,0.,0.]
#    4930028: [0.,0.,8.E+3,0.,0.,0.]
#    4930026: [0.,0.,8.E+3,0.,0.,0.]
#    4930024: [0.,0.,8.E+3,0.,0.,0.]
#    4930020: [0.,0.,-16.E+3,0.,0.,0.]
#    4930018: [0.,0.,-16.E+3,0.,0.,0.]
#    4930016: [0.,0.,-16.E+3,0.,0.,0.]

CFDMeshDeformation:
  OutputFilesPrefix: CFDMeshDeformer/cfdMeshDefo
  UseNearestNeighbourCorrection: false
  WithAutoRepair: true
  ExportNegativeCellMesh: true
  LogPrefix: CFDMeshDeformer/cfdMeshDefo
  MatrixFilePrefix: CFDMeshDeformer/cfdMeshDefo_
  DeformationGroups:
    Wing:
      TargetBoundaryValues: *CFDWingCADGroupIDs
      BoundaryType: Standard
      #MaxBasePoints: 1000
      #RadialBasisFunction: cubic-volume-spline
      #RadiusFullWeight: 0.3
      #RadiusZeroWeight: 20.
      #ReductionMethod: Equidistant
    Sym:
      TargetBoundaryValues: *CFDSymCADGroupIDs
      BoundaryType: NoNormalMovement
      #RadialBasisFunction: cubic-volume-spline
      #RadiusFullWeight: 0.3
      #RadiusZeroWeight: 20.
      #ReductionMethod: Equidistant
    AD1:
      TargetBoundaryValues: 6-11
      BoundaryType: Standard
    FF:
      TargetBoundaryValues: 2
      BoundaryType: Standard


CFDCSMInterpolation:
  #EnableDebugOutput: false
  OutputFilesPrefix: CFDCSMInterpolator/cfdcsmInterpol
  #MirrorCFDCouplingMesh: false
  #MirrorPlane: XZ
  TranslationCFDToCSMInCSMSys: [-118.,0.,30.]
  #RotationCFDToCSM: from_euler('y', 90., degrees=True)
  ScalingFactorsCFDGeometryToCSM: [1.E+3, 1.E+3, 1.E+3]
  ScalingFactorsCFDLoadsToCSM: [1., 1., 1.]
  ScalingFactorsCSMDisplacementsToCFD: [1.E-3, 1.E-3, 1.E-3]
  CFDCouplingComponentDefinitions:
    CFD_Wing: *CFDWing
    CFD_AD1: *CFDAD1
    CFD_AD2: *CFDAD2
    CFD_AD3: *CFDAD3
  CSMCouplingComponentDefinitions:
    CSM_Wing: *CSMWing
    CSM_AD1: *CSMAD1
    CSM_AD2: *CSMAD2
    CSM_AD3: *CSMAD3
  CFDCSMPairings:
    Wing:
      CouplingMeshes: CFD_Wing, CSM_Wing
      CouplingMethod: *CouplingMethod
      CouplingMethodParams:
        numNextCenters: 100
    AD1:
      CouplingMeshes: CFD_AD1, CSM_AD1
      CouplingMethod: nn
      CouplingMethodParams:
        considerRotationsNonlinear: false
    AD2:
      CouplingMeshes: CFD_AD2, CSM_AD2
      CouplingMethod: nn
      CouplingMethodParams:
        considerRotationsNonlinear: false
    AD3:
      CouplingMeshes: CFD_AD3, CSM_AD3
      CouplingMethod: nn
      CouplingMethodParams:
        considerRotationsNonlinear: false
```

D. Design Variations: Aerodynamic Performance

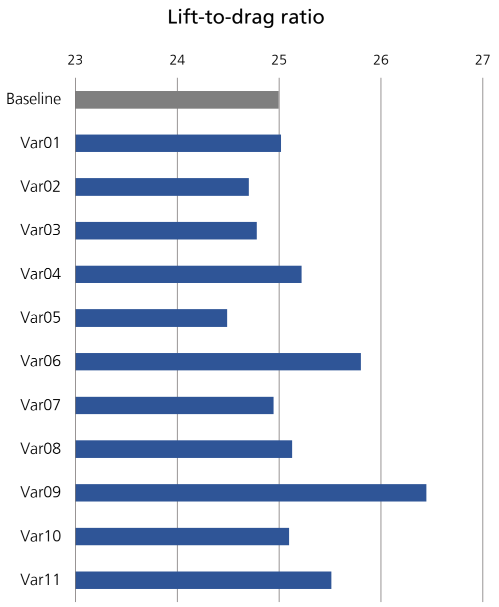


Figure D.1.: Lift-to-drag ratios of all design variations

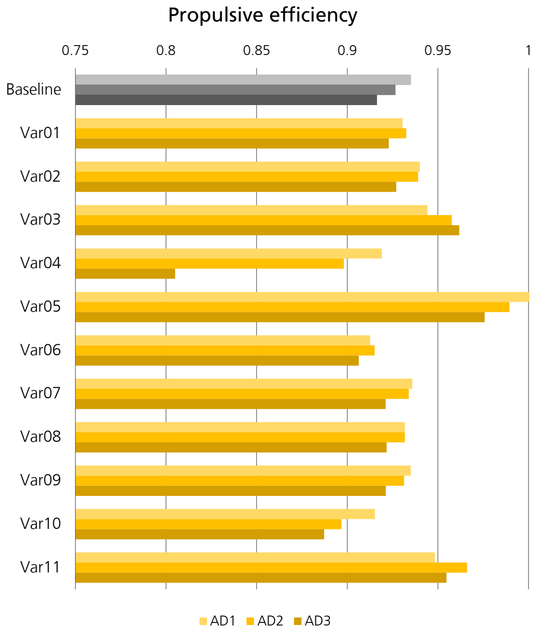


Figure D.2.: Propulsive efficiencies of all design variations

E. Actuator Disk: Propeller Input Data

r/R	*c/R*	*beta*
0.000	0.0000	90.000
0.020	0.0000	90.000
0.040	0.0000	90.000
0.060	0.0000	90.000
0.080	0.0000	90.000
0.100	0.0000	90.000
0.120	0.0000	90.000
0.140	0.0000	90.000
0.160	0.0000	90.000
0.180	0.0000	90.000
0.200	0.0628	77.948
0.220	0.0710	77.146
0.240	0.0789	76.357
0.240	0.0789	76.357
0.260	0.0860	75.593
0.280	0.0922	74.861
0.300	0.0974	74.164
0.320	0.1014	73.494
0.340	0.1047	72.842
0.360	0.1073	72.191
0.380	0.1097	71.528
0.400	0.1122	70.837
0.420	0.1149	70.107
0.440	0.1180	69.333
0.460	0.1215	68.522
0.480	0.1252	67.687
0.500	0.1291	66.840
0.520	0.1330	65.996
0.540	0.1367	65.160
0.560	0.1402	64.335
0.580	0.1434	63.524
0.600	0.1463	62.725
0.620	0.1489	61.935
0.640	0.1512	61.152
0.660	0.1533	60.369
0.680	0.1552	59.581
0.700	0.1569	58.786
0.720	0.1584	57.990
0.740	0.1596	57.202
0.760	0.1605	56.430
0.780	0.1609	55.679
0.800	0.1607	54.945
0.820	0.1597	54.219
0.840	0.1574	53.492
0.860	0.1535	52.756
0.880	0.1471	52.003
0.900	0.1375	51.219
0.920	0.1238	50.384
0.940	0.1054	49.475
0.960	0.0826	48.481
0.980	0.0560	47.409
1.000	0.0273	46.291

Table E.1.: Geometric propeller data

r/R	*alpha*	*cl*	*cd*
0.17	-100.0000	0.0000	1.60000
0.17	-45.0000	-0.8000	0.25000
0.17	-15.0000	-0.5632	0.10944
0.17	-9.0000	-0.5444	0.07900
0.17	-8.0000	-0.5097	0.05339
0.17	-7.0000	-0.4348	0.03634
0.17	-6.0000	-0.3337	0.02535
0.17	-5.0000	-0.2210	0.01849
0.17	-4.0000	-0.1053	0.01463
0.17	-3.0000	0.0141	0.01130
0.17	-2.0000	0.1343	0.01001
0.17	-1.0000	0.2528	0.00825
0.17	0.0000	0.3670	0.00831
0.17	1.0000	0.4831	0.00828
0.17	2.0000	0.5988	0.00827
0.17	3.0000	0.7139	0.00842
0.17	4.0000	0.8205	0.00888
0.17	5.0000	0.9250	0.01234
0.17	6.0000	1.0308	0.01431
0.17	7.0000	1.1319	0.01633
0.17	8.0000	1.2243	0.01884
0.17	9.0000	1.3090	0.02159
0.17	10.0000	1.3719	0.02584
0.17	11.0000	1.4391	0.02785
0.17	12.0000	1.4797	0.03304
0.17	13.0000	1.4969	0.04116
0.17	14.0000	1.4966	0.05229
0.17	45.0000	0.8000	0.25000
0.17	100.0000	0.0000	1.60000

r/R	*alpha*	*cl*	*cd*
0.31	-100.0000	0.0000	1.60000
0.31	-45.0000	-0.8000	0.25000
0.31	-15.0000	-0.5632	0.10944
0.31	-9.0000	-0.5444	0.07900
0.31	-8.0000	-0.5097	0.05339
0.31	-7.0000	-0.4348	0.03634
0.31	-6.0000	-0.3337	0.02535
0.31	-5.0000	-0.2210	0.01849
0.31	-4.0000	-0.1053	0.01463
0.31	-3.0000	0.0141	0.01130
0.31	-2.0000	0.1343	0.01001
0.31	-1.0000	0.2528	0.00825
0.31	0.0000	0.3670	0.00831
0.31	1.0000	0.4831	0.00828
0.31	2.0000	0.5988	0.00827
0.31	3.0000	0.7139	0.00842
0.31	4.0000	0.8205	0.00888
0.31	5.0000	0.9250	0.01234
0.31	6.0000	1.0308	0.01431
0.31	7.0000	1.1319	0.01633
0.31	8.0000	1.2243	0.01884
0.31	9.0000	1.3090	0.02159
0.31	10.0000	1.3719	0.02584
0.31	11.0000	1.4391	0.02785
0.31	12.0000	1.4797	0.03304
0.31	13.0000	1.4969	0.04116
0.31	14.0000	1.4966	0.05229
0.31	45.0000	0.8000	0.25000
0.31	100.0000	0.0000	1.60000

Table E.2.: Aerodynamic propeller data I

r/R	*alpha*	*cl*	*cd*
0.75	-100.0000	0.0000	1.60000
0.75	-45.0000	-0.8000	0.25000
0.75	-15.0000	-0.5632	0.10944
0.75	-9.0000	-0.5444	0.07900
0.75	-8.0000	-0.5097	0.05339
0.75	-7.0000	-0.4348	0.03634
0.75	-6.0000	-0.3337	0.02535
0.75	-5.0000	-0.2210	0.01849
0.75	-4.0000	-0.1053	0.01463
0.75	-3.0000	0.0141	0.01130
0.75	-2.0000	0.1343	0.01001
0.75	-1.0000	0.2528	0.00825
0.75	0.0000	0.3670	0.00831
0.75	1.0000	0.4831	0.00828
0.75	2.0000	0.5988	0.00827
0.75	3.0000	0.7139	0.00842
0.75	4.0000	0.8205	0.00888
0.75	5.0000	0.9250	0.01234
0.75	6.0000	1.0308	0.01431
0.75	7.0000	1.1319	0.01633
0.75	8.0000	1.2243	0.01884
0.75	9.0000	1.3090	0.02159
0.75	10.0000	1.3719	0.02584
0.75	11.0000	1.4391	0.02785
0.75	12.0000	1.4797	0.03304
0.75	13.0000	1.4969	0.04116
0.75	14.0000	1.4966	0.05229
0.75	45.0000	0.8000	0.25000
0.75	100.0000	0.0000	1.60000

Table E.3.: Aerodynamic propeller data II

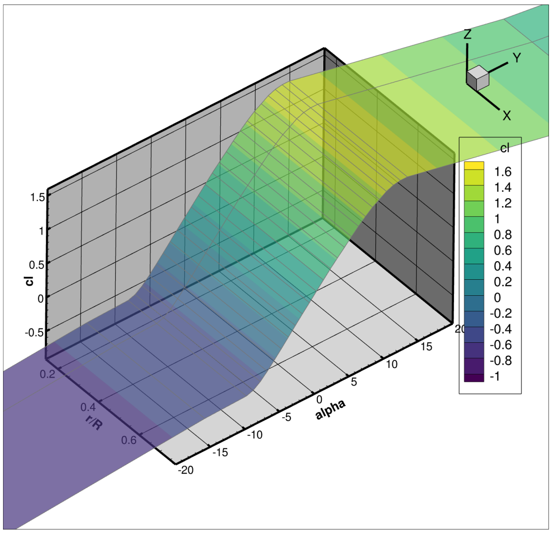


Figure E.1.: Lift coefficient over angle of attack (alpha) and radial coordinate r/R

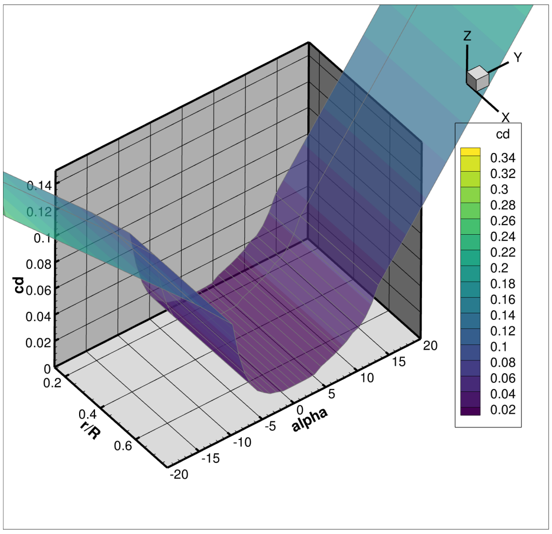


Figure E.2.: Drag coefficient over angle of attack (alpha) and radial coordinate r/R